Variational Bayesian Methods for Discovering Gene Expression Mechanisms from Single-Cell Transcriptomic Data

by

Yichen Gu

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Electrical and Computer Engineering)
in the University of Michigan
2025

Doctoral Committee:

Associate Professor Joshua Welch, Chair
Professor David Blaauw
Professor Satish Narayanasamy
Professor Clayton Scott

Yichen Gu
gyichen@umich.edu
ORCID iD: 0000-0001-6585-9236

© Yichen Gu 2025

DEDICATION

This dissertation is dedicated to my family, friends, colleagues, and all those committed to advancing scientific discovery through improved computational methods.

ACKNOWLEDGEMENTS

This dissertation presents the research work of Yichen Gu. I would like to sincerely express my heartfelt gratitude to my Ph.D. advisor, Dr. Joshua Welch, and all the members of my dissertation committee (alphabetically ordered by last name), Professor David Blaauw, Professor Satish Narayanasamy and Professor Clayton Scott, for their invaluable guidance throughout my Ph.D. program. I would like to thank National Institutes of Health for funding my research projects. Additionally, I would like to thank the ECE graduate program manager, Kristen Thornton, for help with my program. I wish to acknowledge my colleagues and collaborators (alphabetically ordered by last name), Dr. Andrea Bejarano, Professor Reetuparna Das, Dr. Cindy DeLong, Dr. Timothy Dunn, Gaurav Gadhvi, Dr. Chao Gao, Dr. Matthew Karikomi, Dr. April Kriebel, Dr. Takamasa Kudo, Dr. Hojae Lee, Dr. Kun (Justin) Lee, Dr. Romain Lopez, Chen Li, Dr. Jialin Liu, Lu Lu, Weizhou Qian, Andrew Robbins, Dr. Hari Sadasivan, Yuxuan Song, Dr. Yutong Wang, Yichen Wang, Crystal Wen, Mingjia Yao, Neil Zhao, for contributing to my research projects and providing insightful feedback. Finally, I am deeply grateful to my family and friends (alphabetically ordered by last name), Dr. Xing Chen, Liangyu Cui, Dr. Zichen Fan, Jingwen Gong, Yufeng Gu, Ziqian He, Yihuan Huang, Dr. Shiting Li, Minghao Liu, Dr. Yan Long, Yizhou Qiu, Yutong Wang, Dr. Zhehong Wang, Yancheng Yang, Chi Zhang, Dr. Qirui Zhang, Yujie Zhou, for their unwavering support during the most challenging times. This dissertation would not have been possible without your care and encouragement.

TABLE OF CONTENTS

DEDICATION ii

ACKNOWLEDGEMENTS iii

LIST OF FIGURES vii

LIST OF TABLES xvi

LIST OF APPENDICES xvii

LIST OF ACRONYMS xviii

ABSTRACT xx

CHAPTER

1 Introduction 1

1.1 Background 1
1.2 Single-Cell RNA Sequencing 2
1.3 Spatial Transcriptomics 3
1.4 Single-Cell RNA Sequencing with Perturbation 5
1.5 Single-Cell RNA Data Generation 6
1.6 Computational Methods for Single-Cell Transcriptomics 7
1.6.1 Data-Driven Approach to Studying Cell Identity. 8
1.6.2 Understanding Temporal Dynamics During Cell Development. . . . 9
1.6.3 Understanding Spatial Processes of Cell Differentiation. 9
1.6.4 Causal Discovery. 10
1.7 Methodology Overview 11
1.8 Chapter Overview 12

2 Inferring Temporal Mechanism of Gene Expression 14

2.1 Motivation 14
2.2 Technical Contributions 16
2.3 RNA Velocity 16
2.4 Methods 17
2.4.1 Variational Mixture of ODEs 17
2.4.2 Extension of VeloVAE 22

2.4.3 Branching ODE . . . 26
2.4.4 Evaluation Metrics . . . 28
2.5 Experiments . . . 30
2.5.1 Preliminary Benchmark Study . . . 30
2.5.2 Modeling Multi-Lineage RNA Velocity . . . 32
2.5.3 Cell Type Transition Graph Inference and Branching Differential Equation Model . . . 35
2.5.4 VeloVAE Accurately Models Cell Differentiation Across the Whole Mouse Brain. . . . 37
2.5.5 VeloVAE Accurately Models Organ Development in Whole Mouse Embryos. . . . 39
2.6 Discussion . . . 40
3 Inferring Spatial Mechanism of Gene Expression . . . 42
3.1 Motivation . . . 42
3.2 Technical Contributions . . . 43
3.3 RNA Velocity and Cell Development in Space . . . 44
3.4 Methods . . . 45
3.4.1 RNA Velocity in Spatial Contexts . . . 45
3.4.2 Modeling Spatial Data with Graph Neural Networks . . . 45
3.4.3 TopoVelo: Topological Velocity Inference from Spatial Transcriptomic Data . . . 47
3.4.4 Modeling Real Tissue Growth with Cell Velocity . . . 50
3.4.5 Evaluation Metrics . . . 51
3.4.6 Interpreting Graph Attention via Cell Influence Score . . . 52
3.4.7 Interpreting Results from 3D Neural Tube Development . . . 53
3.4.8 Data Preprocessing . . . 56
3.5 Experiments . . . 58
3.5.1 Simulation Study . . . 58
3.5.2 Benchmark Study on Real Data . . . 61
3.5.3 TopoVelo Models Embryonic Mouse Brain Development from Slide-Tags Data . . . 65
3.5.4 TopoVelo Reveals Signatures of Mouse Neural Tube Closure . . . 67
3.5.5 TopoVelo Maps Spatial Patterns of Early Differentiation in Human Embryoid Bodies . . . 70
3.6 Discussion . . . 73
4 Inferring Causal Mechanism of Gene Expression . . . 75
4.1 Motivation . . . 75
4.2 Technical Contributions . . . 76
4.3 Causal Discovery . . . 77
4.3.1 Problem Setup . . . 77
4.3.2 Differentiable Causal Discovery . . . 78
4.4 Methods . . . 83
4.4.1 Probabilistic Modeling of f-DAGs . . . 83

4.4.2 Bayesian Causal Discovery of DAGs 87
4.4.3 Amortized Bayesian Causal Discovery of Extended Factor Graphs . 88
4.5 Experiments 92
4.5.1 Simulation Study 92
4.5.2 Application to Real Cellular Perturbation Screen. 94
4.6 Discussion 96

5 Summary and Future Work 98

5.1 Summary of the Dissertation 98
5.2 Contribution and Broader Impact 100
5.3 Future Work 101
5.4 Final Remarks 105

APPENDICES 106

BIBLIOGRAPHY 175

LIST OF FIGURES

FIGURE

2.1 **Diagram of VeloVAE Model. (a) Biochemical process** The panel depicts the sequential process of gene transcription, RNA splicing and degradation. **(b) VAE architecture.** In the basic model, an encoder network infers a latent time from all u and s values for each cell. The data are reconstructed from inferred time using ordinary differential equation (ODE) solutions, whose analytical form is known. In the full model, an encoder network infers both latent time and latent cell state $\mathbf{c}$. A decoder network generates the transcription rates $\boldsymbol{\rho}$, which are unique for each cell and each gene. The data are then reconstructed from t and $\boldsymbol{\rho}$ using the kinetic function. 19

2.2 **Graphical Model of Variational Mixture of ODE.** Observed variables are colored gray. Dashed arrows indicate sampling. 21

2.3 **Benchmarking Results across 12 Datasets. (a)-(d)** Quantitative performance comparison. We report the correlation between inferred latent time and true capture time (when available) **(a)**, mean squared error (MSE) between the observed and predicted (u, s) counts for continuous-count models **(b)**, gene log likelihood for discrete-count models **(c)** and generalized cross-boundary direction correctness across 12 benchmarking datasets **(d)**. 31

2.4 **VeloVAE Significantly Models Complex Gene Expression Dynamics.** Examples of individual genes fit by different RNA velocity methods for a late-induced gene **(a)** and early-repressed gene **(b)** in pancreas and a gene with branching dynamics in subsampled mouse brain **(c)**. Gene fits are shown for both u and s values, with inferred latent time on the x-axis. Fitted values from scVelo and UniTVelo are shown as lines, with observed data values shown as points. Only fitted values are shown for the other methods, because the fit is a point cloud (rather than a line) that would completely cover the observed data. Points are colored by published cluster assignments, with the same colors as in the corresponding UMAP plots. The arrow length and direction indicate the velocity inferred for each cell. 33

2.5 **Cell Type Transition Graph Inference and Branching Differential Equation Model.** **(a)-(c)** Inferred cell type transition graphs from pancreas **(a)**, iPSC **(b)** and subsampled mouse brain **(c)** datasets. In each graph, a vertex represents a cell type and a directed edge points from a progenitor cell type to its immediate descendant(s). **(e)-(i)** Examples of individual genes fit by the branching ODE model for pancreas **(d)**, iPSC **(f)** and subsampled mouse brain **(h)** datasets. Each column represents a gene and plots the unspliced count, spliced count and RNA velocity versus time from top to bottom. The branching ODE fits are shown as solid lines. Cell-type-specific rate parameters inferred by branching ODE model for pancreas **(e)**, iPSC **(g)** and subsampled mouse brain **(i)** datasets are also shown. Each column represents a gene and plots the transcription (α), splicing (β) and degradation (γ) rates from top to bottom. Each point represents a cell type, the types are arranged in chronological order, and progenitor-descendant relationships are indicated with lines. 36

2.6 **VeloVAE Resolves Cellular Dynamics of Multiple Lineages across the Entire Developing Mouse Brain.** **(a)** t-SNE plots colored by published cell types overlaid with velocity inferred by VeloVAE. **(c)** Examples of individual genes fit by VeloVAE. Gene fits are shown for both u and s values, with inferred latent time on the x-axis. Only fitted values are shown for VeloVAE, because the VeloVAE fit is a point cloud (rather than a line) that would completely cover the observed data. Points are colored by published cluster assignments, with the same colors as in the corresponding UMAP plots. Note that the x-axis values are different for scVelo and VeloVAE because they infer different latent times. The arrow length and direction indicate the velocity inferred for each cell. **(d)** t-SNE plots colored by true capture time and latent time inferred by VeloVAE. **(e)-(f)** t-SNE plots colored by the cell state uncertainty **(e)** and cell time variance **(f)** from VeloVAE. 38

2.7 **Whole Mouse Embryo Analysis.** **(a)** UMAP of entire mouse embryo dataset, colored by broad cell class. **(b)-(c)** UMAP plots colored by published cell types overlaid with velocity inferred by VeloVAE for neuronal **(b)** and mesenchymal **(c)** trajectories. The UMAP coordinates are computed using only cells from each trajectory, but the velocity estimates are from a single VeloVAE model fit on all cells. **(d)-(e)** VeloVAE fits for selected genes from the mesenchymal **(d)** and neuronal **(e)** trajectories. 40

3.1 **Diagram of TopoVelo model. (a)** System of differential equations that describes how unspliced pre-mRNA (u) and spliced mature mRNA (s) values are related during gene expression. ρ is a transcription rate parameter that is different for every cell and every gene. β and γ are splicing and degradation rates, which are different for each gene but shared across cells. **(b)** Diagram of spatial dependence in rate parameters. TopoVelo extends previous RNA velocity models by allowing transcription rate parameters to be a function of the states of neighboring cells in a tissue, which may influence each other through mechanisms such as ligand-receptor signaling. **(c)** Diagram of TopoVelo neural network architecture. TopoVelo takes a graph as input. In this graph, nodes represent cells or spots; u, s, and spatial coordinates are node features; and edges represent spatial proximity. A graph neural network infers cell state and cell time jointly across cells (encoder network). Another graph neural network predicts transcription rates ρ from the states of each cell and its neighbors. The phase of each gene is also predicted (w). The u and s counts are reconstructed using the analytical solution to the differential equation from (a). 46

3.2 **Illustration of Neural Tube Unfolding. (a)** Animation of a mouse embryo with a folded neural tube (left) and an unfolded neural tube (right). **(b)** Proposed algorithm to unfold a neural tube. 55

3.3 **TopoVelo Accurately Models Spatial Dynamics in Simulated Datasets. (a)** Simulated datasets with bidirectional(top) and radial(bottom) growth patterns. The true velocity stream is plotted over spatial coordinates colored by the true time values. Note that these stream plots indicate cell velocity in physical coordinates, not RNA velocity in abstract UMAP coordinates. **(b)** Quantitative performance on Simulated Datasets. The top row shows six metrics used for benchmarking. Higher values are better for all metrics except for reconstruction error. The bottom row contains line plots showing k-CBDir using k-step neighbors with k up to five. Note that we computed two versions of k-CBDir, one using spatial nearest neighbors and one using nearest neighbors in gene expression space. **(c)** Comparison of inferred cell velocity. Streamlines with arrow indicate the direction of cell velocity in the physical space. Coordinates are colored by inferred time values. 59

3.4 **Quantitative Comparison of Model Performance on Real Datasets. Top**: bar plots comparing methods across four benchmark datasets. Each color represents a method and each bar group represents a dataset. **Bottom**: line plot showing k-Step CBDir metric. Each color represents a method. 62

3.5 **Qualitative Comparison of TopoVelo and Existing RNA Velocity Methods. (a)** Velocity stream plots shown on spatial (physical) coordinates. **(b)** Cell coordinates colored by inferred latent cell time. The last column entitled "Expected Temporal Order" shows a rough approximation of the expected time based on prior knowledge that cortex grows from ventricular zone to layers of excitatory neurons and gut and lung form from the inside out. **(c)** Velocity stream plots on the thymus tissue. **(d)** Thymus cell coordinates colored by inferred latent cell time. The last column entitled "Expected Temporal Order" is the ground truth by the CMA score. **(e)** Morphology image of the thymus[1]. . 63

3.6 **TopoVelo analysis of mouse neural tube closure.** **(a)** Schematic of registering whole mouse embryo Slide-seq data. Each dot represents a slide-seq spot and is colored by its cell type. Red represents the brain and neural tube. Grey represents all other cell types. **(b)** Quiver plot of the cell velocity of the brain and neural tube cells in 3D from TopoVelo. Each dot represent a cell with color indicating the brain or neural tube region. **(c)** Illustration of 3D geometric transformation of the brain and neural tube. **(d)** 3D scatter plot of the brain and neural tube with the principal curve passing through. **(e)** Heatmap plots showing marker gene expression in different brain subregions. **(f)** 3D scatter plot of the original (left) and transformed (right) brain and neural tube. Cells are colored by different subregions, namely forebrain (green), midbrain (red) and hindbrain (blue). **(g)** Top view of the transformed brain and neural tube with putative closure points. Cells are colored by a heatmap indicating the magnitude of cell velocity along the anterior (red) - posterior (blue) axis. 66

3.7 **TopoVelo Analysis of Mouse Neural Tube Closure.** **(a)** Schematic of registering whole mouse embryo Slide-seq data. Each dot represents a slide-seq spot and is colored by its cell type. Red represents the brain and neural tube. Grey represents all other cell types. **(b)** Quiver plot of the cell velocity of the brain and neural tube cells in 3D from TopoVelo. Each dot represent a cell with color indicating the brain or neural tube region. **(c)** Illustration of 3D geometric transformation of the brain and neural tube. **(d)** 3D scatter plot of the brain and neural tube with the principal curve passing through. **(e)** Heatmap plots showing marker gene expression in different brain subregions. **(f)** 3D scatter plot of the original (left) and transformed (right) brain and neural tube. Cells are colored by different subregions, namely forebrain (green), midbrain (red) and hindbrain (blue). **(g)** Line plot showing the net flow into or out of a position on the A-P axis. **(h)** Top view of the transformed brain and neural tube with putative closure points. Cells are colored by a heatmap indicating the magnitude of cell velocity along the anterior (red) - posterior (blue) axis. 69

3.8 **TopoVelo Analysis of Human Embryoid Bodies.** **(a)** Schematic representation of the embryoid bodies (EBs) differentiation and spatial transcriptomics. **(b)** Cell velocity streamline plot from TopoVelo. Cells are colored by annotated cell types. **(c)** Heatmap showing the cell influence score. Cells are colored by annotated cell types. **(d)** Heatmap showing the number of significant cell-cell WNT signaling interactions inferred by CytoSignal. **(e)** Zoomed-in view of the cell velocity streamline plot mapped to the H&E of a single embryoid. 72

4.1 **Illustration of f-DAG.** **(a)** Graph representation of an f-DAG and its corresponding half-square node graph. Circular shapes denote nodes and diamond shapes denote factors. **(b)** Algebraic representation of an f-DAG and its corresponding half-square node graph. The cross on left hand side denotes matrix Boolean product. x_1 - x_6 denote nodes and f_1 - f_3 denote factors. $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{A}$ denote a node-to-factor, factor-to-node and node-to-node adjacency matrix. . . 81

4.2 **Construction of an f-DAG.** Dashed lines and blocks indicate optional steps for extended f-DAGs with unknown intervention targets. **(a)** Generative process of f-DAGs. Generation proceeds from left to right. **Left**: Random sampling of node partition (top), undirected node-to-factor graph (middle) and optionally intervention-to-node bipartite graph (bottom). Note the sampling steps are repeated independently for each node. **Middle**: Sampling Results. Random node insertion results in a partial ordering of node and factors. Connection sampling specifies undirected edge connection. **Right**: Formation of an f-DAG by combining the information from partial ordering, undirected edge connection and optional intervention-to-node connection. 84

4.3 **Overview of the SPN-FG Model. (a)** Building blocks of a sum-product network (SPN). **Top**: An input node encodes a PMF or PDF given an input value x. **Middle**: A product node generates a product of input distributions as the output. **Bottom**: A sum node generates a mixture of input distributions as the output. **(b)** An Example of SPN Architecture. Assume inputs are random bits $x_1 \dots x_4$. Input nodes generate both 0's and 1's for each bit. Next, a product layer merges x_1, x_2 and x_3, x_4 by generating all 2-bit sequences for $\{x_1, x_2\}$ and $\{x_3, x_4\}$ respectively. Then, a sum layer downsamples inputs. Finally, a product and a sum layer merge $x_1, \dots, x_4$ together and output a 4-bit sequence. **(c)** Illustration of the SPN-FG Model. **Left**: Parametric models for random quantities specifying an f-DAG. Node-factor partial ordering is determined by a Categorical random variable $\boldsymbol{Y}$. Undirected node-to-factor and directed intervention-to-node connections are determined by random binary matrices $\mathbf{B}$ and $\mathbf{W}$. Each row of $\mathbf{B}$ and $\mathbf{W}$ follows a joint PMF parameterized by an SPN. **Middle**: Sampled partition values $y_1, \dots, y_4$ and binary matrices $\mathbf{B}$ and $\mathbf{W}$. **Right**: Matrix form of f-DAG-to-DAG conversion. Node-to-factor and factor-to-node matrices are determined by the partition $\boldsymbol{Y}$ and connection matrix $\mathbf{B}$. 85

4.4 **Overview of ABCDEFG. Top Left:** Bayesian framework. A prior p with a support of all DAGs and a variational distribution with a support of f-DAGs. The red star represents the ground-truth DAG and light blue dots with increasing transparency show an optimization process w.r.t. the variational distribution. **Top Right:** Real vs. generated data distribution. **Bottom:** ABCDEFG model architecture. Binary matrices $\mathbf{U}, \mathbf{V}, \mathbf{W}$ are sampled from a parametric f-DAG model such as SPN-FG. Next, observations are masked by sampled causal relations (under Hadamard product, $\odot$) and fed to a VAE model fitting data distribution. Arrows show direction of data flow and back propagation. 88

4.5 **Benchmarking Results on Interventional Data.** The scatter plots show precision vs. recall on datasets with **(a)** linear targeted, **(b)** non-linear targeted, **(c)** linear untargeted and **(d)** non-linear untargeted SEMs. Markers represent simulation types and colors represent methods. The box plots show the SHD on datasets with **(e)** linear targeted, **(f)** non-linear targeted, **(g)** linear untargeted and **(h)** non-linear untargeted SEMs. Methods starting with "Intv." refer to results on intervention target identification. 94

4.6 **Evaluation of the DAG Constraint. (a)** Bar plot showing the proportion of acyclic vs. cyclic graphs inferred by each method. **(b)** Box plot showing upper and lower bound of minimum number of edge deletion operations to convert cyclic graphs into acyclic ones. 95
4.7 **Extended Factor Graph Inferred from Growth Factor Screen.** Inferred causal edges among interventions with unknown targets (growth factors; pentagons), factors (circles), and genes (text) are shown. Factor colors indicate gene ontology terms enriched in the upstream (blue) and downstream (orange) genes. Edges from interventions to factors are shown in gray arrows, and edges between genes and factors are shown in black arrows. 96

A.1 **Comparison of Other Performance Metrics.** We compared VeloVAE with all other methods in terms of Generalized CBDir on both gene expression space **(a)** and 2D low-dimensional embedding **(b)**, velocity accuracy on both gene expression space **(c)** and 2D low-dimensional embedding **(d)** and time accuracy. All metrics are computed based on prior knowledge of cell development in each tissue. Note that we only compared the genes which scVelo fitted, i.e. velocity genes, as there are genes scVelo didn't fit or fitted with zero likelihood. 110
A.2 **Histogram of scVelo and VeloVI Time Correlation.** We computed the the Spearman correlation between scVelo **(a)** and VeloVI **(b)** locally fitted (gene-specific) time and their global latent time respective for twelve benchmarking datasets. This gives a correlation value for each fitted gene. The figures show box plots of these correlations. 110
A.3 **Cell State-time Plot.** 2D UMAP embeddings of the latent cell state **c** plotted versus inferred latent time. Each plot has the 2D UMAP embedding as the x-y plane and cell time as the z axis. The plots can be interpreted as time evolution of the cell state space. The seven panels correspond to pancreas **(a)**, erythroid **(b)**, subsampled mouse brain **(c)**, iPSC **(d)**, human bone marrow **(e)**, full mouse brain **(f)** and mouse embryo **(g)**. 111
A.4 **3D Velocity Plot.** 3D plots computed as described in Fig. A.3, with velocity vectors added. The x-y plane is either UMAP (a-e) or t-SNE (f-g) coordinates. The 3D embedding is computed by averaging displacement vectors towards k neareast neighbors in the immediate future, i.e. arrows point upwards along the z axis. The seven panels correspond to pancreas **(a)**, erythroid **(b)**, subsampled mouse brain**(c)**, iPSC **(d)**, human bone marrow **(e)**, full mouse brain **(f)** and mouse embryo **(g)**. 112
A.5 **Inferred Cell State Uncertainty.** Each panel shows a UMAP or t-SNE plot colored by the multi-variate coefficient of variation of the cell state. Cell state is a continuous identification of cell type, so conceptually it should have high uncertainty in progenitor cell types, as cell fate is undetermined. Our results verified this intuition as well as the capability of learning meaningful representations using VeloVAE. The 7 panels correspond to pancreas **(a)**, erythroid **(b)**, subsampled mouse brain **(c)**, iPSC **(d)**, human bone marrow **(e)**, full mouse brain **(f)** and mouse embryo **(g)**. 113

A.6 **Time Variance in Test Datasets.** Each panel shows a UMAP or t-SNE plot colored by the coefficient of variation of the inferred cell time. The 7 panels correspond to pancreas **(a)**, erythroid **(b)**, subsampled mouse brain**(c)**, iPSC **(d)**, human bone marrow **(e)**, full mouse brain **(f)** and mouse embryo **(g)**. . . . 113
A.7 **Cell State-time Plot.** We computed 2D UMAP embeddings of the latent cell state **c** and plot it versus inferred latent time. Following the setting of the original work, we used cosine distance and 15 neighbors. The 10 panels correspond to endothelial **(a)**, epithelial **(b)**, haematopoiesis **(c)**, hepatic **(d)**, mesenchymal **(e)**, melanocyte **(f)**, PNS glia**(g)**, PNS neuron **(h)** and neural tube and notochord **(i)** trajectories in the mouse embryo dataset. 114
A.8 **3D Velocity Plot.** 3D velocity plot computed as described in Fig. S4. The 10 panels correspond to endothelial **(a)**, epithelial **(b)**, haematopoiesis **(c)**, hepatic **(d)**, mesenchymal **(e)**, melanocyte **(f)**, PNS glia **(g)**, PNS neuron **(h)** and neural tube and notochord **(i)** trajectories in the mouse embryo dataset. 115
A.9 **Latent Time of Major Trajectories from the Mouse Embryo.** We compare the capture time, monocle 3 pseudotime and VeloVAE latent time by showing the UMAP plots colored by latent time of ten major trajectories in the embryo dataset: endothelial**(a)**, epithelial **(b)**, haematopoiesis **(c)**, hepatic **(d)**, mesenchymal **(e)**, melanocyte **(f)**, PNS glia **(g)**, PNS neuron **(h)** and neural tube and notochord **(i)** trajectories. 116
A.10 **Velocity Stream Plots for Major Trajectories from the Mouse Embryo.** We plotted the velocity embedding on UMAP coordinates of ten major trajectories in the embryo dataset: endothelial **(a)**, epithelial **(b)**, haematopoiesis **(c)**, hepatic **(d)**, mesenchymal **(e)**, melanocyte **(f)**, PNS glia **(g)**, PNS neuron **(h)** and neural tube and notochord **(i)** trajectories. 117
A.11 **Cell State Uncertainty in Major Mouse Embryo Trajectories.** Each panel shows a UMAP plot colored by the multi-variate coefficient of variation of the cell state. The 10 panels correspond to endothelial **(a)**, epithelial **(b)**, haematopoiesis **(c)**, hepatic **(d)**, mesenchymal **(e)**, melanocyte **(f)**, PNS glia **(g)**, PNS neuron **(h)** and neural tube and notochord **(i)** trajectories in the mouse embryo dataset. 118
A.12 **Cell Time Variance in Major Mouse Embryo Trajectories.** Each panel shows a UMAP plot colored by the coefficient of variation of the cell time. The 10 panels correspond to endothelial **(a)**, epithelial **(b)**, haematopoiesis **(c)**, hepatic **(d)**, mesenchymal **(e)**, melanocyte **(f)**, PNS glia **(g)**, PNS neuron **(h)** and neural tube and notochord **(i)** trajectories in the mouse embryo dataset. 119
A.13 **Rate Parameter Histograms.** Box and histogram plots showing the overall range of transcription, splicing and degradation rates estimated by different methods. For VeloVAE, transcription rates are reported as $\rho = 1$. For splicing and degradation, we report βu and γs to match the unit of α (mRNA / minute). Here, u and s are chosen to be half of the 95th-percentile count number. The first four panelsT correspond to pancreas **(a)**, erythroid **(a)**, iPSC **(b)**, subsampled mouse brain **(c)**, full mouse brain **(d)**, and mouse embryo **(i)** datasets. 120

B.1 **Spatial Neighborhood Size Under Different Ball Graph Parameters.** The line plot shows the average number of spatial neighbors vs. the radius of a ball graph constructed on the spatial coordinates. Each line represents a dataset. 126
B.2 **RCTD Clustering and Cell Type Compositions of Spatial Transcriptomic Data of Human Embryoid Bodies (EBs). (a)** Cell Type Assignments of Each Location Inferred by RCTD. **(b)** EB Identity Assignments of Each Location. **(c)** Cell Type Composition within Each Whole EB. 127
B.3 **Evaluation of Rate Parameter Estimation on Simulated Datasets.** Each color bar represents a method. For the cell-specific gene transcription rate (alpha), each bar represents the mean metric and the error bar represents the standard deviation. 128
B.4 **Comparison of Cell Velocity.** Each velocity stream plot visualizes the cell velocity inferred from a dataset (row) by a method (column). The stream plots, from top to bottom, are generated from Slide-seq E15 mouse brain cortex, Slide-seq E13.5 mouse gut and mouse lung, and Visium postnatal 3-month mouse thymus. Each column is a separate RNA velocity method shown at the top. "N/A" means a method failed to run on a certain datast. 129
B.5 **Comparison of Inferred Cell Time.** Each heatmap visualizes the cell time inferred from a dataset (row) by a method (column). The stream plots, from top to bottom, are generated from Slide-seq E15 mouse brain cortex, Slide-seq E13.5 mouse gut and mouse lung, and Visium postnatal 3-month mouse thymus. Each column is a separate RNA velocity method shown at the top. "N/A" means a method failed to run on a certain dataset. 130
B.6 **TopoVelo Robustly Models Spatial Dynamics in Simulated Datasets with Low Spatial Resolution.** Simulated datasets with layered growth patterns **(a)** and radial growth patterns **(b)**. The true velocity stream is plotted over spatial coordinates colored by the true time values. Note that these stream plots indicate cell velocity in physical coordinates, not RNA velocity in abstract UMAP coordinates. 131
B.7 **Sensitivity of TopoVelo Performance to Epsilon-Ball Graph Radius. (a)** Bar plots comparing time correlation, cross-boundary direction correctness, spatial velocity consistency and spatial time consistency from TopoVelo with trained with different radii. **(b)** Line plots showing the k-step CBDir (K-CBDir) metric. Each line with distinct marker and color represents one radius choice. x-axis is the step number and y-axis is the K-CBDir value. 132
B.8 **Sensitivity of TopoVelo Performance to KNN Neighborhood Size. (a)** Bar plots comparing time correlation, cross-boundary direction correctness, spatial velocity consistency and spatial time consistency from TopoVelo with trained with different KNN neighborhood size. **(b)** Line plots showing the k-step CBDir (K-CBDir) metric. Each line with distinct marker and color represents one KNN graph choice. x-axis is the step number and y-axis is the K-CBDir value. 133
B.9 **Sensitivity of Inferred Cell Time to Epsilon-ball Graph Radius.** Each panel is a heatmap showing the inferred cell time using an epsilon-ball graph with the given radius shown at the top. For the Visium thymus dataset, a different set of radius values are used due to a lower spatial resolution. 134

B.10 **Sensitivity of Cell Velocity to Epsilon-ball Graph Radius.** Each panel is a velocity stream plot showing the cell velocity on the physical coordinates. The radius is shown at the top. For the Visium thymus dataset, a different set of radius values are used due to a lower spatial resolution. 135
B.11 **Sensitivity of Inferred Cell Time to KNN Neighborhood Size.** Each panel is a heatmap showing the inferred cell time using a spatial KNN graph with the number of neighbors shown at the top. 136
B.12 **Sensitivity of Cell Velocity to KNN Neighborhood Size.** Each panel is a velocity stream plot showing the cell velocity on the physical coordinates. The number of neighbors of the spatial KNN graph is shown at the top. 137
B.13 **Results on Visium Data (Part 1).** Results from the first **(a)**, fourth**(b)** and fifth**(c)** slices from a 10X Visium mouse embryo dataset. Each panel shows the spatial coordinates colored by cell type annotations, cell velocity stream plot and a heatmap plot showing the developmental time inferred by TopoVelo. 138
B.14 **Results on Visium Data (Part 2).** Results from the sixth **(a)**, and nineth**(b)** slices from a 10X Visium mouse embryo dataset. Each panel shows the spatial coordinates colored by cell type annotations, cell velocity stream plot and a heatmap plot showing the developmental time inferred by TopoVelo. 139
B.15 **Quantifying Mouse Neural Tube Closure with Cell Velocity. Top**: 2D quiver plot showing cell velocity on the transformed anterior-posterior(A-P) axis of a E9 mouse neural tube. **Bottom**: 1D divergence of cell velocity on the A-P axis. 140

C.1 **Illustration of Proposition C.20 (a)** Case I: $i \to j' \to k$ in $D_2^{\mathcal{I}}$. **(b)** Case II: $k \to j' \to i$ in $D_2^{\mathcal{I}}$. Contradictory edges are colored in red. 159
C.2 **Illustration of Cases in Proposition C.21 (a)** $k \in chd(j), k \notin C_j(D_1^{\mathcal{I}})$. **(b)** Contradictory Case I: $\exists k_1, k_2 \in C_j(D_1^{\mathcal{I}})$ s.t. $k_1 \to f_1 \to i, k_2 \to f_2 \to i$ in $D_2^{\mathcal{I}}$ **(c)** Contradictory case II: $\exists$ unique h s.t. $h \to f' \to i$ in $D_2^{\mathcal{I}}$. **(d)** Contradictory Case: there exists multiple factors connecting i to $chd(j; D_1^{\mathcal{I}})$. **(e)** Contradictory Case: $chd(j; D_)^{\mathcal{I}} \subset chd(j'; D_2^{\mathcal{I}})$. Red edges indicate contradictory v-structures. . 161
C.3 **Illustration of Cases in Case I of Proposition C.22 (a)** Subcase I-1. **(b)** Subcase I-2. **(c)** Subcase I-3. **(d)** Subcase I-4. Red edges indicate contradictory v-structures. 162
C.4 **Illustration of Cases in Case II of Proposition C.22. (a)** Contradictory case when $P_j(D_1^{\mathcal{I}}) \not\subseteq P_f(D_2^{\mathcal{I}})$. **(b)** Contradictory case when $P_j(D_1^{\mathcal{I}}) \subseteq P_f(D_2^{\mathcal{I}})$ but $P_f(D_2^{\mathcal{I}}) \not\subseteq P_j(D_1^{\mathcal{I}})$. **(c)** Contradictory case when $C_j(D_1^{\mathcal{I}}) \not\subseteq C_f(D_2^{\mathcal{I}})$. **(d)** Contradictory case when $C_j(D_1^{\mathcal{I}}) \subseteq C_f(D_2^{\mathcal{I}})$ but $C_f(D_2^{\mathcal{I}}) \not\subseteq C_j(D_1^{\mathcal{I}})$. Red edges indicate contradictory v-structures. 165
C.5 **Training Time of ABCDEFG.** Each box represents one type of simulation. We groups simulation regarding the ground truth graph type and known vs. unknown intervention targets. We use the suffix "-T" for known intervention targets and "-U" for unknown ones. 169

LIST OF TABLES

TABLE

A.1 **Performance on scRNA-seq Datasets.** 109
A.2 **Default Hyperparameters of VeloVAE.** 121
A.3 **Default Hyperparameters of Branching ODE.** 122
A.4 **Variable Hyperparameters across Datasets.** We utilized the capture time in certain datasets and reversed gene mode assignment to avoid bad initialization. 122

B.1 **Spatial Graph Construction** 123
B.2 **Data Preprocessing** 124
B.3 **Parametric Model for Spatial Dynamics.** The table lists different forms of spatial velocity and spatially-dependent transcription rates used for simulating 1D bidirectional and 2D radial cases. 125

C.1 **Performance on Simulated Datasets with Linear SEM on 16 Nodes.** Best performance is in bold text and second best is underlined. 167
C.2 **Performance on Simulated Datasets with Non-linear SEM on 16 Nodes.** Best performance is in bold text and second best is underlined. 167
C.3 **Default Hyper-Parameter Setting of ABCDEFG in a Simulation Study.** 168
C.4 **Hyper-Parameter Setting of ABCDEFG in a Simulation Study.** 169

LIST OF APPENDICES

A Supplementary Materials for Chapter 2 . . . . . **106**

B Supplementary Materials for Chapter 3 . . . . . **123**

C Supplementary Materials for Chapter 4 . . . . . **141**

1 A Greedy Algorithm for the FAS Problem 171
2 Upper bound of acyclicity distance 172
3 Lower bound of acyclicity distance 173

LIST OF ACRONYMS

bulk RNA-seq bulk RNA sequencing

cDNA complementary DNA

mRNA messenger RNA

scRNA-seq Single-cell RNA sequencing

ABCDEFG Amortized Bayesian Causal Discovery of Extended Factor Graphs

CBDir Cross-Boundary Direction Correctness

CME chemical master equation

DAG directed acyclic graph

DEG differentially expressed gene

ELBO evidence lower bound

FACS fluorescence-activated cell sorting

GAT graph attention

GCN graph convolution

GNN graph neural network

GRN gene regulatory network

KNN k nearest neighbor

MSE mean-squared error

ODE ordinary differential equation

PC probabilistic circuit

PDF probability density function

PMF probability mass function

SEM structural equation model

SPN sum-product network

UMI unique molecular identifier

VAE variational autoencoder

VI variational inference

ABSTRACT

Understanding the principles underlying gene expression is crucial for numerous fields, including cancer research, neuroscience, and medicine. Recent advancements in biotechnology, such as single-cell RNA sequencing and spatial transcriptomics, have made it possible to measure gene expression at single-cell resolution and offered new opportunities to quantitatively study gene expression and its relationships to cell phenotypes. The vast amount of transcriptomic data has created a pressing need for novel computational methods to uncover hidden biological insights. While many existing computational approaches focus on distinguishing cell identities based on gene expression profiles, a gap remains in achieving a comprehensive and mechanistic understanding of how and why these differences arise. This dissertation seeks to bridge this gap by addressing three key challenges unique to this domain and summarizing findings from three distinct yet interconnected studies. The first study presents VeloVAE, a variational Bayesian method for recovering temporal information on RNA transcription, splicing, and degradation from single-cell RNA sequencing data. The second study introduces TopoVelo, a graph learning method for uncovering the spatial dynamics of cell migration and differentiation using spatial transcriptomic data. The third study introduces ABCDEFG, a differentiable causal discovery method designed to reveal causal relationships from single-cell perturbation data. These methods integrate deep learning techniques with domain knowledge to extract hidden information from existing biological data. Through a series of studies, we demonstrate that these approaches achieve state-of-the-art performance and effectively capture biological insights related to the temporal, spatial, and causal mechanisms of gene expression.

CHAPTER 1

Introduction

1.1 Background

The human body consists of trillions of cells[2]. Although all cells within the same organism share the same genetic information encoded in their DNA, their properties and functions vary widely. Uncovering the underlying mechanisms of this diversity helps scientists better understand cell development, with significant implications for various scientific and medical applications. The central dogma of molecular biology[3] states that DNA, which carries genetic information, is first transcribed into RNA, specifically messenger RNA (mRNA). These RNA molecules are then translated into proteins, which perform essential functions in the body. Consequently, the frequency at which different genes are transcribed directly influences cellular properties. In other words, understanding the mechanisms that regulate mRNA abundance is crucial for addressing many questions in molecular biology.

Recent advancements in sequencing technology have enabled the generation of large-scale transcriptomic data — data that capture gene expression levels through mRNA counts. Access to these data allows researchers to explore the underlying mechanisms driving observed biological phenomena. Key areas of interest include understanding how mRNAs are produced over time and across spatial contexts, as well as how mRNA molecules from different genes interact causally. However, such information cannot be directly measured through experiments. This limitation has spurred the research community to develop computational methods to address these biological questions.

Discovering hidden information from observational data is typically approached as a latent variable problem. Due to the high dimensionality, extracting meaningful insights from transcriptomic data remains a significant challenge. Recently, deep generative models have demonstrated remarkable capabilities in learning latent representations and generating data with rich information, including images[4, 5, 6, 7, 8], text[9, 10, 11, 12, 13, 14], and molecules[15, 16, 17, 18]. Unlike models that learn a static latent space containing only

information about cells and genes, our goal is to uncover the mechanisms that connect them.

1.2 Single-Cell RNA Sequencing

Since the discovery of the central dogma, researchers have consistently sought to understand the relationship between gene expression and cell heterogeneity. As mentioned in the previous section, the abundance of mRNA molecules reflects gene expression levels. Consequently, RNA sequencing technology has become essential for generating data suitable for computational research. RNA is a single-stranded molecule composed of four types of nitrogenous bases: adenine (A), cytosine (C), guanine (G), and uracil (U) — with the key difference from DNA being the presence of uracil (U) instead of thymine (T). Genetic information is encoded in the sequence of these nitrogenous bases. With the advancement of next-generation sequencing, we can capture mRNA molecules by synthesizing their complementary DNA (cDNA) sequences and reading the base sequences of the captured molecules. Each output sequence is referred to as a read. By mapping these reads to a reference genome, such as the human genome assembled during the Human Genome Project[19], we can identify the corresponding genes. Finally, the number of reads mapped to each gene quantifies its expression level.

In the early days, bulk RNA sequencing (bulk RNA-seq) was frequently used in quantitative study, especially in cancer biology[20, 21]. As its name suggests, each sequencing experiment takes a piece of tissue and aggregate the gene expression levels over all cells within the sample. Despite being cost effective, bulk RNA-seq has limited cell-level resolution and hence, does not help detect heterogeneity within a cell population[21]. Nevertheless, it has been widely used to identify differentially expressed genes from different tissue samples, e.g. healthy vs. tumor tissues.

Single-cell RNA sequencing (scRNA-seq) is an advanced technique that allows researchers to examine the gene expression profiles of individual cells, offering a more detailed view of cellular function and diversity compared to traditional methods. Unlike bulk RNA sequencing, which averages gene expression signals across a large population of cells, scRNA-seq captures the heterogeneity and unique characteristics of each cell within a sample. This ability to analyze individual cells is crucial for understanding complex tissues composed of diverse cell types and states — such as tumors or developing organoids — where subtle differences can have significant biological implications. By providing insights into the specific functions and interactions of individual cells, scRNA-seq facilitates the precise characterization of cellular responses, the identification of rare cell types, and the exploration of specific cellular pathways.

A scRNA-seq experiment consists of several major steps: (1) sample preparation, (2) cell capture and isolation, (3) cell lysis, (4) reverse transcription, (5) amplification and (6) sequencing[22]. Sample preparation usually refers to taking a section or slice of a tissue as the experiment subject. Cell capture and isolation are the process of retrieving individual cells from a tissue section. Major cell capturing methods include fluorescence-activated cell sorting (FACS), microfluidics, layer dissection, etc. Next, cells are dissociated in the third step called lysis so that RNA molecules are exposed to be captured. After that, as mentioned previously, RNA molecules are fragmented, captured by beads, reversely transcribed to cDNA, amplified in quantity and finally sent to a sequencing machine. Each bead contains a unique barcode so that we can identify all RNA molecules from a specific cell. In addition, a unique molecular identifier (UMI) is often used to distinguish different genes and correct errors. The final outcome of an experiment is a cell-by-gene matrix whose entries are the number of mRNA molecules associated with each gene. Since its first emergence in 2009[23], scRNA-seq technology keeps evolving and a number of sequencing technology has emerged[24, 25, 26, 27, 28, 29].

1.3 Spatial Transcriptomics

Another milestone in single-cell genomics was reached with the advent of spatial transcriptomics[30]. The primary goal of spatial transcriptomics is to measure gene expression levels while preserving the spatial organization of cells within a tissue. This approach allows researchers to investigate gene expression variation and cell heterogeneity across a defined microenvironment, capturing the intricate interactions between spatially adjacent cells as they coordinate during processes such as development, differentiation, and disease progression. In some cases, spatial transcriptomics can be seen as an extension of single-cell RNA sequencing (scRNA-seq) that is enriched with spatial information. However, there are several critical differences between these technologies.

While scRNA-seq isolates individual cells to capture their transcriptomes, spatial transcriptomics preserves the spatial context of gene expression, enabling the mapping of gene activity to its original anatomical location within the tissue. This spatial resolution is essential for understanding tissue architecture, the organization of distinct cellular niches, and the dynamic interactions between neighboring cells. One of the limitations of scRNA-seq is that it often requires enzymatic or mechanical dissociation of tissues — a process that can disrupt cell-cell interactions and eliminate spatial information crucial for contextual analysis. In contrast, spatial transcriptomics maintains the native tissue environment, allowing for in situ profiling of gene expression patterns. This advantage enhances the study of cell-

cell communication, tissue compartmentalization, and region-specific molecular processes [31, 32]. However, spatial transcriptomics presents its own set of challenges. The spatial resolution — the smallest unit at which gene expression can be measured — depends on the technology used and can range from subcellular regions to larger spots containing multiple cells[33]. Additionally, the inherent complexity of tissue organization, coupled with the need to align molecular data with histological features, introduces computational challenges in data integration and interpretation. Despite these limitations, spatial transcriptomics has emerged as a transformative tool for dissecting the molecular landscapes of tissues, offering unprecedented insights into spatially resolved biological processes.

Spatial transcriptomics technologies can be broadly categorized into two main types: image-based and sequencing-based approaches [34, 33]. Image-based spatial transcriptomics relies on the use of fluorescently labeled probes designed to hybridize with target mRNA molecules within a tissue section. Once the probes bind to their complementary mRNA sequences, fluorescence signals are detected using microscopy, generating spatially resolved images where the intensity of the fluorescence corresponds to the abundance of the target mRNA at each location. This method provides high spatial resolution, often down to the subcellular level, but is typically limited to detecting a predefined set of genes, as only those targeted by the probes can be visualized.

In contrast, sequencing-based technology directly measures the number of mRNA molecules. Similar to scRNA-seq, sequencing-based methods start with a tissue slice. The slice is placed on a specially designed slide. Each slide is pre-coated with an array of spatially barcoded nucleotides. Each barcode corresponds to a unique spatial coordinate, allowing for the mapping of transcripts back to their original location within the tissue. After a tissue slice is placed onto a slide, cells are permeabilized and release mRNA. The mRNA molecules have long poly-A tails, a sequence of many base A's (adenine), and the sequencing technology makes use of this property by designing a capturing sequence complementary to the poly-A tail. The capture sequences are part of a longer sequence with spatial barcodes and UMIs. The next step is to amplify mRNA reads so that we detect enough signal. The amplified cDNA molecules are sequenced to produce reads. Finally, the reads are mapped to a reference DNA, and gene expression is quantified as the number of unique reads mapped to the region on DNA corresponding to the gene. In addition, spatial barcodes are decoded to produce the spatial locations of each spot. For sequencing-based methods, the size and density of spatial barcodes determine spatial resolution. Each spot on a slide can correspond to multiple cells, a single cell, or a subregion of a cell. The spacing varies from several micrometers up to 100 micrometers[33]. Low resolution often results in cell mixing — each spot contains a sum of all mRNAs from multiple cells, leading to inaccuracy in quantitative

studies. After sequencing, read mapping, and spatial barcode decoding, the final outcome is a cell-by-gene mRNA count matrix and a list of spatial coordinates, usually 2D coordinates, of all spots.

1.4 Single-Cell RNA Sequencing with Perturbation

In cell biology, a key problem is understanding how genes coordinate and regulate each other during normal and diseased states. This question has further implications for applications such as drug development, where it is essential to predict cellular responses to various stimuli and discover effective drugs for treatment. While scRNA-seq already provides rich information about cellular heterogeneity, it is insufficient for identifying gene regulation due to the extreme complexity of the gene regulatory network. However, with new CRISPRi/a technology, we are able to interfere or activate different genes in the genome and measure their expression levels. Thus, a new type of single-cell RNA sequencing technology based on the CRISPR-dCas9 system has emerged.

CRISPR, short for "Clustered Regularly Interspaced Short Palindromic Repeats", originally referred to a set of DNA sequences found in microbial immune systems. Later, it was used in the well-known CRISPR-Cas9 technology to perform gene editing[35]. CRISPR contains pieces of DNA sequences that are transcribed into RNA. Then, Cas9, a type of enzyme, is guided by the RNA sequence to specific locations on the genome and cuts the corresponding gene sequence. Later, the use of nuclease-deficient Cas9 (dCas9)[36], combined with CRISPR technology, enabled genomic targeting without cleaving the DNA sequence. This new technology opened the door for intervening gene expression without introducing genetic mutation. Currently, CRISPR interference (CRISPRi)[36, 37, 38] and CRISPR activation (CRISPRa)[39, 40] are two most popular techniques. The former suppresses gene expression by blocking RNA polymerase from gene transcription, while the latter activates gene expression by recruiting transcription activators[41]. Several studies have combined CRISPR-dCas technology and scRNA-seq to enable single-cell measurement of gene expression under different genetic perturbations[42, 43, 44, 45, 46, 47, 48]. Other teams have investigated chemical perturbations, which do not directly knock out genes but can impact multiple genes simultaneously[49, 50, 51]. We refer to the data obtained from single-cell RNA sequencing with genetic or chemical perturbations as single-cell perturbation data.

1.5 Single-Cell RNA Data Generation

Computational tools are ubiquitous along the sequencing pipeline, as information is processed from one form to another. Although data generation is not the main focus of this dissertation, we provide a brief introduction to the entire process and describe relevant computational steps, as data generation is fundamental to any downstream method.

The first step in data acquisition is to read out the sequence of nucleotides in each RNA molecule. Most single-cell RNA sequencing protocols rely on next-generation sequencing technology. Given an RNA molecule, next-generation sequencing sequentially builds its complementary DNA sequence base by base using fluorescently labeled nucleotides. Each nucleotide corresponds to a different color. Each time a base is added, the entire flow cell is illuminated by different fluorescence across the slide, and a picture is taken. With (mostly commercial) computer algorithms, different colors are recognized and converted to one of the four bases (A, C, G, or T). Later technologies, represented by nanopore sequencing, can produce longer sequences ranging from several thousand to over one million bases. Rather than using fluorescence, nanopore sequencing measures the ionic current passing through nanopores embedded in a membrane. As RNA or DNA molecules pass through the pore, the four nucleotides cause distinct shifts in the ionic current, resulting in time-variant current levels[52]. While there are no explicit rules for manually determining read sequences from current measurements, deep neural networks have been successfully applied to this computational task[53, 54, 55].

Once RNA reads are generated, we need to produce a cell-by-gene count matrix. The barcode from each read contains cell (or spot, in spatial transcriptomics) information. To obtain the gene information, we need additional algorithms. The process can be considered a string-searching problem. Given a reference genome, often with billions of bases, we need an algorithm that tells us which part of the genome each read comes from. This computational problem is called read alignment. A number of methods and tools have been developed based on a common framework: (1) build an index of the reference genome, (2) search the query read from the index, (3) compare against the reference and specify mismatches, insertions, and deletions, and (4) look up the corresponding gene in a gene annotation file. This process is repeated for every read. This traditional map-and-align strategy can be time-consuming. In fact, read alignment provides more information than just the gene. Extra computational cost is incurred when determining base pair mismatches, insertions, and deletions in the RNA sequence. This information is important when studying mutations but is irrelevant when we only need gene count numbers. Thus, another type of computational method has been designed to accelerate this process. For example, kb-python[56] is a open-source software that

does not rely on read alignment algorithms. Instead, it uses a pseudo-alignment algorithm. The pseudo-alignment algorithm works by first sequentially segmenting a reference genome into k-mers — length-k substrings of the entire DNA sequence — then building a directed graph, called a De Bruijn graph, and finally aligning all reads to this graph.

After read alignment or pseudo-alignment, additional computational steps take care of quality control. The steps might vary depending on the sequencing platform and technology. In general, quality control may involve the following steps:

1. Filtering out reads with low mapping quality.
2. Filtering out reads from the intronic regions, i.e. parts of the reference DNA that do not correspond to any protein-encoding gene.
3. Barcode and UMI error correction.
4. Total count anomaly detection. This step filters out cells with extremely high or low total RNA counts. High counts can be caused by multiple cells being captured together into a single sample with the same barcode, while low counts often mean failure in capturing a cell.

The final outcome of the data generation pipeline is a cell-by-gene count matrix.

1.6 Computational Methods for Single-Cell Transcriptomics

As scRNA-seq technology brings cell biology into a new era of large-scale, high-dimensional data, data-driven methods have emerged as a new paradigm for biological discovery. Consequently, biological questions have also evolved from being mostly descriptive to quantitative, creating an increasing demand for computational methods. Statistical methods have been widely used in bioinformatic analysis to test hypotheses. Meanwhile, machine learning, especially deep learning, has emerged as a new paradigm for extracting biological insights from single-cell transcriptomic data. Previously proven successful in other application domains such as image recognition and natural language processing, machine learning exemplifies a way to automatically find patterns in data. Although deep learning methods are often criticized as black-box models, they can be valuable for subjects where humans have yet to develop a solid theoretical framework. Cell biology is a good example because humans have only decoded a very small proportion of the cell system. In fact, with a combination of biologically interpretable models and machine learning models, computer scientists and

engineers have shown the possibility of bringing the best of both worlds together to help biologists demystify the cell. Here, we introduce some of the most important computational methods in single-cell transcriptomics.

1.6.1 Data-Driven Approach to Studying Cell Identity.

Cells are often classified into different types based on their identities. Single-cell transcriptomic data are usually contained in a cell-by-gene RNA count matrix. Such a matrix quantifies the level of gene expression in different cells. Thus, computational methods have been developed to study differences in gene expression across different cell identities. This is often referred to as differential gene expression analysis. A number of statistical methods have been developed for this task[57, 58, 59, 60]. These methods usually include additional preprocessing steps to remove low-quality data and noise. Preprocessing typically involves (1) cell and gene filtering based on total counts and expression levels, (2) data normalization, and (3) log transformation. The first step removes low-quality cells or genes from the data. The second step corrects for non-biological artifacts — differences in gene counts due to sequencing depth. The third step acts as a distribution transformation. Because the distribution of discrete counts is often skewed, performing a log transformation helps to normalize the distribution. Popular preprocessing tools include Scanpy[61] and Seurat[62, 63, 64].

The above analysis depends on prior knowledge of cell types. Such information is usually unavailable in an experiment. Cell type annotation typically relies on prior knowledge about marker genes, i.e., genes that are highly expressed in a certain cell population, and is performed manually by experienced scientists. This creates a chicken-and-egg problem for differential gene expression analysis and cell type annotation. Nevertheless, we do have a starting point — clustering. Clustering is an unsupervised learning problem, meaning it is purely data-driven and does not require prior knowledge. The goal is to partition cells into clusters that ideally correspond to biologically meaningful cell types or states. A variety of clustering methods have been developed, each with its own strengths and limitations. Graph-based community detection algorithms, such as the Louvain[65] and Leiden[66] algorithms, are widely used due to their ability to detect clusters of varying sizes and shapes by constructing a nearest-neighbor graph from data. Due to the curse of dimensionality, distinguishing between cells based solely on gene expression is challenging. Therefore, these clustering methods often use low-dimensional representations as inputs, such as PCA.

Another orthogonal direction for extracting cell type information is to learn better cell representations. Previous works applied various dimensionality reduction techniques to learn informative cell representations. Matrix factorization methods learn a low-rank approxima-

tion of a cell-by-gene RNA count matrix, decomposing it into a loading matrix and a factor matrix [67, 68, 69, 70, 71]. Nonlinear dimensionality reduction methods for scRNA-seq data include non-parametric[72] and parametric[73] methods. Later, deep generative models started to be widely used in extracting better cell representation for downstream applications including clustering[73, 74, 75, 76]. Additionally, other nonlinear dimensionality reduction algorithms, such as t-SNE[77] and UMAP[78, 79], are frequently used for visualization.

1.6.2 Understanding Temporal Dynamics During Cell Development.

While learning a low-dimensional cell representation is the most straightforward way to directly distinguish cell identities, it does not imply an understanding of the true mechanisms underlying the differences between cells. This motivates a second computational question: learning a cell trajectory. Unlike the static information of cell identity, a cell development trajectory represents dynamic information and is, therefore, more challenging to infer.

With scRNA-seq data alone, early computational methods quantify cell differentiation by either ranking cells according to their developmental stage[80, 81, 82, 83, 84, 85, 86] or directly placing cells on trajectories[87, 88]. In 2018, La Manno et al. developed the concept of RNA velocity by distinguishing unspliced (immature) mRNA from spliced (mature) mRNA and using a linear ODE to model the splicing process[89]. RNA velocity inference can be viewed as an example of learning a dynamical system from measurement data. Such problems have been explored in other literature with different applications[90, 91, 92, 93, 94, 95, 96, 97]. However, the absence of longitudinal data prevents the direct adaptation of general methods to this specific problem. To overcome this challenge, several methods have proposed various inference techniques. ScVelo[98] and UniTVelo[99] used an EM algorithm to alternately infer latent cell time and ODE parameters. DeepVelo[100] and CellDancer[101] applied neural networks to directly learn the rate parameters. VeloVAE[102, 103], VeloVI[104], and PyroVelocity[105] employed variational Bayes for RNA velocity inference. In contrast to explicitly modeling such biological processes, neural ODE methods[106] learn trajectories on a low-dimensional manifold that generates gene expression data.

1.6.3 Understanding Spatial Processes of Cell Differentiation.

Spatial transcriptomics can be categorized into sequencing-based [107, 108, 109, 110] and imaging-based technologies using in situ hybridization[111, 112, 113]. Compared to scRNA-seq, spatial transcriptomics provides the relative positions of cells within a tissue. This additional layer of information introduces new computational challenges and meth-

ods. Statistical methods have been developed to analyze spatial patterns of gene expression and identify spatially variable genes[114, 115, 116]. Another line of research focuses on understanding cell-cell communication through ligand-receptor signaling[117, 118, 119]. Spateo[120] has been used to infer velocity from spatial transcriptomic data using the Dynamo model[121], but it does not incorporate spatial positions during parameter estimation. cell2fate[122] infers velocity modules from dissociated scRNA data and projects them onto spatial coordinates. Neither spateo nor cell2fate uses spatial information during inference, and thus neither can model the relationship between space and time or among neighboring cells. Similarly, STT[123] estimates rate parameters using scRNA data, then projects the resulting velocity vectors onto spatial coordinates. GASTON[124] learns a one-dimensional isosurface from spatial gene expression but does not model spliced and unspliced counts and cannot assign directionality to the isofurface. In summary, while these previous approaches each incorporate spatial information in some way, they do not directly model how a cell's microenvironment affects its gene expression and differentiation. In contrast to previous computational methods, our recent work, TopoVelo[125], investigates the spatiotemporal process of mRNA splicing and its connection to cell migration using graph neural networks.

Graph neural networks (GNN) have been widely applied to node- and graph-level classification, link prediction, regression tasks, and more. Numerous deep learning architectures have been proposed to enhance performance on these tasks[126, 127, 128, 129, 130]. These methods have demonstrated significant impact in various downstream applications, including molecule generation and property prediction[131, 132], social networks[133], and recommendation systems[134]. In spatial transcriptomics, existing graph neural network (GNN)-based methods focus on learning better cell representations for clustering and cell subtype detection [135, 136].

1.6.4 Causal Discovery.

The gene regulatory network is essentially a causal graph. Classical causal discovery methods can generally be categorized into two types: constraint-based and score-based methods. Constraint-based methods date back to the 1990s when Peter Spirtes and Clark Glymour[137] proposed the PC algorithm. Later works improved upon this initial approach by applying the PC algorithm to large-scale systems[138], incorporating cycles[139], and accounting for latent variables[140]. Score-based methods aim to search for a graph that maximizes a score function. However, they suffer from high time complexity because the acyclicity constraint of Bayesian networks is combinatorial, and the size of the graph set grows superexponentially with the number of nodes[141]. In fact, learning a Bayesian network using discrete

searching can be NP-complete in many cases[142]. Meanwhile, theories about the identifiability of causal systems have provided insights into computational cost[142] and experimental design[143, 144]. Other works focus on extending the notion of graph equivalence to interventional graphs[145] and cyclic graphs[146].

Later, Zheng et al. proposed a continuous relaxation to avoid combinatorial search[147]. This work provides an alternative that is both computationally efficient and effective. Subsequent works in this line of research include NO-BEARS[148], DCDI[149], GOLEM[150], DCD-FG[151], NODAGS-Flow[152], and SDCD[153]. Recently, there has been a growing interest in deep-learning-based causal discovery methods, including neural causal discovery[154] and attention-based causal discovery[155].

A parallel line of work on Bayesian methods begins from a probabilistic setting. Cundy et al.[156] applied variational inference (VI) to linear Gaussian SEMs. Hägele et al.[157] established a Bayesian framework for causal discovery with interventional data. Annadani et al.[158] adopted the NoCurl DAG model[159] and derived a VI method for the parameters. None of these methods have been applied to large-scale, high-dimensional datasets or demonstrated theoretical identifiability guarantees.

Meanwhile, in the field of biotechnology, a series of works have made large-scale single-cell perturbation data available. Chemical perturbation[49, 50] allows us to examine cellular responses under different chemical treatments, mimicking the effects of drugs. In contrast, genetic perturbation methods[43, 42, 44, 45, 46, 47] combine CRISPR-based perturbations and scRNA-seq to enable large-scale experiments with genetic screening. This has led to a wave of machine learning models that predict the effects of perturbations[160, 161, 162, 163, 164]. However, these methods do not provide a direct identification of gene regulatory relationships. Thus, there is still significant room for improving genetic causal discovery methods by leveraging single-cell perturbation data.

1.7 Methodology Overview

This dissertation investigates discovering gene expression mechanisms by dividing the topic into three subproblems concerning temporal, spatial, and causal relationships during cell development. These biological questions are then formulated as computational problems. To address them, a general machine learning framework is proposed to uncover hidden information that explains gene expression mechanisms from single-cell transcriptomic data. This framework comprises a series of variational Bayesian methods and deep learning architectures, along with auxiliary computational techniques for bioinformatic analysis.

The choice of variational Bayes is motivated by both the similarities and differences

among the various biological questions. Inferring temporal and spatial dynamics, as well as the causal relationships underlying gene expression, are all latent variable problems. Consequently, using a unified learning framework ensures consistency throughout the dissertation. At the same time, the three subproblems involve different types of information, and variational Bayes provides the flexibility needed to accommodate diverse learning tasks.

The proposed methodology leverages the power of deep learning by implementing the variational Bayesian learning framework with a variational autoencoder (VAE), a deep generative model. With real biological applications in mind, this dissertation highlights the importance of incorporating domain knowledge into deep learning. Specifically, the proposed methods integrate mRNA splicing dynamics and spatial tissue structures to enhance the learning of temporal and spatial dynamics. In genetic causal discovery, a causal graph is parameterized using probabilistic circuits, a type of deep mixture model, and the causal relationships are explicitly incorporated into the VAE.

All methods in this dissertation are evaluated and compared with existing approaches using either simulated data or single-cell transcriptomic data from various tissues, primarily from mouse and human samples.

1.8 Chapter Overview

This dissertation contains the following chapters:

1. **Introduction** This chapter provides the necessary background information on the application scenarios and motivates the development of new methodologies to address specific challenges. It then reviews the existing literature in relevant research areas, describes general methodologies, and briefly introduces the subsequent chapters. Finally, it outlines the contributions of this dissertation and its long-term impact on both the research community and the general public.

2. **VeloVAE: Variational Bayesian Estimation of RNA Velocity** This chapter introduces single-cell RNA sequencing technology and RNA velocity, a key concept for understanding the temporal behavior of gene expression. It then discusses the use of variational Bayes and deep learning, informed by biochemistry, to recover the temporal dynamics of RNA splicing. The content is based on our published article, "Variational Mixtures of ODEs for Inferring Cellular Gene Expression Dynamics," presented at the 39th International Conference on Machine Learning in 2022[102], as well as a subsequent preprint, "Bayesian Inference of RNA Velocity from Multi-Lineage Single-Cell Data," currently available on bioRxiv[103]. Additional results are presented due

to the emergence of several recent methods on this topic.

3. **TopoVelo: Topological Velocity Inference from Spatial Transcriptomic Data** This chapter introduces spatial transcriptomic sequencing technologies. It then examines the spatial patterns of cell maturation and migration and their relationship to RNA velocity. The content is based on our published work, "Mapping Cell Fate Transition in Space and Time," presented at the 28th Annual International Conference on Research in Computational Molecular Biology[125], as well as a journal manuscript, "Topological Velocity Inference from Spatial Transcriptomic Data Maps Cell Fate Transition in Space and Time," currently under a second round of review following revision.

4. **ABCDEFG: Amortized Bayesian Causal Discovery of Extended Factor Graphs** This chapter introduces genetic perturbation and its associated single-cell perturbation data. It then discusses the challenges of uncovering gene regulatory relationships from such data, framing these challenges as a causal discovery problem. After reviewing existing theories on differentiable causal discovery, the chapter presents a novel differentiable Bayesian causal discovery method and explores the theory behind the identifiability of Bayesian approaches. Finally, it includes a benchmark study using simulated data, along with results from a real single-cell perturbation dataset. This chapter is primarily based on the paper titled "Amortized Bayesian Causal Discovery of Extended Factor Graphs," which is currently under review.

5. **Summary and Future Work** The dissertation concludes with a summary of the key questions of interest, the proposed methods, and the findings. Additionally, it discusses contributions and prospects for future research directions in relevant areas.

CHAPTER 2

Inferring Temporal Mechanism of Gene Expression

2.1 Motivation

The human body contains many cell types with distinct forms and functions, which arise from progenitor cells in a stepwise developmental process. A key question in molecular biology is what regulates this process of cellular development. In general, the diversity of cell types arises not from cell-to-cell differences in the DNA sequence itself, but in which portions of the DNA sequence (genes) are used (expressed) in each cell. The central dogma of molecular biology states that genes are first transcribed into mRNAs and these mRNAs are then translated into proteins, which carry out biochemical functions. The expression level of a gene in a cell can thus be quantified by the number of mRNA molecules present in the cell. Therefore, understanding cellular development requires modeling how mRNA expression changes over time. Such models are crucial for numerous areas of biology and medicine, such as neuroscience, cancer research, and regenerative stem-cell therapies.

The scRNA-seq technology[23] is the predominant experimental technology for measuring gene expression. It counts the number of mRNA sequences expressed within a set of individual cells, ultimately yielding a matrix of expression levels for $20,000$ genes across $10^4 - 10^6$ cells. However, measurement destroys the cell, so scRNA-seq gives only one single static snapshot of each cell at some moment in time. Consequently, it is currently impossible to measure the gene expression in a cell longitudinally over time.

Two main types of computational approaches have been developed for this problem: pseudotime inference and RNA velocity. Pseudotime inference methods use distance from a manually-specified starting cell to rank cells according to degree of development[80, 81]. Many pseudotime methods also aim to infer a tree or graph structure that represents the underlying structure of the developmental process[82, 83, 84, 85, 86]. In contrast, La Manno

et al.[89] developed the concept of RNA velocity based on the observation that both unspliced and spliced mRNA molecules appear in sequencing outputs. The relative ratio of spliced and unspliced counts indicates whether the gene was being turned on or turned off at the time the cell was sequenced. La Manno et al. introduced an ODE model to describe the gene expression process, used a steady state assumption to estimate parameters, and implemented the method in a package called velocyto[89]. Later work[98] relaxed the steady-state assumption, allowing all cells to be used in parameter estimation and inferring a latent time value for each cell. Bergen et al. implemented their method in a package called scVelo[98]. RNA velocity methods have been widely used by biologists to help understand cellular developmental processes[165, 166, 167]. With the advent of single-cell multiomics, multiple modalities bring a more comprehensive view of cell development. Consequently, the concept of RNA velocity is extended to different modalities, such as protein[168] and chromatin accessibility[169]. Li et al. recently extended the dynamical model of scVelo to incorporate chromatin accessibility data, packaged in a tool called MultiVelo[169].

While they have proven useful for biological discovery in many cases, existing approaches for pseudotime inference and RNA velocity inference have significant limitations. Pseudotime inference requires manually specifying a starting cell, is based purely on pairwise cell similarity, and cannot infer the direction or rate of cell development. RNA velocity addresses some of these limitations and, in principle, can infer the direction, rate, and origins of developmental processes. However, current RNA velocity methods rely on numerous simplifying assumptions and often fail to yield accurate results [170].

To address these limitations, we developed VeloVAE, a Bayesian model for RNA velocity inference. VeloVAE uses variational Bayesian inference to estimate the posterior distribution of latent time, latent cell state, and kinetic rate parameters for each cell. Our approach addresses key limitations of previous methods by inferring a global time and cell state value for each cell, explicitly modeling the emergence of multiple cell types, incorporating prior information such as time point labels, using scalable minibatch optimization, and quantifying parameter uncertainty. We show that VeloVAE significantly outperforms previous approaches in terms of data fit and accuracy of inferred differentiation directions. VeloVAE can also capture qualitative features of expression dynamics neglected by previous methods, including late induction, early repression, transcriptional boosts, and bifurcations. These improvements allow VeloVAE to accurately model gene expression dynamics in complex biological systems, including hematopoiesis, induced pluripotent stem cell reprogramming, neurogenesis, and organogenesis. We find that the latent time automatically inferred using all cells can even outperform pseudotime inferred using manually chosen cell subsets and root cells. We further use the inferred parameters to construct cell type transition graphs and fit

branching differential equation models that describe the effects of cell type bifurcations on kinetic rate parameters.

2.2 Technical Contributions

We make the following technical contributions:

1. We solve the problem of inferring latent time and dynamical systems from static snapshot data.

2. We introduce a novel variant of VAE that incorporates domain-specific ODE models into its generative process.

3. We extend the RNA velocity model to broader applications, including full variational Bayes on rate parameters, discrete counts and branching effects in multi-lineage systems.

4. We are among the first to demonstrate RNA velocity analysis on a mega-scale, single-cell mouse atlas.

2.3 RNA Velocity

La Manno et al.[89] developed the concept of RNA velocity based on the observation that both unspliced and spliced mRNA molecules appear in sequencing outputs. The relative ratio of spliced and unspliced counts indicates whether the gene was being turned on or turned off at the time the cell was sequenced. This biological process can be modeled by a set of linear ODEs

$$\frac{du}{dt} = \alpha I_{\{t<t_{off}\}} - \beta u, \ \frac{ds}{dt} = \beta u - \gamma s, \tag{2.1}$$

where $I_{\{.\}}$ is an indicator function for the condition in brackets. The model parameters α, β and γ correspond to the RNA transcription, splicing and degradation rates, respectively. The model assumes that two discrete phases can occur in the gene expression process: (1) induction, when new unspliced RNA molecules are being transcribed at time $t < t_{off}$ (2) repression, when the transcription process stops and no new unspliced molecules are made

at time $t \geq t_{off}$. An analytical solution to the ODEs exists:

$$u(t) = u_0 \exp(-\beta\tau) + \frac{\widetilde{\alpha}}{\beta}(1 - \exp(-\beta\tau)) \tag{2.2}$$

$$s(t) = s_0 \exp(-\gamma\tau) + \frac{\widetilde{\alpha}}{\gamma}(1 - \exp(-\gamma\tau)) + \frac{\widetilde{\alpha} - \beta u_0}{\gamma - \beta}(\exp(-\gamma\tau) - \exp(-\beta\tau)) \tag{2.3}$$

$$\widetilde{\alpha} := \alpha I_{\{t<t_{off}\}},\ \tau := tI_{\{t<t_{off}\}} + (t - t_{off})I_{\{t\geq t_{off}\}}$$

Previous methods[89, 98], though proven effective on certain scRNA-seq data, fail to capture complex dynamics in RNA splicing. For example, scVelo suffers from several significant limitations: (1) lack of a common time scale during model training (2) over-simplified ODE model with constant transcription rates (3) insufficient modeling capability for cell type bifurcations. This dissertation builds a machine learning framework based on existing biological modeling techniques to address these limitations.

2.4 Methods

We propose a variational Bayesian learning framework to extract hidden information from transcriptomic data. We show that this framework can be readily instantiated as a deep generative model with proper engineering techniques and it is applicable to datasets of different sizes and complexity.

2.4.1 Variational Mixture of ODEs

Model Description We formulate the computational problem of inferring temporal gene expression dynamics in the following way. Each sample (cell), indexed by i, is represented by a vector $X_i(t) \in \mathbb{R}^d$ parametrized by time t. The trajectory $X_i(t)$ is governed by some differential equation plus random noise. However, for each i, only the vector $\mathbf{x_i} := X_i(t_i)$ is observed at some unknown time t_i. Our goal is two-fold: recover the latent time t_i for each sample and predict future states, i.e., $X_i(t)$ for $t > t_i$. To formally describe our method, we introduce a few definitions.

Definition 2.1. *Let u_g and s_g denote the unspliced and spliced mRNA count of the g-th gene. Let $\mathcal{G} = \{1, 2, \ldots, G\}$ be a set of genes measured in an scRNA-seq experiment. The* ***feature vector*** *of a cell is defined as $\mathbf{x} = [u_1, u_2, \ldots, u_G, s_1, s_2 \ldots, s_G]^T$.*

Definition 2.2. *The* ***kinetic equation*** *of gene g is defined as a system of ordinary differential equations relating changes in u and s over time. If there exists a solution $F(t; \boldsymbol{\theta})$ to the*

initial value problem with $u(0) = u_0, s(0) = s_0$*, we call this solution the* ***kinetic function*** *for* g*.*

Definition 2.3 (La Manno el al.[89])**.** *Given a kinetic function* $u(t)$ *and* $s(t)$ *of a gene, the* ***RNA velocity*** *of the gene is defined as* $\frac{ds}{dt}$*.*

Our key insight is that knowledge about the biochemical steps required for gene expression can serve as a regularization or constraint for this otherwise ill-posed problem (**Fig. 2.1a**). To recover the latent time, we view the problem from an inference perspective. If we know $(u(t), s(t))$ values for cells, we can infer something about the latent variables that generated them. For predicting future states, we designed a generative model that predicts the spliced counts given cell time and other rate parameters of RNA splicing dynamics[102]. We assume that cell time t is first randomly sampled from a normal prior $\mathcal{N}(t_0, \sigma_0^2)$. If cell capture times are available (e.g., if cells were isolated separately on days 7 and 14), we can use them as an informative prior; otherwise, we can simply use a standard normal prior. We model the u and s counts for each gene using the kinetic function given by the analytical ODE solution, assuming that the genes are conditionally independent given cell times. Then, given N i.i.d. time samples $t_1, t_2, \ldots, t_N$, gene expression data $\mathbf{x}_i$, $i = 1, 2, \ldots, N$, are generated by $\mathbf{x_i} = F(t_i; \boldsymbol{\theta}) + \mathbf{r_i}, \mathbf{r_i} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma_r})$. Here, $F(t; \boldsymbol{\theta}) = [u_1(t), u_2(t), \ldots, u_G(t), s_1(t), s_2(t), \ldots, s_G(t)]^T$ is a vector containing all kinetic functions evaluated at t and $\mathbf{r}$ is Gaussian random noise. Equivalently, this means that the distribution of $\mathbf{x}$ conditioned on time is $p(\mathbf{x}|t, \boldsymbol{\theta}) \sim \mathcal{N}(F(t; \boldsymbol{\theta}), \boldsymbol{\Sigma_r})$. We further assume that the noise variables $\mathbf{r}$ are mutually independent, i.e. $\boldsymbol{\Sigma}_\mathbf{r}$ is diagonal with nonzero entries $\sigma_{u,1}^2, \sigma_{u,2}^2, \ldots, \sigma_{u,G}^2, \sigma_{s,1}^2, \sigma_{s,2}^2, \ldots, \sigma_{s,G}^2$.

We use a similar ODE model for $F(t; \boldsymbol{\theta})$ as in previous works[89, 98], with a slight modification. Instead of assuming all genes start generating mRNA at $t = 0$, we allow asynchronous generation by adding a gene-specific parameter, t_{on}. This small change, in addition to inferring latent time jointly across genes, should alleviate scVelo's difficulty in fitting genes with an absent or short induction phase. The kinetic function thus has the same form as **Eq. 2.2, 2.3**, except that the definition of τ changes:

$$\tau := (t - t_{on}) I_{\{t_{on} \leq t < t_{off}\}} + (t - t_{off}) I_{\{t \geq t_{off}\}}$$

We also make another key change to the ODE model. Rather than assuming a single fixed transcription rate parameter, α, for each gene across all cells, we allow each cell to have its own transcription rate ρ for each gene. $\rho \in [0, 1]$ is a function of latent cell state $\boldsymbol{c}$, and thus

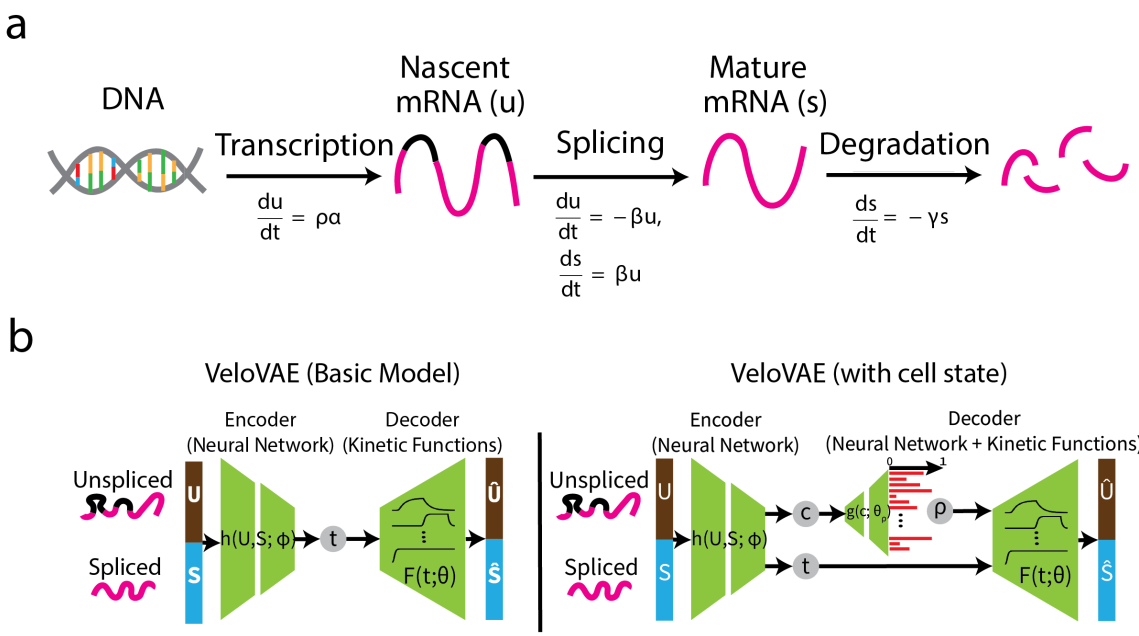

Figure 2.1: **Diagram of VeloVAE Model. (a) Biochemical process** The panel depicts the sequential process of gene transcription, RNA splicing and degradation. **(b) VAE architecture.** In the basic model, an encoder network infers a latent time from all u and s values for each cell. The data are reconstructed from inferred time using ODE solutions, whose analytical form is known. In the full model, an encoder network infers both latent time and latent cell state $\mathbf{c}$. A decoder network generates the transcription rates $\boldsymbol{\rho}$, which are unique for each cell and each gene. The data are then reconstructed from t and $\boldsymbol{\rho}$ using the kinetic function.

may be slightly different in each cell. The new kinetic equation is:

$$\frac{du}{dt} = \rho(\boldsymbol{c})\alpha - \beta u, \quad \frac{ds}{dt} = \beta u - \gamma s \tag{2.4}$$

The new ODE formulation is capable of modeling cell bifurcation during mRNA splicing kinetics.

The generative process for our model, variational mixture of ODE, is as follows:

$$\begin{aligned} t \sim \mathcal{N}(t_0, \sigma_0^2),\ \mathbf{c} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \\ \tilde{\boldsymbol{\alpha}} = \boldsymbol{\rho} \odot \boldsymbol{\alpha},\ \boldsymbol{\rho} = g(\mathbf{c}; \boldsymbol{\theta}_{\boldsymbol{\rho}}) \\ \mathbf{x} \sim \mathcal{N}(F(t; \boldsymbol{\theta}), \boldsymbol{\Sigma}_{\mathbf{r}}) \end{aligned}$$

Here, $g(\cdot)$ is a neural network with parameters $\boldsymbol{\theta}_{\boldsymbol{\rho}}$, $\odot$ is the elementwise product, F is the kinetic function of all genes, and $\boldsymbol{\Sigma}_{\mathbf{r}}$ is a diagonal covariance matrix. This generative process

relies on a function g mapping latent cell states $\mathbf{c}$ to relative transcription rates $\boldsymbol{\rho}$. Intuitively, the cell states can model continuous and bifurcating developmental paths, allowing the entire set of cells to be described as a family of ODEs whose parameters vary smoothly over the cell state manifold. We assume that $\boldsymbol{\rho}$ varies smoothly across the cell state space and that each point in cell state space maps to a unique $\boldsymbol{\rho}$. Although g is deterministic for given $\mathbf{c}$, the inferred cell state for each cell $\mathbf{x}$ is probabilistic. Thus, the distribution of $\boldsymbol{\rho}$ can encompass multiple states near a bifurcation. Our generative model is summarized in figure **Fig. 2.2**.

To fit our probabilistic model on real data, we use autoencoding variational Bayes[171], a powerful statistical inference method in which neural networks approximate the posterior distribution of latent variables that may be nonlinearly related to observed data. One advantage of using variational Bayes is to avoid intractability when we perform exact inference of latent variables from data (**Supplemental Section A.1**). Our method continuously optimizes a lower bound of the total data likelihood, known as the evidence lower bound (ELBO). Based on the generative process, we can derive ELBO as follows:

$$\begin{aligned}
\text{ELBO} &= \sum_{i=1}^{N} \mathbb{E}_{q(t|\mathbf{x_i})} \left[\log p(\mathbf{x_i}|t)\right] - KL(q(t|\mathbf{x_i})||p(t)) \\
\approx &\frac{1}{2} \sum_{i=1}^{N} \left[-2G\log(2\pi) - \log|\boldsymbol{\Sigma_r}| - d(\mathbf{x_i}, F(t_i;\boldsymbol{\theta}); \boldsymbol{\Sigma_r})^2\right] \\
&+ \frac{1}{2}\left[\log\frac{\sigma_p^2}{\sigma_q^2} + \frac{\sigma_q^2}{\sigma_p^2} + \frac{(\mu_p - \mu_q)^2}{\sigma_p^2} - 1\right]
\end{aligned} \tag{2.5}$$

where $F(t_i;\boldsymbol{\theta})$ is the kinetic function and $d(\cdot,\cdot;\boldsymbol{\Sigma})$ denotes the Mahalanobis distance with $\boldsymbol{\Sigma}$ as the covariance matrix. We can then jointly estimate the neural network weights $\boldsymbol{\phi}$, the ODE parameters $\boldsymbol{\theta}$, and the cell times t_i by minimizing the negative ELBO using minibatch stochastic gradient descent.

The variational mixture of ODE model can be interpreted as a variational autoencoder (VAE) that reconstructs the data with an ODE whose parameters vary continuously across a latent space of cell states (**Fig. 2.1b**). Thus, we refer to our approach as a variational mixture of ODEs. Specifically, we name our VAE model VeloVAE.

Parameter Inference and Neural Network Architecture The objective function is the ELBO shown in **Eq. 2.5**, with modified kinetic functions $F(t;\boldsymbol{\theta})$ and an updated KL divergence term incorporating the prior for $\mathbf{c}$. For h, we use the same MLP structure as the simple model with two additional outputs to produce the posterior mean and standard deviation of $\mathbf{c}$. We use an MLP that is the mirror image of h (two layers with 250 and 500

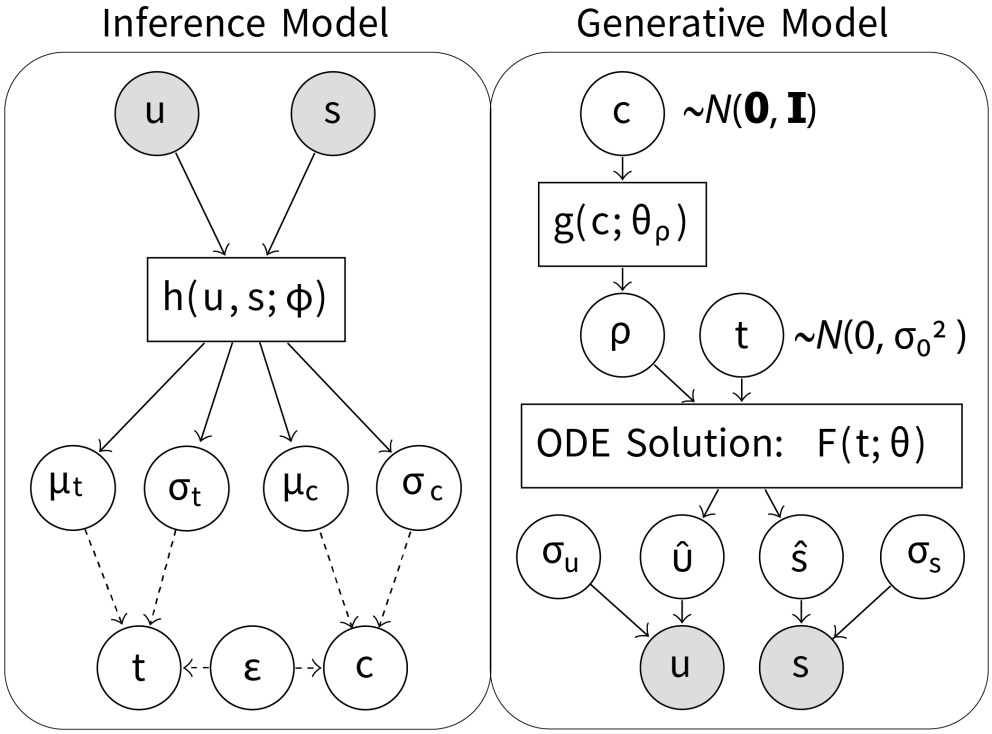

Figure 2.2: **Graphical Model of Variational Mixture of ODE.** Observed variables are colored gray. Dashed arrows indicate sampling.

neurons, respectively) to learn the mapping g from $\mathbf{c}$ to $\boldsymbol{\rho}$.

The encoder is a multilayer perceptron (MLP) containing two hidden layers (500 and 250 neurons, respectively) with batch normalization[172] and dropout[173]. The bottleneck layer outputs the mean and standard deviation parameters of the variational distribution.

We performed minibatch stochastic gradient descent using the ADAM optimizer[174]. For each dataset, we trained on 90% of the data (70% training and 20% validation) until the ELBO converged on the training set (number of epochs varied due to differences in dataset size), then evaluated the reconstruction error and likelihood on the held-out test set. We used 5 latent dimensions for cell state $\mathbf{c}$ in all experiments. For datasets with more than one capture time, we used the capture times to initialize the ODE parameters; otherwise, we used the steady-state approximation for initialization.

Parameter Initialization Because each cell now potentially has different ODE parameters, determining the initial conditions is more complex. Thus, instead of making the initial conditions trainable parameters, we simply train the model with $u_0 = s_0 = 0$ in all of our experiments. This still yields excellent data reconstruction and latent time inference (Table

??). However, the initial conditions are important for accurately predicting the future state of each cell. To improve the accuracy of future state prediction, we first train the VeloVAE to convergence using $u_0 = s_0 = 0$ so that latent times and cell states are accurate, then determine the initial conditions for a cell at time t by simply averaging the (u, s) values observed in an immediately preceding time interval $[t - \delta_1, t - \delta_2]$. We then fine-tune the ODE parameters using these updated initial conditions, keeping latent time and cell state fixed.

2.4.2 Extension of VeloVAE

Since our initial publication of VeloVAE, we have been working on extending it for more use cases and overcoming limitations.

Full Variational Bayes The VAE model can be extended to account for uncertainty in the parameters of the generative model[171], which in our case are the kinetic rate parameters of the ODE system. Let $\boldsymbol{\theta}$ be the set of all ODE parameters. We assume that some prior distribution $p_{\boldsymbol{\lambda}}(\boldsymbol{\theta})$ generates the parameters. Similar to the (intractable) problem of inferring the latent cell time and state variables, we can use a variational approximation of the posterior $q_{\boldsymbol{\phi}}(\boldsymbol{\theta})$. The marginal likelihood of the input features can be bounded by

$$\begin{aligned}\log(\mathbf{X}) &\geq \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{\theta})}\left[\log\left(p_{\boldsymbol{\theta}}(\mathbf{X})\right)\right] - KL\left(q_{\boldsymbol{\phi}}(\boldsymbol{\theta})||p_{\boldsymbol{\lambda}}(\boldsymbol{\theta})\right) \\ &\geq \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{\theta})}\left[ELBO(\mathbf{X};\boldsymbol{\theta})\right] - KL\left(q_{\boldsymbol{\phi}}(\boldsymbol{\theta})||p_{\boldsymbol{\lambda}}(\boldsymbol{\theta})\right)\end{aligned}$$

For the prior $p_{\boldsymbol{\lambda}}(\boldsymbol{\theta})$, we choose a factorized log-normal distribution, i.e. each rate parameter α, β or γ is a random variable drawn from a log-normal distribution. The KL divergence between $q_{\boldsymbol{\phi}}(\boldsymbol{\theta})$ and $p_{\boldsymbol{\lambda}}(\boldsymbol{\theta})$ can be viewed as a regularization of the ODE parameters.

BasisVAE BasisVAE[175] was orignally designed to account for feature-level clustering. It aims at clustering different features (dimensions) of a high-dimensional observation $x = [x_1, x_2, \ldots, x_d]^T \in \mathbb{R}^d$ via a variational auto-encoder. A potential application is to cluster genes with similar patterns.

The model assumes each feature is generated by one of K basis functions, i.e.

$$x_i = \sum_{k=1}^{K} w_k^{(i)} f_k(\boldsymbol{z}), \quad f_k : \mathbb{R}^l \to \mathbb{R}^d, i \in \{1, 2, \ldots, d\}$$

Here, $\boldsymbol{z}$ is a low-dimensional latent variable generating the observation x_i. The clustering in-

formation is stored in $\boldsymbol{w}_i = [w_1^{(i)}, \ldots, w_K^{(i)}]^T \sim \text{Categorial}(\pi_1, \ldots, \pi_K)$ and $\boldsymbol{\pi} := [\pi_1, \ldots, \pi_K] \sim \text{Dirichlet}(\psi_1, \ldots, \psi_K)$. Thus, there are two additional latent variables, $\boldsymbol{w}$ and $\boldsymbol{\pi}$. According to Märtens and Yau[175], we can marginalize $\boldsymbol{\pi}$ using collapsed inference. The collapsed evidence lower bound (ELBO) is given by

$$\begin{aligned}\mathcal{L} = &\sum_{j=1}^{n} \mathbb{E}_{q(\boldsymbol{z_j}|\boldsymbol{x_j})} \mathbb{E}_{q(\boldsymbol{w})} \log p\left(\boldsymbol{x_j}|\boldsymbol{z_j}, \boldsymbol{w}\right) \\ &+ \log\left(\int \exp\left(\mathbb{E}_{q(\boldsymbol{w})}\left[p(\boldsymbol{w}|\boldsymbol{\pi})\right]\right) p(\boldsymbol{\pi})\right) \\ &- \mathbb{E}_{q(\boldsymbol{w})}[\log q(\boldsymbol{w})] \\ &- \sum_{j=1}^{n} KL(q(\boldsymbol{z_j}|\boldsymbol{x_j})||p(\boldsymbol{z_j}))\end{aligned} \tag{2.6}$$

Here, the second term has an analytical form of

$$\log\left[\frac{\Gamma(\sum_k \psi_k)\prod_k \Gamma(n_k + \psi_k)}{\prod_k \Gamma(\psi_k)\Gamma(\sum_k (n_k + \psi_k))}\right]$$

where $n_k := \sum_{i=1}^{d} \psi_k^{(i)}$ and $\Gamma(\cdot)$ is the gamma function.

The VeloVAE model can be extended to a BasisVAE by considering two types of splicing dynamics – genes with an induction and repression phase and genes with only a repression phase. To account for repression-only genes, we add a second set of ODEs:

$$\frac{du}{dt} = -\beta u, \quad \frac{ds}{dt} = \beta u - \gamma s, \quad u(0) = u_0 > 0, \quad s(0) = s_0 > 0 \tag{2.7}$$

Let $(u_g^{ind}(t), s_g^{ind}(t))$ and $(u_g^{rep}(t), s_g^{rep}(t))$ be the analytical solutions to equations 2.4 of the g-th gene. Define $\boldsymbol{F}_{ind} = [u_1^{ind}(t), \ldots, u_G^{ind}(t), s_1^{ind}(t), \ldots, s_G^{ind}(t)]^T$ and $\boldsymbol{F_{rep}} = [u_1^{rep}(t), \ldots, u_G^{rep}(t), s_1^{rep}(t), \ldots, s_G^{rep}(t)]^T$. Given t, $\boldsymbol{F_{ind}}$ and $\boldsymbol{F_{rep}}$ are vectors containing solutions to two sets of kinetic equations with different assumptions. Thus, $\boldsymbol{F_{ind}}$ and $\boldsymbol{F_{rep}}$ are considered two basis functions of our generative model. Each gene has a probability of π_0 to be a repressive gene and we use a random variable $\boldsymbol{w} \sim \text{Categorical}(\boldsymbol{\pi})$ to model random assignment of each basis function to each gene. Finally, we have the following

generative process:

$$
\begin{aligned}
t \sim \mathcal{N}(t_0, \sigma_0^2),\ \boldsymbol{z} &\sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \\
\boldsymbol{\pi} &\sim \text{Dirichlet}(\boldsymbol{\psi}) \\
\boldsymbol{w} &\sim \text{Categorical}(\boldsymbol{\pi}) \\
\tilde{\boldsymbol{\alpha}} = \boldsymbol{\rho} \odot \boldsymbol{\alpha},\ \boldsymbol{\rho} &= g(\mathbf{z}; \boldsymbol{\theta_\rho}) \\
\boldsymbol{x_{ind}} &\sim \mathcal{N}(\boldsymbol{F_{ind}}(t; \boldsymbol{\theta}), \Sigma_\mathbf{r}) \\
\boldsymbol{x_{rep}} &\sim \mathcal{N}(\boldsymbol{F_{rep}}(t; \boldsymbol{\theta}), \Sigma_\mathbf{r}) \\
\boldsymbol{x} &= \boldsymbol{I}_{\{\boldsymbol{w}=1\}} \odot \boldsymbol{x_{ind}} + \boldsymbol{I}_{\{\boldsymbol{w}=2\}} \odot \boldsymbol{x_{rep}}
\end{aligned}
$$

Here, $g(\cdot)$ is a parametric function, e.g. a neural network, with parameters $\boldsymbol{\theta_\rho}$; $\boldsymbol{I}_{\{\cdot\}}$ is a vector indicator function; $\odot$ is the elementwise product; $\boldsymbol{\Sigma_r}$ is a diagonal covariance matrix.

The initialization and training of BasisVAE extension of VeloVAE follows the same procedure as in VeloVAE. The only additional part is $\boldsymbol{w}$. For $\boldsymbol{w}$, we initialize its variational distribution $q(\boldsymbol{w}; \boldsymbol{\pi})$ by using the steady-state[89] and dynamical models[98]. In addition, we performed gene-level clustering to robustly estimate $\boldsymbol{\pi}$.

Discrete VeloVAE Model The biological process of RNA splicing is a chemical reaction network described by the chemical master equation. It considers the change of molecule numbers, which are discrete quantities, as a stochastic process. As Bergen et al. pointed out[98], the dynamical model of RNA velocity is justified by considering the first moment of a stochastic process.

Here, we close the gap between the chemical master equation and its moment equation by introducing the discrete VeloVAE model. We start with chemical equations of all reactions. Consider a gene with transcription, splicing and degradation. For any gene g, the reactions are described by the following chemical equations:

$$
\begin{aligned}
\emptyset &\xrightarrow{\alpha\rho} U_g \\
U_g &\xrightarrow{\beta} S_g \\
S_g &\xrightarrow{\gamma} \emptyset
\end{aligned}
$$

Suppose we have a total of G genes. Then, there are $3 \times G$ reactions and $2 \times G$ molecules in total. Since all reactions are monomolecular, the system has a closed form solution if the initial condition is a product Poisson distribution. This is formally described as the following

theorem from a previous work[176]:

Definition 2.4. *A product Poisson distribution with mean* $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_d]$ *is a discrete distribution with a support of* $\mathbb{N}^d$ *and PMF of the following form:*

$$P(\boldsymbol{x}; \boldsymbol{\lambda}) = \prod_{i=1}^{d} \frac{e^{-\lambda_i} \lambda_i^k}{k!}$$

Theorem 2.5 (Jahnke, 2007[176])**.** *Let* $P(t, \boldsymbol{x})$ *be the pmf of a feature vector (mRNA counts) at time* t*. If* $P(0, \boldsymbol{x})$ *is a product Poisson distribution, then,* $P(t, \mathbf{x})$ *at any time* $t > 0$ *is a product Poisson distribution whose mean is given by the following ODE:*

$$\frac{d\boldsymbol{\lambda}}{dt} = \mathbf{A}\boldsymbol{\lambda} + \mathbf{b}$$

Here the matrix $\mathbf{A}$ contains splicing and degradation rates while $\mathbf{b}$ is a vector of transcription rates. An exact solution of a single-gene system, as a special case, is presented in another work[177]. The discrete VeloVAE model is based on the Poisson initial condition assumption. By **Thm. 2.5**, solving the chemical master equation is equivalent to solving an ODE describing the Poisson mean. In the discrete VeloVAE model, we again use the variational mixture of ODEs as a cell-specific ODE model for the Poisson mean. The generative process is the same as the continuous model except for the last few steps:

$$\begin{aligned}
\boldsymbol{x_{ind}} &\sim Poisson(l \cdot F_{ind}(t; \boldsymbol{\theta}), \Sigma_{\mathbf{r}}) \\
\boldsymbol{x_{rep}} &\sim Poisson(l \cdot F_{rep}(t; \boldsymbol{\theta}), \Sigma_{\mathbf{r}}) \\
\boldsymbol{x} &= \boldsymbol{I}_{\{\boldsymbol{w}=1\}} \odot \boldsymbol{x_{ind}} + \boldsymbol{I}_{\{\boldsymbol{w}=2\}} \odot \boldsymbol{x_{rep}}
\end{aligned}$$

Here, we still have a mixture of basis functions for each gene. However, the mixture components are Poisson distributions instead of Gaussian distributions. Besides, we have an additional cell-wise library size scaling factor l to account for differences in count numbers resulting from different sequencing depths.

Because count numbers are drawn from a Poisson distribution, we define the RNA velocity of a discrete model as $v = \frac{d\lambda_s}{dt}$, where λ_s is the Poisson rate as well as the mean. This is a natural extension from the continuous model.

2.4.3 Branching ODE

Although VeloVAE accurately models bifurcations using a continuous cell state variable, the resulting parameters are not readily interpretable in terms of discrete cell types. Instead of a cell-specific transcription rate and fixed splicing and degradation rates, the branching ODE model assigns each cell type a unique ODE with cell-type-specific transcription, splicing, and degradation rates, and an initial condition determined by the progenitor cell type. The model relies on a directed graph relationship among discrete cell types. We constrain the branching ODE model so that each cell type emerges at a specific time and that the initial conditions of each cell type match the ODE prediction of its parent cell type at the time the child cell type emerges. This provides a qualitative view of the changes in kinetic rates during cell development. We use the branching ODE model to replace the decoder of VeloVAE, providing an alternative, more interpretable generative model.

Model Description The fundamental assumptions we make about branching ODE include:

1. Each cell belongs to exactly one of the cell types $y_1, \ldots, y_k$.

2. At least one of the cell types is a stem cell type. Each non-stem-cell type is a descendant of exactly one other cell type. (Note that we could relax this assumption to allow multiple progenitors, but we choose not to purely for simplicity.) Each cell type has an initial time t_0 when it emerges in the differentiation process.

3. Cells of the same type, y, share identical transcription, splicing and degradation rates $(\alpha_y, \beta_y, \gamma_y)$.

With these assumptions, we can summarize differentiation with directed cell type transition graph $G = (V = \{v_1, \ldots, v_k\}, E)$, called a transition graph. Each cell type corresponds to a vertex in G, and each edge (u, v) represents the relation of u differentiating into v. By our second assumption, G is composed of one or multiple trees. The kinetic functions retain the same analytical form, except for type-specific rate parameters and initial conditions. Furthermore, the equations are defined recursively because the initial condition of any non-

stem-cell type depends on its progenitor cell type.

$$u(t,y) = u_0(y)e^{-\beta_y \tau} + \frac{\alpha_y}{\beta_y}\left(1 - e^{-\beta(y)\tau}\right) \tag{2.8}$$

$$s(t,y) = s_0(y)e^{-\gamma(y)\tau} + \frac{\alpha_y}{\gamma_y}\left(1 - e^{-\gamma_y \tau}\right) + \frac{\alpha_y - \beta_y u_0(y)}{\gamma_y - \beta_y}\left(e^{-\beta_y \tau} - e^{-\gamma_y \tau}\right) \tag{2.9}$$

$$u_0(y) = \begin{cases} u_{init} & \text{y is a stem cell type} \\ u(t_0(y), \text{par}(y)) & \text{otherwise} \end{cases} \tag{2.10}$$

$$s_0(y) = \begin{cases} s_{init} & \text{y is a stem cell type} \\ s(t_0(y), \text{par}(y)) & \text{otherwise} \end{cases} \tag{2.11}$$

Here, $\text{par}(\cdot)$ refers to the parent vertex in the transition graph. Note that τ is redefined as $\tau := t - t_0(y)$ for each cell type y, i.e. τ is the time duration starting from the initial time of each cell type.

Inferring the Transition Graph A challenging aspect of applying branching ODEs is the absence of a transition graph. In many cases, we have some prior knowledge of the transition graph, but it is often more desirable to infer the graph directly from the data when possible. Therefore, we apply a computational method to infer the transition graph based on cell time and states. Since the transition graph is a collection of arborescences (i.e., rooted trees in a directed graph), we can solve the problem using simple graph algorithms.

First, we partition the cells into distinct clusters. Specifically, we perform Leiden clustering with a low resolution on the UMAP coordinates of the data.

Next, we build a complete subgraph in each partition. Let $\mathcal{I}$ be the set of all cells and $\mathcal{I}(y)$ be the set of all cells of type y in a partition. For each cell $i \in \mathcal{I}$ with time t_i, we take a time window $[t_i - \delta_1, t_i - \delta_2]$ and find k nearest neighbors based on cell state $\mathbf{c}$. Denote $\mathcal{J}_i$ as the set of neighbors of i. Then, for any two cell types y and z, the empirical transition probability from y to z is defined as

$$P(y,z) = \frac{\sum_{i \in \mathcal{I}(z)} |\mathcal{J}_i \cap \mathcal{I}(y)|}{\sum_{i \in \mathcal{I}(z)} |\mathcal{J}_i|} \tag{2.12}$$

In other words, we group cells into the cell types and count the number of transitions from any cell type to any other type.

Finally, we apply Edmond's algorithm[178] to find the maximum spanning arborescence in each partition. The earliest cell type is choosen to be the root. The algorithm starts by picking the parent y of each non-root vertex z greedily, i.e. $y = \arg\max_v P(v,z)$. Next, it checks loops, collapses loops into super-vertices and is recursively applied to the new graph

until no loop exists. Finally, it breaks the loops after each level of recursion.

Training a Branching ODE We assume that the cell time has already been inferred from VeloVAE. Thus, the branching ODE is a regression model with an analytical form shown in **Eq. 2.8, 2.9**. To train the model, we find the cell-type-specific rate parameters $\boldsymbol{\theta}$ that maximize the Gaussian likelihood, which is equivalent to minimizing a Mahalanobis distance:

$$\min_{\boldsymbol{\theta}} \sum_{i=1}^{n} (\hat{\mathbf{x}}(t;\boldsymbol{\theta}) - \mathbf{x})^T \boldsymbol{\Sigma} (\hat{\mathbf{x}}(t;\boldsymbol{\theta}) - \mathbf{x}) \tag{2.13}$$

Here, $\hat{\mathbf{x}}$ is predicted expression level. We train using mini-batch stochastic gradient descent with the ADAM optimizer.

2.4.4 Evaluation Metrics

Due to limitations in measurement technology, we are unable to directly measure the true biochemical reaction (transcription, splicing, and degradation) rates. This makes it difficult to quantitatively assess the correctness of inferred RNA velocity. However, based on experimental verification, we have prior knowledge of certain aspects of cell development. Evaluating the performance of RNA velocity remains an open question. Here, we take a step toward a more systematic and quantitative yet biologically meaningful evaluation of RNA velocity methods. We apply both previously published and newly defined evaluation metrics to assess the inferred RNA velocity and cell time.

Time Correlation Given true cell time $T = [t_1, \ldots, t_n]$ and inferred cell time $\hat{T} = [\hat{t}_1, \ldots, \hat{t}_n]$, we define time correlation as the Spearman correlation coefficient between T and $\hat{T}$.

Cross-Boundary Direction Correctness (CBDir) In a previous work[99], the authors quantitatively measured the accuracy of velocity flow on any 2D visualization. Denote $\mathcal{C}$ as the set of all cell types. For any two cell types $A, B \in \mathcal{C}$, we define (A, B) to be a transition pair if and only if B is a descendant cell type of A in a cell developmental process. The CBDir metric takes a set of know transition pairs $\mathcal{T} = \{(A, B) : A, B \in \mathcal{C}\}$ as input and computes an average directional accuracy via cosine similarity. For any cell in $c \in A$ with

transition pair (A, B), its CBDir is defined as:

$$\text{CBDir}(c) := \frac{1}{|nbr(c) \cap C_B|} \sum_{c' \in nbr(c) \cap C_B} \frac{\boldsymbol{v_c}^T(\boldsymbol{x_{c'}} - \boldsymbol{x_c})}{\|\boldsymbol{v_c}\| \|\boldsymbol{x_{c'}} - \boldsymbol{x_c}\|}$$

Here, $\boldsymbol{v_c}$ is the projection of RNA velocity to a low-dimensional space, $\boldsymbol{x_c}$ is the coordinates of c in the same space, and $nbr(c)$ is the set of all neighbors of c in a KNN graph. While the original CBDir assumes KNN built from gene expression similarity, we replace it with a spatial KNN graph to evaluate the correctness of cell velocity in a physical space. The overall CBDir of all known transition pairs is defined as

$$\text{CBDir} := \frac{1}{|\mathcal{T}|} \sum_{(A,B) \in \mathcal{T}} \frac{1}{|C_A|} \sum_{i \in C_A} \text{CBDir}(c)$$

k-Step CBDir (k-CBDir) The CBDir metric is the first one directly measuring the correctness of velocity flow. It connects visual perception of velocity flow to a quantity. We think it beneficial to extend it for the following reasons:

1. The velocity in CBDir is only a projection to a low-dimensional space. Hence, it's not a direct measure of velocity at a gene expression level.

2. Only the direct neighbors in the KNN graph are considered as targets, i.e. future transcriptomic states. In fact, such assumption might not hold due to stochastic noise in the dataset. Directly connected neighbors on a KNN graph are too similar at the transcriptomic level to be considered as a future direction.

3. CBDir doesn't take the ordering of cells into account. Time order is important because any real-world system must proceed with increasing time.

Due to all the limitations described above, we devised k-step CBDir (k-CBDir) metric. Given a cell $c \in C_A$ with transition pair (A, B), K-CBDir of c is defined as follows:

$$k\text{-CBDir}(c) := \frac{1}{|nbr_k(c) \cap C_B|} \sum_{c' \in nbr_k(c) \cap C_B} (-1)^b \frac{|\boldsymbol{v_c}^T(\boldsymbol{x_{c'}} - \boldsymbol{x_c})|}{\|\boldsymbol{v_c}\| \|\boldsymbol{x_{c'}} - \boldsymbol{x_c}\|}$$

where

$$b := I_{\{t_{c'} > t_c \wedge \boldsymbol{v_c}^T(\boldsymbol{x_{c'}} - \boldsymbol{x_c}) > 0\}} \tag{2.14}$$

Here, we replace direct neighbors with k-step neighbors, $nbr_k(c)$, on a KNN graph. Besides, $\boldsymbol{x_c}$ and $\boldsymbol{v_c}$ can be from either a low-dimensional embedding or the spliced counts.

Given all cells c, we can calculate a final scalar metric by averaging K-CBDir over all c's:

$$k\text{-CBDir} = \frac{1}{|\mathcal{T}|} \sum_{(A,B)\in\mathcal{T}} \frac{1}{|C_A|} \sum_{i\in C_A} k\text{-CBDir}(c)$$

Finally, we define Generalized CBDir (GCBDir) as the set $\{k\text{-CBDir} : k = 1, 2, \ldots, K\}$, i.e. set of all k-CBDir from 1 to K.

2.5 Experiments

2.5.1 Preliminary Benchmark Study

To validate our method, we implemented VeloVAE as a python package and performed benchmarking experiments on datasets with known biological truth. As a preliminary benchmark study, we compared VeloVAE with a popular SOTA method, scVelo. Our results show that VeloVAE outperformed scVelo in data reconstruction and time correlation across six datasets (**Supplemental Table A.1**). Since our initial publication of VeloVAE in 2022[102], different research groups have spent much efforts into improving RNA velocity modeling and a number of computational methods have emerged[99, 104, 101, 105, 100]. Hence, we did more comprehensive benchmarking by comparing different variants of VeloVAE and the other methods. In addition to continuous-count models, we also evaluated the discrete-count version of VeloVAE and PyroVelocity[105], since these are the only two methods directly modeling raw count numbers. We evaluated all models in terms of data reconstruction, how accurately they recover latent time and how well velocity flows match the biological truth. We also directly looked into each individual genes and compare the fitting qualitatively. We trained VeloVAE with default hyper-parameters with minimal changes (**Supplemental Table A.2-A.4**). Our results show that VeloVAE achieves comparable or better performance across all performance metrics, while existing methods are less versatile (**Fig. 2.3, Supplemental Fig. A.1**).

To evaluate the accuracy of latent time inference, we selected scRNA-seq datasets with cells sampled from multiple time points. These time points usually have rather coarse granularity (e.g., day 7 and day 14), and cells captured at the same time may span a wide range of developmental stages. VeloVAE consistently achieves comparable or better performance in terms of latent time accuracy, with scVelo often inferring latent time that is anticorrelated with real time (**Fig. 2.3a**). Although scVelo and VeloVI infer latent time separately for each gene, they provide a post-hoc procedure for estimating a single global time for each cell. Using this global time for comparison with our methods casts these methods in the

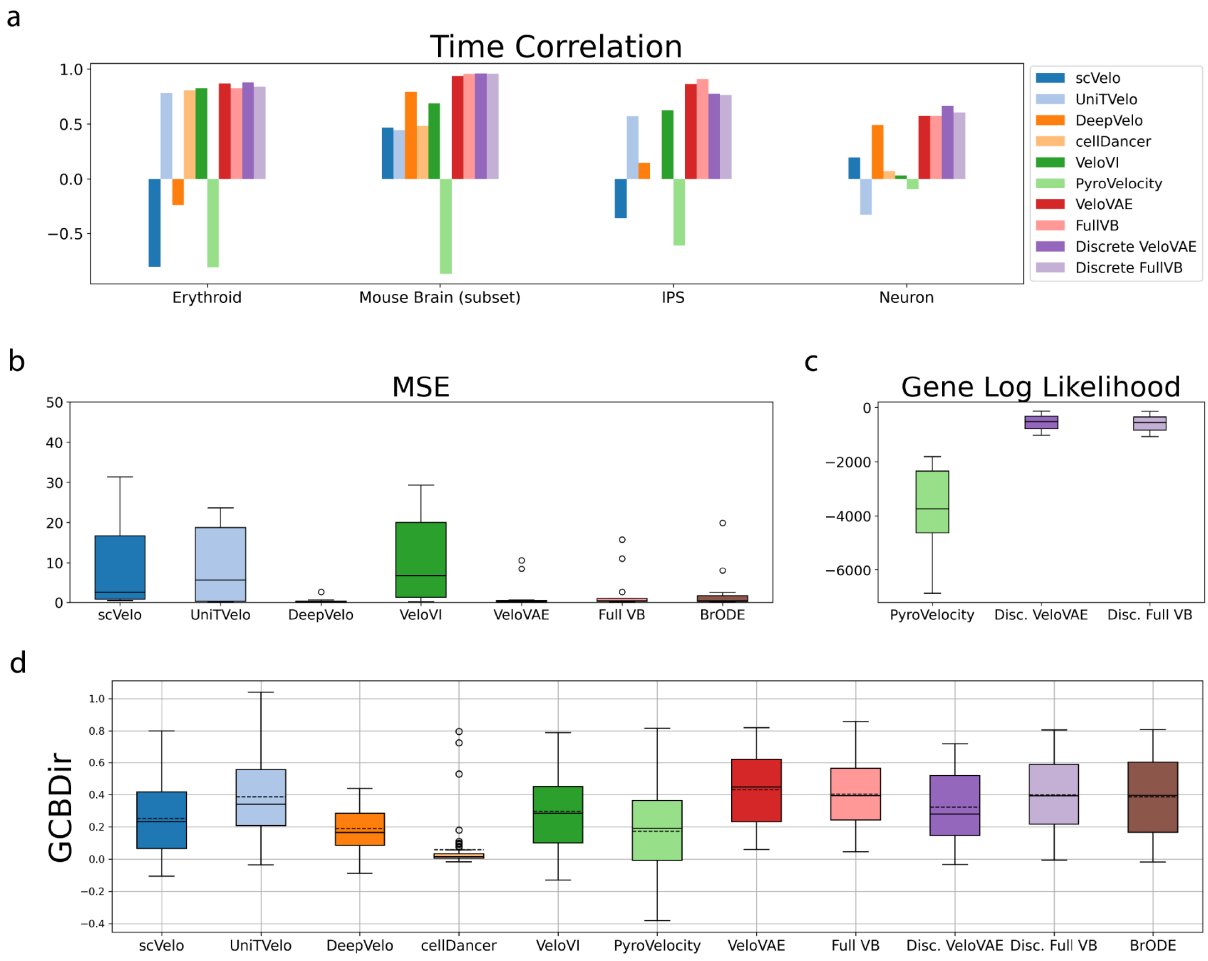

Figure 2.3: **Benchmarking Results across 12 Datasets. (a)-(d)** Quantitative performance comparison. We report the correlation between inferred latent time and true capture time (when available) **(a)**, mean squared error (MSE) between the observed and predicted (u, s) counts for continuous-count models **(b)**, gene log likelihood for discrete-count models **(c)** and generalized cross-boundary direction correctness across 12 benchmarking datasets **(d)**.

best possible light because the global time is more robust than the gene-specific latent times. The low time correlation from scVelo may be partly explained by inconsistency among the different notions of time fitted for each gene. To investigate this further, we computed the average time correlation between scVelo's and VeloVI's gene-specific and global latent time. As **Supplemental Fig. A.2** shows, the correlation between the global latent time and the latent time for each gene is indeed quite low; the latent time values for many genes are even anticorrelated with global latent time. We also note that cellDancer and DeepVelo do not include cell time in their models, but rather perform a post-hoc global time estimation based on velocity.

To quantify model performance on prediction, we calculated mean squared error (MSE) for continuous-count models and and log likelihood (LL) for discrete-count models (**Fig. 2.3b, c**). For VeloVAE, DeepVelo, VeloVI and PyroVelocity, we calculated these metrics on a held-out test set not used during training. Note that scVelo and UniTVelo cannot perform out-of-sample prediction, so we were not able to evaluate it on a held-out test set. CellDancer does not directly model count data and its training objective is cosine similarity, so we do not compute its MSE. Our results show that VeloVAE achieves comparable or better data fit than other methods in the evaluation. Furthermore, this better performance does not come from overfitting the training dataset - VeloVAE shows good performance on the held-out datasets. Finally, we show that VeloVAE achieved top performance in terms of generalized CBDir across 12 benchmarking datasets and neighborhoods up to 5 steps in the KNN graph (**Fig. 2.3d, Supplemental Fig. A.1**).

In addition, we also analyzed the latent space learnt by VeloVAE. We qualitatively demonstrate that the semantic space of learned cell latent space corresponds to different cell types manually annotated based biologically verified marker genes (**Supplemental Fig. A.3, A.4**). Another important aspect of our Bayesian approach is uncertainty estimation. Thus, we investigated the variance of cell state and cell time by visualizing the coefficient of variation as a heatmap on UMAP[78] coordinates. The results show that cells facing fate decision is generaly have a higher coefficient of variation (**Supplemental Fig. A.5, A.6**). For example, in the mouse pancreas, the Ngn3^{-} cells are either developing into Ngn3^{+} cells or Ductal cells, and pre-endocrine cells are facing cell fate decision to be one of alpha, beta, delta and epsilon cells[98](**Supplemental Fig. A.5a, A.6a**). Another example is the developing mouse brain, where the neural tube and neural crest cells have the highest uncertainty (**Supplemental Fig. A.5c, f, A.6c, f**).

2.5.2 Modeling Multi-Lineage RNA Velocity

We designed the VeloVAE model to relax several restrictive assumptions of previous RNA velocity approaches. Thus, we expect the model to exhibit greater expressiveness, enabling it to capture qualitative properties of gene expression changes that previous approaches cannot. To assess this, we fit VeloVAE and other models to two representative datasets—mouse pancreas and brain—and examined the resulting model fits. We identified two types of qualitative behaviors that existing approaches cannot accurately model, whereas VeloVAE can.

Late Induction and Early Repression We observed that genes with a late, short, or missing induction phase are particularly prone to being fit incorrectly by previous methods,

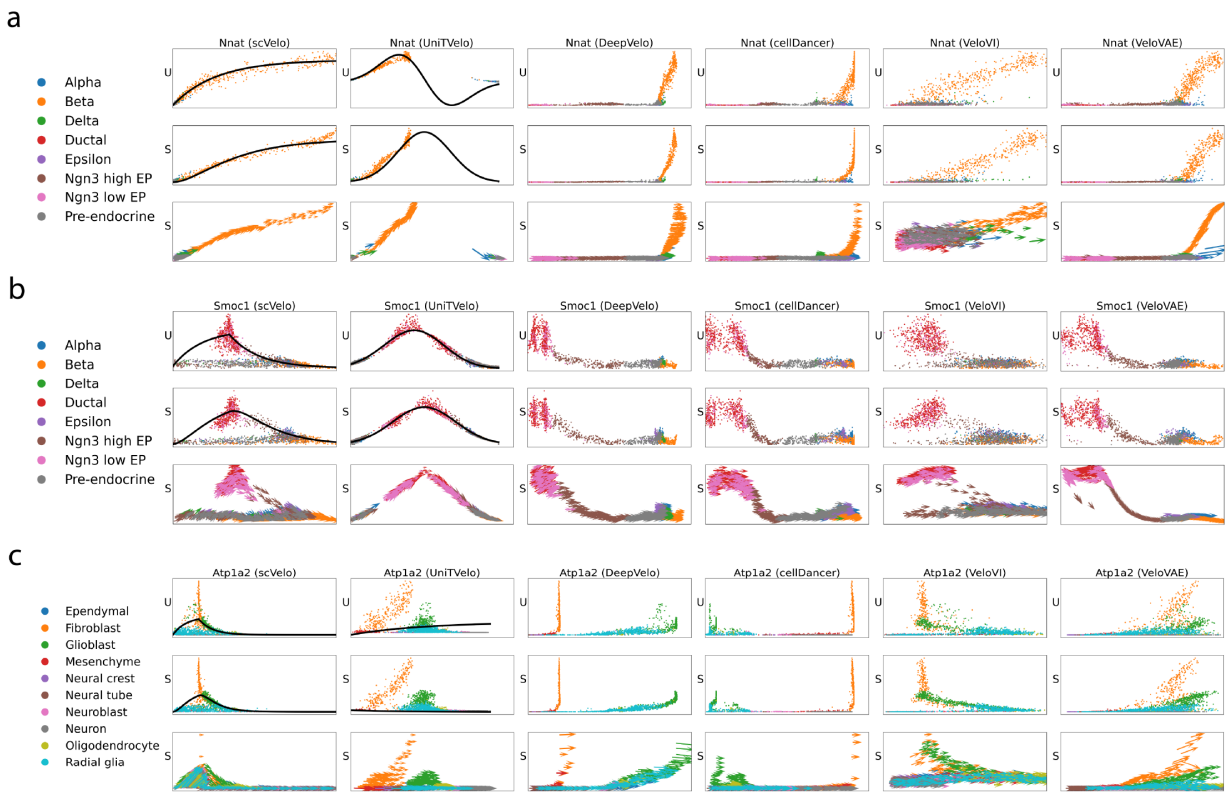

Figure 2.4: **VeloVAE Significantly Models Complex Gene Expression Dynamics.** Examples of individual genes fit by different RNA velocity methods for a late-induced gene **(a)** and early-repressed gene **(b)** in pancreas and a gene with branching dynamics in subsampled mouse brain **(c)**. Gene fits are shown for both u and s values, with inferred latent time on the x-axis. Fitted values from scVelo and UniTVelo are shown as lines, with observed data values shown as points. Only fitted values are shown for the other methods, because the fit is a point cloud (rather than a line) that would completely cover the observed data. Points are colored by published cluster assignments, with the same colors as in the corresponding UMAP plots. The arrow length and direction indicate the velocity inferred for each cell.

due to each own limitation. First, for scVelo, the lack of a common time scale, combined with the assumption that induction starts at $t = 0$, also leads to frequent errors in estimating the overall direction of a gene (increasing or decreasing). Second, for UniTVelo, the absence of a mechanistic model results in unrealistic predictions. Last, velocity estimation methods like DeepVelo and cellDancer do not incorporate time into model fitting and thus, produce velocity that is inconsistent with human interpretation. A dataset from the mouse pancreas[179] illustrates these behaviors.

All methods in our study yield latent time values and stream plots for the pancreas dataset that are consistent with prior knowledge. However, by inspecting the fits for individual

genes, we observe that previous methods either produce fits that contradict biological reality or generate uninformative RNA velocity. **Fig. 2.4a, b** show two sample genes, *Nnat* and *Smoc1*. The *Nnat* gene is not turned on until the pre-endocrine cells appear, whereas *Smoc1* is immediately switched off at the beginning of the differentiation process. We can see that scVelo, UniTVelo and VeloVI fail to detect late induction in *Nnat*. ScVelo assigns the latent time to zero for almost all cell types except for beta cells, while UniTVelo assigns some cells to the repression phase. The time inference from VeloVI is more accurate, but still mixes beta cells with endocrine progenitor cells. On the other hand, the post-hoc latent time from DeepVelo and cellDancer are consistent with the true developmental process, but the velocity does not match human interpretation that the mRNA count is increasing fast for *Nnat*. For *Smoc1*, scVelo and UniTVelo incorrectly rearrange the order of progenitor and descendent cell types in an effort to force the gene to have a induction phase. The velocity from DeepVelo and cellDancer remains uninformative as there is no significant decreasing trend in the endocrine progenitor (Ngn3 low EP and high EP) cells. In contrast, VeloVAE not only correctly inferred the late induction pattern for *Nnat* and the early repression pattern for *Smoc1* but also produced perceptually correct velocity flow.

Cell Type Bifurcations Many cell differentiation processes produce multiple descendant cell types from a single progenitor type, but previous RNA velocity approaches model only a single cell type. By including a cell-specific latent state, VeloVAE can model the continuous emergence of multiple cell types from a single progenitor type. For example, in the pancreas dataset, the *Nnat* gene is upregulated as cells differentiate toward the beta cell fate, but not in any other cell type (**Fig. 2.4a**).

As another example, we fit all models on a developing mouse brain atlas[180]. For clarity, we subsampled the dataset to include only cell types arising from the neural tube and neural crest. In this system, neural tube cells develop into radial glia. Some of the radial glia cells differentiate into neuronal cell types, while the others give rise to the glial lineage, including glioblasts, oligodendrocytes, astrocytes, and ependymal cells. The neural crest cells develop to become fibroblast cells. In short, this is a complex system with many distinct lineages emerging. Nevertheless, VeloVAE can accurately model the complex, multi-lineage dynamics of genes in this system. For example, VeloVAE accurately models the behavior of the *Atp1a2* gene (**Fig. 2.4c**), which is turned on in the fibroblast and glial cells with distinct transcription rates and remains off in the neuronal cells differentiated from radial glia. In contrast, scVelo rearranges the latent time values of the cells in a vain attempt to fit the *Atp1a2* gene into a single induction and repression cycle. VeloVI reverses the time order as fibroblast cells flow backwards to mesenchymal cells. We trained UniTVelo with

unified cell time and the cell time is generally correct. However, without explicitly modeling branching, the parametric model from UniTVelo underfits the count data. Neither DeepVelo nor cellDancer accurately recovered the cell time for the fibroblast branch, and they suffer from the same problem of uninformative velocity.

2.5.3 Cell Type Transition Graph Inference and Branching Differential Equation Model

Having established the accuracy of the results from VeloVAE, we used these results to infer cell type transition graphs. We show three examples of these graphs (**Fig. 2.5a-c**), inferred from the pancreas, iPS reprogramming, and mouse brain datasets from **Fig. 2.3**. The inferred cell type transition graphs match closely with biological expectations for these systems, even for the complex mouse brain dataset.

To train the model, we first obtain the time and state assignments for each cell by training VeloVAE. Next, we infer a transition graph describing the progenitor-descendant relations among cell types. Finally, we fix the encoder of VeloVAE (so that latent time and cell state estimates are fixed) and estimate the parameters of the branching ODE model. We perform this parameter estimation by maximizing the Gaussian likelihood of all genes under the branching ODE model, which is equivalent to minimizing the Mahalanobis distance between model fit and observed data.

We trained branching ODEs on pancreas, iPSC and subsampled mouse brain datasets. The cell type transition graphs are all consistent with prior knowledge (**Fig. 2.5a-c**). For example, our computational method successfully finds the expected cell differentiation path in pancreatic development, starting from ductal cells and branching into α, β, δ and ϵ cells (**Fig. 2.5a**).

In addition, the branching ODE is able to infer different and asynchronous gene expression kinetics in multiple branches and successfully captures different rates in different branches in these datasets. For example, *Ppp1r1a* has almost all transcription activity in β cells (**Fig. 2.5d**), which was verified by previous studies[181, 182]. The *Rnf130* and *1500009L16Rik* genes similarly show significant branching trends that are accurately modeled by the branching ODE (**Fig. 2.5e**). The branching ODE model also accurately fits genes that show differential kinetics among lineages that emerge during induced pluripotent stem cell reprogramming. For example, *Nr2f1* is strongly and specifically upregulated in neural-like cells; *Hsp90aa1* is upregulated moderately in Trophoblast and strongly in IPS cells; *Krt7* is upregulated in epithelial-like and trophoblast-like cells with differing expression levels (**Fig. 2.5f, g**).

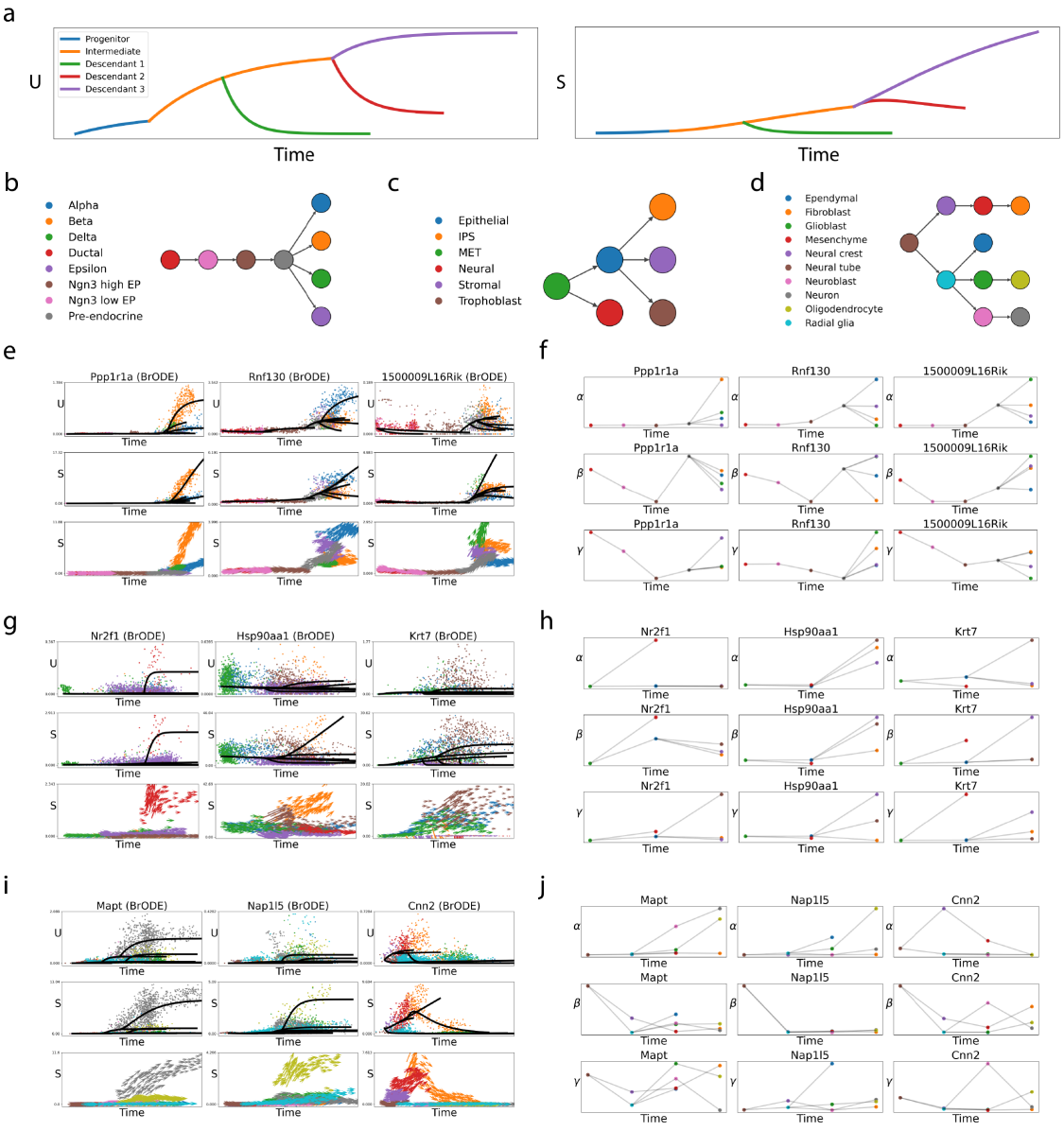

Figure 2.5: **Cell Type Transition Graph Inference and Branching Differential Equation Model.** **(a)-(c)** Inferred cell type transition graphs from pancreas **(a)**, iPSC **(b)** and subsampled mouse brain **(c)** datasets. In each graph, a vertex represents a cell type and a directed edge points from a progenitor cell type to its immediate descendant(s). **(e)-(i)** Examples of individual genes fit by the branching ODE model for pancreas **(d)**, iPSC **(f)** and subsampled mouse brain **(h)** datasets. Each column represents a gene and plots the unspliced count, spliced count and RNA velocity versus time from top to bottom. The branching ODE fits are shown as solid lines. Cell-type-specific rate parameters inferred by branching ODE model for pancreas **(e)**, iPSC **(g)** and subsampled mouse brain **(i)** datasets are also shown. Each column represents a gene and plots the transcription (α), splicing (β) and degradation (γ) rates from top to bottom. Each point represents a cell type, the types are arranged in chronological order, and progenitor-descendant relationships are indicated with lines.

Another example is *Mapt* from the mouse brain dataset (**Fig. 2.5h, i**). The gene is upregulated strongly in neurons and subsequently transcribed at much lower levels in oligodendrocytes, which coheres with previous studies[183]. The *Napl15* gene shows the opposite trend, with high transcription in oligodendrocytes and low but detectable transcription in glioblasts and neurons (**Fig. 2.5i**). As another example, the *Cnn2* gene is most highly transcribed in the early transition from neural crest to mesenchyme and then to fibroblast (**Fig. 2.5i**). The cell-type-specific rate parameters inferred by fitting the branching ODE describe the differences in transcription, splicing, and degradation that cells undergo as they differentiate (**Fig. 2.5e, g, i**).

2.5.4 VeloVAE Accurately Models Cell Differentiation Across the Whole Mouse Brain.

Single-cell technologies now enable large-scale measurements of differentiating cells across entire organs or organisms. However, modeling the emergence of cell fates at this scale remains extremely challenging with existing computational approaches. For example, La Manno et al. recently published an atlas of the entire developing mouse brain but did not perform any RNA velocity analysis[180]. The dataset contains 292,495 cells representing 22 major types and 798 subtypes, collected between embryonic days 7 and 18. Starting from late gastrulation, the neuronal lineage develops from neural tube and neural crest cells. Neural crest cells differentiate into fibroblasts, while neural tube cells differentiate into radial glia cells, which then give rise to both neuronal and glial cell types.

As **Fig. 2.6a** shows, VeloVAE captured the cellular dynamics in all major lineages. This is also manifested in the gene expression dynamics (**Fig. 2.6c**). Furthermore, the latent time from VeloVAE is qualitatively close to the cell capture times, reflecting the true time order of the cells (**Fig. 2.6d**).

In addition to the velocity and latent time inferred by VeloVAE, we also analyzed the inferred cell states. Recall that the latent cell state inferred by VeloVAE is a low-dimensional embedding of mRNA counts of each cell. To visualize these cell states, we generated 3D plots where the x-y plane is the 2D UMAP[78] coordinates of cell states and z axis corresponds to the cell time (**Fig. 2.6 b, Supplementary Fig. A.3f**). As a comparison, we generated a 3D quiver plot whose x-y plane is 2D t-SNE[77] coordinates provided by the authors and z axis is still the cell time (**Supplemental Fig. A.4f**). In this way, we can visualize the whole differentiation process. In addition, our variational posterior also contains the variance of the cell state. We define the cell state uncertainty as the multi-variate coefficient of variation (CV)[184] of the cell state and visualize it in a t-SNE plot (**Fig. 2.6e, Supplemental Fig.**

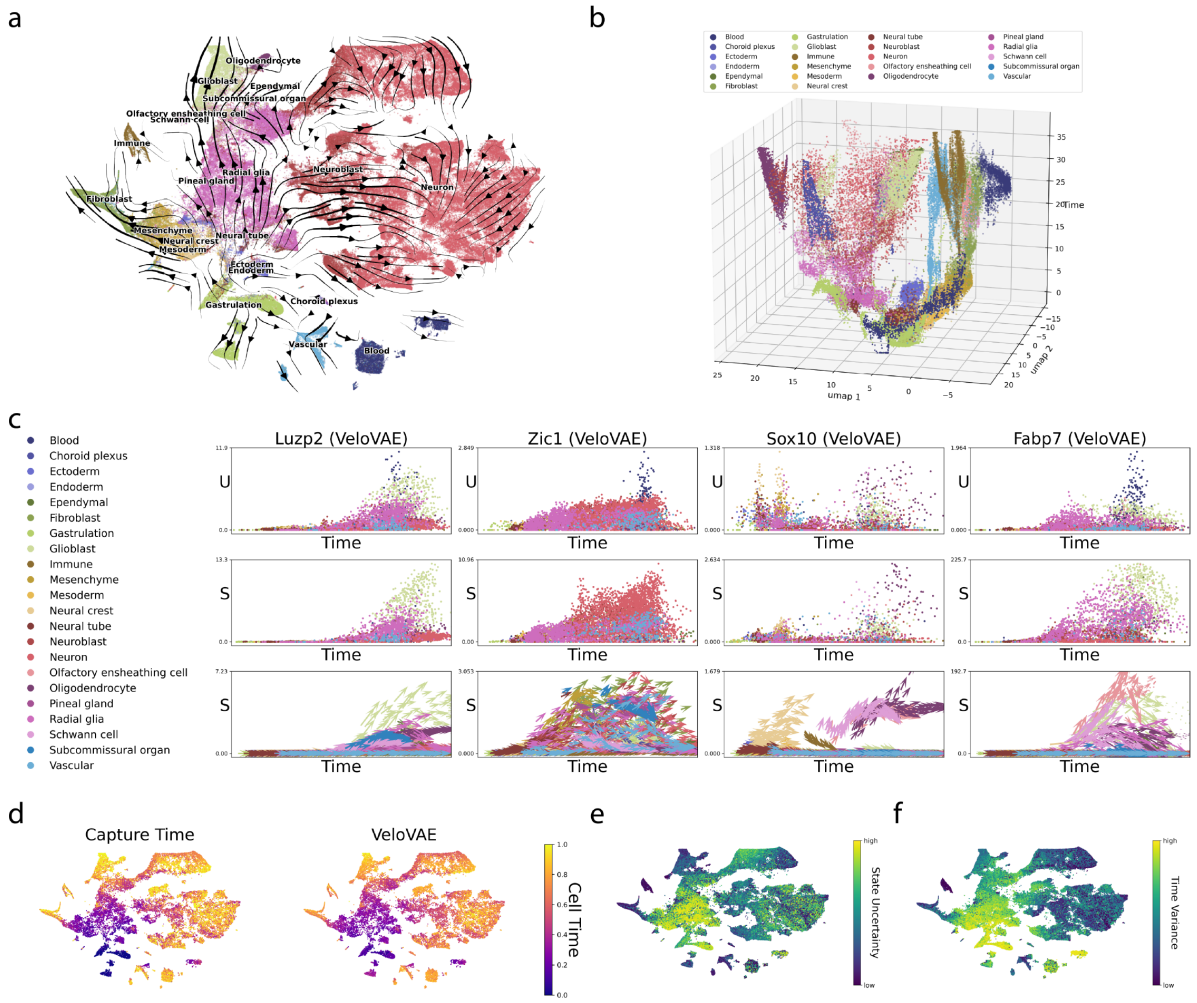

Figure 2.6: **VeloVAE Resolves Cellular Dynamics of Multiple Lineages across the Entire Developing Mouse Brain.** **(a)** t-SNE plots colored by published cell types overlaid with velocity inferred by VeloVAE. **(c)** Examples of individual genes fit by VeloVAE. Gene fits are shown for both u and s values, with inferred latent time on the x-axis. Only fitted values are shown for VeloVAE, because the VeloVAE fit is a point cloud (rather than a line) that would completely cover the observed data. Points are colored by published cluster assignments, with the same colors as in the corresponding UMAP plots. Note that the x-axis values are different for scVelo and VeloVAE because they infer different latent times. The arrow length and direction indicate the velocity inferred for each cell. **(d)** t-SNE plots colored by true capture time and latent time inferred by VeloVAE. **(e)-(f)** t-SNE plots colored by the cell state uncertainty **(e)** and cell time variance **(f)** from VeloVAE.

A.5f). Similarly, we obtained time variance for each cell (**Fig. 2.6f, Supplemental Fig. A.6f**).

From these plots, we noticed an interesting phenomenon: multi-potent progenitor cells often have very high cell state uncertainty. For example, radial glia cells in mouse brain

development can differentiate into both neuronal and non-neuronal types, so they have high cell state uncertainty. In contrast, the more differentiated cell types show relatively lower cell state uncertainty. This suggests that the cell state distributions learned by VeloVAE can capture biologically meaningful uncertainty in the states of cells undergoing fate decisions.

2.5.5 VeloVAE Accurately Models Organ Development in Whole Mouse Embryos.

The challenges of studying cell differentiation with single-cell data become particularly acute at the scale of an entire organism. Despite their success on simple biological systems, RNA velocity methods face challenges when they are applied to complex systems with a large number of cells. Here, we demonstrate that our method is able to handle dataset as complex as an entire embryo. Cao et al. performed single-cell RNA-seq on 61 entire mouse embryos sampled from E9.5-E13.5, the developmental period when mouse organogenesis occurs[82]. The dataset contains 1,191,071 cells after preprocessing, which fall into 38 major cell types and can be divided into 9 major lineages. Pseudotime analysis and RNA velocity analysis have been performed on this dataset, but both analyses were highly manual processes that required separate curation of dozens of cell subsets[82, 185].

In principle, the continuous cell state variable of VeloVAE is sufficiently expressive to provide a single model of the differentiation potential for an entire organism. To investigate this, we trained VeloVAE on the entire mouse organogenesis dataset. Because of the large size and cellular diversity of this dataset, we used a larger batch size of 1024 and increased the dimension of cell state from 5 to 30. Our results show that VeloVAE discovered meaningful dynamics on the latent cell state space representing the whole developmental process (**Fig. 2.7a, Supplemental Fig. A.7, A.8**). To examine the results in more detail, we ran UMAP individually on neural and mesenchymal cell lineages that were identified in the initial paper (**Fig. 2.7b, c**). Results on other lineages also suggest cell differentiation directions and temporal order (**Supplemental Fig. A.9, A.10**). The cell time and state uncertainty analysis, together with RNA velocity visulization, also show that cells with most differentiation potential, have the highest uncertainty (**Supplemental Fig. A.11, A.12**). Importantly, we performed this subset analysis purely for visualization purposes–the cell time, cell state, and kinetic rate parameters were estimated only once using all cells jointly. In addition, we demonstrate that our model can fit genes with branching dynamics in different cell types (**Fig. 2.7d, e**) and at gene level, the inferred rate parameter values have meaningful biological interpretation (**Supplemental Fig. A.13**).

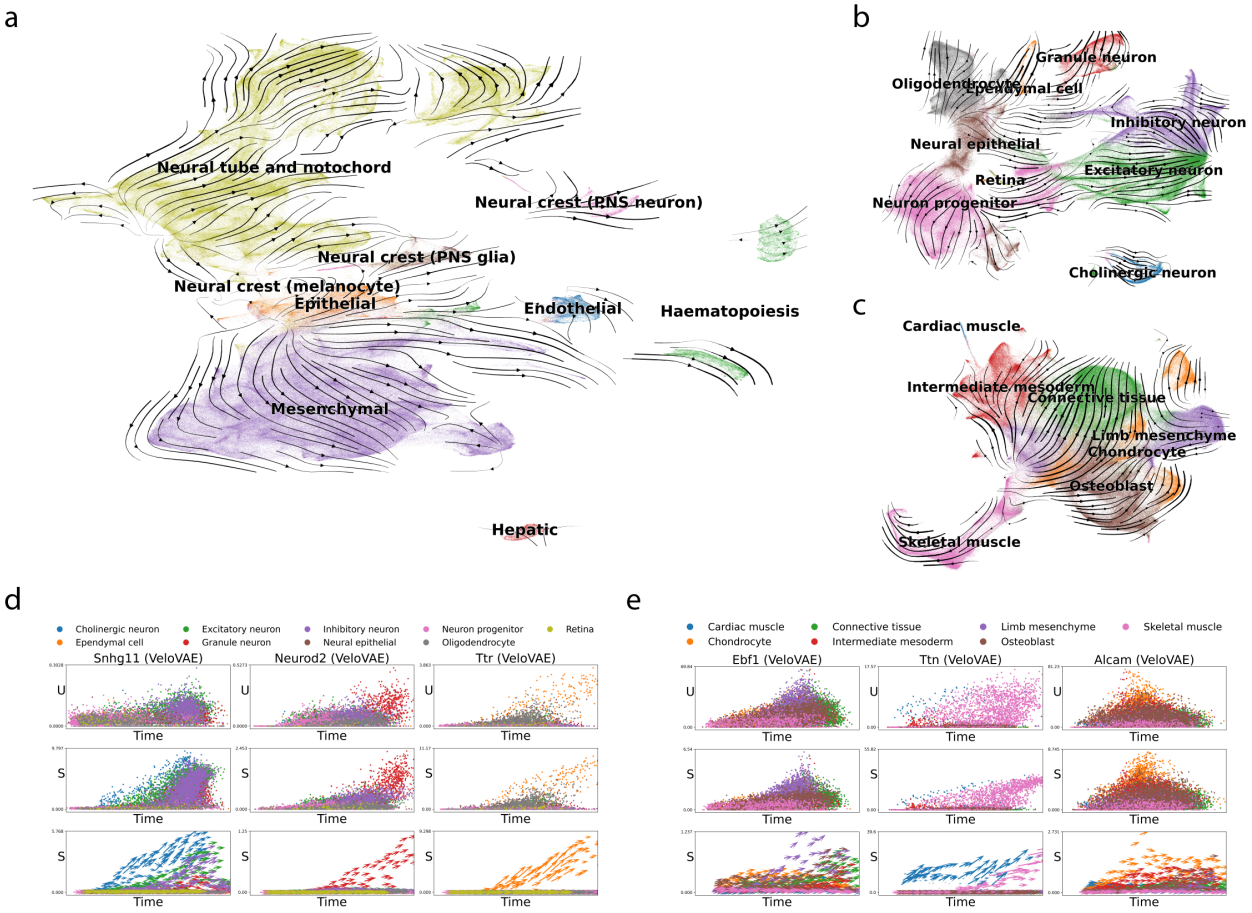

Figure 2.7: **Whole Mouse Embryo Analysis.** **(a)** UMAP of entire mouse embryo dataset, colored by broad cell class. **(b)-(c)** UMAP plots colored by published cell types overlaid with velocity inferred by VeloVAE for neuronal **(b)** and mesenchymal **(c)** trajectories. The UMAP coordinates are computed using only cells from each trajectory, but the velocity estimates are from a single VeloVAE model fit on all cells. **(d)-(e)** VeloVAE fits for selected genes from the mesenchymal **(d)** and neuronal **(e)** trajectories.

2.6 Discussion

In this chapter, we approach the problem of uncovering the temporal mechanisms of gene expression through the lens of RNA velocity and treat it as a latent variable problem. We developed a deep generative model, VeloVAE, which employs variational Bayesian inference to estimate cell differentiation progress, cell state, and kinetic rate parameters in a statistically principled manner. The expressiveness of neural networks enables the model to adapt to various types of gene expression kinetics in biological differentiation systems of varying complexity. Moreover, this model's expressiveness does not come at the cost of interpretability; it still learns a set of rate parameters with direct biophysical interpretations. Another

key advantage of VeloVAE is its ability to perform inference using only unspliced and spliced mRNA counts from a set of cells while also being capable of incorporating additional prior information, such as cell capture times, when available. Additionally, the use of mini-batch stochastic gradient descent allows the method to process large datasets without requiring all cells to be loaded into memory simultaneously

Through a series of benchmark studies and experiments on real scRNA-seq datasets, we demonstrate that VeloVAE significantly improves the usefulness and interpretability of RNA velocity results in several ways. First, the latent times inferred by the model are sufficiently accurate to order cells according to their differentiation progress—surpassing the accuracy of pseudotime inference in some cases. Second, the ability to use cell capture times as a prior distribution places the inferred cell times on a time scale with known units (e.g., hours or days). Knowing the time units also provides the kinetic rate parameters with a more direct interpretation. Furthermore, the fitted and extrapolated values are now qualitatively accurate for individual genes, in contrast to previous RNA velocity methods, where individual gene fits are often poor even when global latent time and stream plots appear reasonable. This improvement enables new applications where understanding the direction and rate of change for individual genes is crucial, rather than relying solely on qualitative global visualizations. For example, researchers can now directly compare the transcription times of different genes, revealing the relative order in which genes are activated. Additionally, we show that the latent time results and future cell state predictions are sufficiently accurate to infer transition graphs among cell types in many cases. Finally, our branching ODE model provides important insights into how transcription, splicing, and degradation rates change during cell type bifurcations, offering valuable perspectives on the regulatory mechanisms governing cell fate decisions.

In summary, this chapter presents new methods and results demonstrating how deep learning and Bayesian inference can be leveraged to enhance our understanding of mRNA splicing kinetics at single-cell resolution.

CHAPTER 3

Inferring Spatial Mechanism of Gene Expression

3.1 Motivation

Cell fate transitions are fundamentally spatiotemporal processes regulated by factors that vary over space, including secreted signaling factors that diffuse to nearby cells and bind receptors, extracellular matrix interactions, and tight junctions between adjacent cells. While temporal activities play an important role in cell fate decision, the spatial context is pertinent as well. As cells divide over time, daughter cells arise in locations spatially adjacent to the mother cell. Moreover, cells often develop in defined spatial directions over time – for example, stratified epithelium forms from the basal layer to the luminal layer[186, 187]. During early development, many cells migrate to specific locations to form complex tissues. For example, during embryonic brain development, radial glia cells in the ventricular zone of the cortex migrate radially toward the pial surface[188]. It is during this spatial migration that stem cells develop into neurons in the cortical layer. Hence, understanding the impact of spatial organization of cells on gene expression is key to unraveling the rules of cell fate decision in space. This has profound implication in downstream applications in medical science, such as predicting tumor growth and tissue culturing.

Spatial transcriptomics enables whole-transcriptome or targeted measurement of gene expression and spatial position at cellular or sub-cellular resolution[107, 109, 108, 110], thereby affording a layer of context that is typically lost in techniques that require cell dissociation, such as single-cell RNA sequencing[30, 107, 189, 190]. Other spatial transcriptomic technologies can measure selected genes at single-molecule resolution using multiplexed in situ hybridization[111, 112, 191]. Spatial transcriptomics can measure not only a single slice of tissue but multiple slices from the same 3D sample at a single time point or multiple samples across successive time points.

This novel technology provides the data we need to address a fundamental biological question - how does the spatial niche of a cell influence its molecular state during cell fate transition? However, no existing computational approaches can directly address this question. Methods for analyzing spatially variable gene[114, 115, 116, 192, 193] do not relate spatial patterns to any biochemical process. Pseudotime methods[84, 83, 87] do not model spatial positions, cannot infer direction or rate of change, and are not mechanistic. Approaches based on optimal transport[88, 194] can learn a spatial or temporal coupling between groups of cells, but they require multiple time points, cannot be applied to single-slice data, and are not mechanistic. RNA velocity methods[98, 99, 102, 100, 105, 104] can infer directions and rates of change from a simplified mechanistic model by fitting rate parameters, but do not consider spatial information. Spateo[120] has been used to infer velocity from spatial transcriptomic data using the Dynamo model[121], but it does not incorporate spatial positions during parameter estimation. cell2fate[122] infers velocity modules from dissociated scRNA data and projects them onto spatial coordinates. Neither spateo nor cell2fate uses spatial information during inference, and thus neither can model the relationship between space and time or among neighboring cells. Similarly, STT[123] estimates rate parameters using scRNA data, then projects the resulting velocity vectors onto spatial coordinates. GASTON[124] learns a one-dimensional isosurface from spatial gene expression but does not model spliced and unspliced counts and cannot assign directionality to the isosurface. In summary, while these previous approaches each incorporate spatial information in some way, they do not directly model how a cell's microenvironment affects its gene expression and differentiation. Thus, we would like to develop a computational approach that is mechanistic, models spatial coupling among cells, and can analyze datasets containing as few as one time point. In this chapter, we look at gene expression from a different angle and introduce cell velocity, a spatial counterpart of RNA velocity.

3.2 Technical Contributions

We make the following technical contributions:

1. We are the first to develop a method, TopoVelo, that mechanistically incorporates spatial information and graph neural network techniques into RNA velocity.

2. We propose the concept of cell velocity and demonstrate the connection between RNA velocity and the physical migration of cells during tissue growth.

3. We are the first to show that computational results from TopoVelo can be used for the quantitative study of tissue development. In particular, TopoVelo (1) accurately pre-

dicted average migration speed of neuronal cells in a developing mouse brain, (2) generated meaningful graph attention scores correlated with computational results from cell-cell communication, (3) quantitatively identified three closure points from a E9 developing mouse neural tube and (4) suggested possible spatial dynamics of embryoid cell development.

3.3 RNA Velocity and Cell Development in Space

A common computational setting for spatial transcriptomic data is to assume a cell-by-gene mRNA count matrix, $\mathbf{X} \in \mathbb{R}^{n \times d}$ and a set of spatial coordinates $\{(x_i, y_i) : i \in [n]\}$. An initial step to understand the spatial mechanism for gene expression is to study their spatial patterns. Computationally, this means we want to test whether the gene expression values, $g_1, g_2, \ldots, g_n$, are correlated with spatial coordinates, $\{(x_i, y_i) : i \in [n]\}$, in some way. As mentioned in the previous chapter, RNA velocity is a powerful tool for understanding the underlying biochemical process of gene expression, namely DNA transcription, RNA splicing and degradation. With spatial transcriptomic data, we are able to further investigate how spatial locations affect the biochemical activities underlying gene expression. Thus, we come up with a new computation task: can we add a spatial mechanism to complement current RNA velocity model and jointly infer this underlying spatial information and RNA velocity? We emphasize that this computation task offers significantly more information and interpretation than merely understanding the statistical association between gene expression and spatial information. Inspired by another line of work in cell-cell communication, we start with a basic assumption for our spatial mechanism: cells communicate with each other via ligand-receptor binding to control mRNA production. Thus, the RNA velocity of each cell depends not only on its own cell state, but also its neighboring cells in a tissue.

We convert the problem of understanding the spatial mechanism of gene expression to a computational problem as follows. Each sample (cell or spot), indexed by i, is represented by a vector of gene expression values $X_i(t) \in \mathbb{R}^d$ and spatial coordinates (x, y), both parametrized by developmental stage (time) t. The trajectory $X_i(t)$ is governed by some differential equation plus random noise. For each i, only the vector $x_i := X_i(t_i)$ is observed at some unknown time t_i. Furthermore, the parameters of the differential equations are spatially coupled in an unknown way, forming a random field. Our goal is two-fold: recover the latent times t_i for each sample and predict future gene expression values and spatial positions. Our approach is based on the previous chapter about Bayesian inference of RNA velocity and existing graph learning techniques.

3.4 Methods

3.4.1 RNA Velocity in Spatial Contexts

To approach the computational problem mentioned above, we first make a number of assumptions. First, we describe the system using a differential equation that reflects the biochemical steps required for gene expression: a gene is transcribed as unspliced pre-mRNA before being converted into spliced mature mRNA (**Fig. 3.1a**). This builds on our previous work and others[89, 98, 169, 102]. That is, gene expression dynamics can be described by a set of ODEs and each cell has a low-dimensional cell state, $\boldsymbol{z}$, that controls a cell-specific transcription rate, $\boldsymbol{\rho}$. Our next critical assumption is the spatial coupling between cells. We represent the population of cells as a graph $\mathcal{G} = (V, E)$, where $V = [n]$ is a set of nodes representing cells and E contains the neighboring relations between any two cells. For each node (cell) i, we take its unspliced and spliced mRNA read counts as its node features, denoted as $\boldsymbol{o_i}$. Importantly, we assume that this graph captures spatial dependence among the mRNA counts, and ρ parameters of the cells (**Fig. 3.1b**). The transcription rate for each gene g in each cell i is a function of both the cell state of i and the cell states of the neighboring cells in the tissue, $nbr(i)$. The splicing and degradation rates are different for each gene, but shared across cells, giving the following system of ODEs for any gene indexed by g:

$$\frac{du_{i,g}}{dt} = \rho_{i,g}(\{\boldsymbol{c_j} : j \in nbr(i)\}) - \beta_g u_{i,g} \tag{3.1}$$

$$\frac{ds_{i,g}}{dt} = \beta_g u_{i,g} - \gamma_g s_{i,g} \tag{3.2}$$

Using this differential equation and graph structure, we can describe how the gene expression levels vary given rate parameters, cell state, and cell time. The ODE system has an analytical solution, so if we knew the cell state and cell time values, we could readily estimate the parameters. However, cell state and cell time are latent. Thus, we can reduce the modeling problem to a latent variable inference problem. We denote $\boldsymbol{\mathcal{Z}} = \{\boldsymbol{z_i} : i \in [n]\}$ and $\boldsymbol{\mathcal{T}} = \{t_i : i \in [n]\}$ as the set of all cell states and time. We next show how to perform Bayesian estimation of $\boldsymbol{\mathcal{Z}}$ and $\boldsymbol{\mathcal{T}}$ using autoencoding variational Bayes. Intuitively, our TopoVelo approach extends the RNA velocity framework to spatial transcriptomic data by incorporating cell position with graph neural networks (GNN).

3.4.2 Modeling Spatial Data with Graph Neural Networks

A key aspect of understanding spatial transcriptomic data is properly modeling spatial relationships. Unfortunately, our understanding of spatial interactions between cells remains

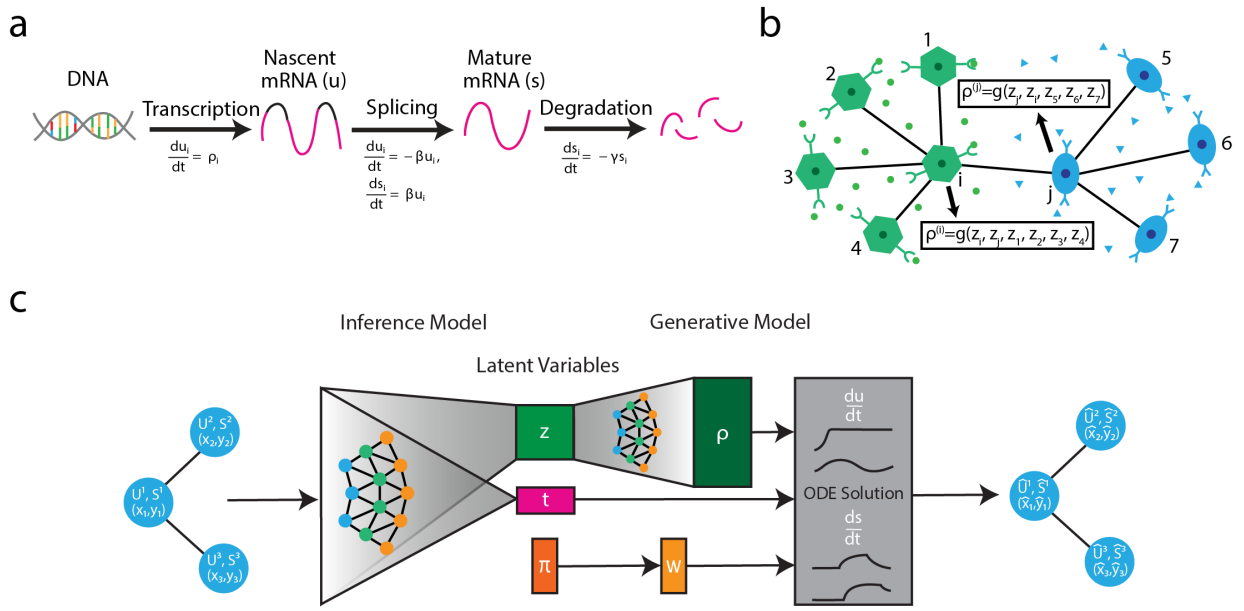

Figure 3.1: **Diagram of TopoVelo model.** **(a)** System of differential equations that describes how unspliced pre-mRNA (u) and spliced mature mRNA (s) values are related during gene expression. ρ is a transcription rate parameter that is different for every cell and every gene. β and γ are splicing and degradation rates, which are different for each gene but shared across cells. **(b)** Diagram of spatial dependence in rate parameters. TopoVelo extends previous RNA velocity models by allowing transcription rate parameters to be a function of the states of neighboring cells in a tissue, which may influence each other through mechanisms such as ligand-receptor signaling. **(c)** Diagram of TopoVelo neural network architecture. TopoVelo takes a graph as input. In this graph, nodes represent cells or spots; u, s, and spatial coordinates are node features; and edges represent spatial proximity. A graph neural network infers cell state and cell time jointly across cells (encoder network). Another graph neural network predicts transcription rates ρ from the states of each cell and its neighbors. The phase of each gene is also predicted (w). The u and s counts are reconstructed using the analytical solution to the differential equation from (a).

incomplete, and no comprehensive theory mechanistically describes these relationships. As a result, manually designing rules for modeling spatial data remains challenging. However, recent advancements in deep learning have provided powerful tools for extracting information from complex data. Graph neural networks (GNNs) have emerged as a promising approach for analyzing spatial transcriptomic data.

The have been numerous research in designing GNN architecture and among them, spectral graph convolution (GCN)[126] and graph attention (GAT)[127] are two most popular methods.

Spectral Graph Convolution Graph convolution is a form of message passing. The hidden state of a node is a weighted sum of contributions from all its neighbors. The convolution operation can be written in the following spectral form[126]:

$$\boldsymbol{H} = \tilde{\boldsymbol{D}}^{-\frac{1}{2}}\tilde{\boldsymbol{A}}\tilde{\boldsymbol{D}}^{-\frac{1}{2}}\boldsymbol{O}\boldsymbol{W} \tag{3.3}$$

where $\boldsymbol{O}$ is a cell-by-feature matrix, $\tilde{\boldsymbol{A}}$ is an adjacency matrix with self-loops, $\tilde{\boldsymbol{D}}$ is a diagonal degree matrix of $\tilde{\boldsymbol{A}}$, and $\boldsymbol{W}$ is a weight matrix learned from the data.

Graph Attention Graph attention (GAT) applies the attention mechanism[127] to nodes in a graph. The original GAT paper mentioned several benefits, including that GAT is scalable to different neighborhood sizes and can be generalized to different graph structures[127]. For each node, GAT first linearly projects input features to another space, then computes an attention score over its neighborhood and aggregates the hidden information from the neighborhood. It can be written in the following form[127]:

$$\boldsymbol{h_i} = \sum_{j \in nbr(i)} k_{ij}\boldsymbol{W}\boldsymbol{o_i} \quad \text{where } k_{ij} = \frac{\exp(\boldsymbol{a}^T[\boldsymbol{W}\boldsymbol{o_i} \| \boldsymbol{W}\boldsymbol{o_j}])}{\sum_{j' \in nbr(i)} \exp(\boldsymbol{a}^T[\boldsymbol{W}\boldsymbol{o_i} \| \boldsymbol{W}\boldsymbol{o_{j'}}])} \tag{3.4}$$

We can enable multi-head attention by using multiple projection matrices $\boldsymbol{W}$ and concatenating the outputs.

3.4.3 TopoVelo: Topological Velocity Inference from Spatial Transcriptomic Data

Probabilistic Model With the modeling assumptions and ODE formulation from **Section 3.4.1**, we build a probabilistic model for the generative process of unspliced and spliced mRNA counts. Given the time, $\boldsymbol{\mathcal{T}} = \{t_i : i \in [n]\}$, and low-dimensional latent cell representations, $\boldsymbol{\mathcal{Z}} = \{\boldsymbol{z_i} : i \in [n]\}$, of the whole cell population, the node features (gene expression values) are generated from $\boldsymbol{\mathcal{Z}}$ and $\boldsymbol{\mathcal{T}}$ via a spatially-coupled dynamical system:

$$P\left(\boldsymbol{\mathcal{O}} | \boldsymbol{\mathcal{Z}}, \boldsymbol{\mathcal{T}}\right) = \prod_{i=1}^{n} P(\boldsymbol{o_i} | \{\boldsymbol{z_j}, t_j : j \in nbr(i)\}) \tag{3.5}$$

Here $nbr(i)$ denotes the set of spatial neighbors of cell i. We further assume $P(\boldsymbol{o_i}|\{\boldsymbol{z_j}, t_j : j \in nbr(i)\})$ is a factorized Gaussian distribution over all genes, i.e., $P(\boldsymbol{o_i}|\{\boldsymbol{z_j}, t_j : j \in nbr(i)\}) = \prod_{g=1}^{G} \mathcal{N}(\mu_{u,g}^{(i)}, \sigma_{u,g}^2)\mathcal{N}(\mu_{s,g}^{(i)}, \sigma_{s,g}^2)$ where $\mathcal{N}(\cdot)$ denotes a Gaussian distribution. Here, $\mu_{u,g}^{(i)}$ and $\mu_{s,g}^{(i)}$, the mean unspliced and spliced counts of each gene, are given by the solution to Eq.

3.1-3.2. In practice, we find that it is helpful to explicitly model repression-only genes (for which ρ is identically 0), so we introduce an indicator variable k_g, which is 0 if a gene is only repressed and 1 otherwise. So now the transcription rate $\rho_i^{(k_g)}$ is also a function of k_g, and in particular $\rho_i^{(k_g=0)} = 0$ for all i. We model each gene as a Gaussian mixture of the two states $k_g = 0$ and $k_g = 1$. We use $\boldsymbol{w_g}$ to denote a one-hot encoding of k_g and assume it is categorically distributed: $\boldsymbol{w}_g = \left[w_1^{(g)}, \ldots, w_K^{(g)}\right]^T \sim \text{Categorical}(\pi_1^{(g)}, \ldots, \pi_K^{(g)})$, where K is the total number of categories and $K = 2$ in our case. We then set a Dirichlet prior, $\boldsymbol{\phi} := [(\pi_1^{(g)}, \ldots, \pi_K^{(g)})] \sim \text{Dirichlet}(\psi_1, \ldots, \psi_K)$.

Since exact inference of the latent variables $\boldsymbol{\mathcal{T}}$ and $\boldsymbol{\mathcal{Z}}$ from the probabilistic model is intractable, we again resort to variational Bayes to solve this computational problem. Different from previous graphical models[102, 105, 104], which consider data generation of all cells as independent and identically distributed, we consider data generation of all cells as coupled and jointly distributed. Our inference model uses a GNN to learn a variational posterior of latent cell time and cell state, while we use a generative model consisting of a second GNN mapping the cell state to high-dimensional gene-wise transcription rates and then, apply the analytical solutions to ODEs for RNA velocity. Following similar arguments to those used in previous autoencoding variational Bayes settings[171, 195, 175], we can derive the following evidence lower bound (ELBO) for our model:

$$\begin{aligned}\mathcal{L} = \mathbb{E}_{q(\boldsymbol{\mathcal{Z}},\boldsymbol{\mathcal{T}}|\boldsymbol{\mathcal{O}})}\mathbb{E}_{q(\boldsymbol{w})}\left[\log P(\boldsymbol{\mathcal{O}}|\boldsymbol{\mathcal{Z}},\boldsymbol{\mathcal{T}},\boldsymbol{w})\right] - KL(q(\boldsymbol{\mathcal{Z}},\boldsymbol{\mathcal{T}}|\boldsymbol{\mathcal{O}}) \parallel P(\boldsymbol{\mathcal{Z}},\boldsymbol{\mathcal{T}})) \\ + \log\left(\int \exp\left(\mathbb{E}_{q(\boldsymbol{w})}\left[P(\boldsymbol{w}|\boldsymbol{\pi})\right]\right) P(\boldsymbol{\pi})\right) - \mathbb{E}_{q(\boldsymbol{w})}[\log q(\boldsymbol{w})] \end{aligned} \tag{3.6}$$

We can then perform approximate Bayesian inference by using stochastic minibatch gradient descent to optimize the ELBO with respect to the neural network parameters and ODE parameters. We name this method TopoVelo, short for **Topo**logical **Velo**city inference.

Spatial Graph Construction TopoVelo adopts two GNNs as the encoder and decoder of a variational autoencoder (VAE). To exploit the spatial information, we adopt a spatial graph as the input graph to the GNNs. We consider three types of spatial graphs: (1) k-nearest-neighbor(KNN) graph, (2) epsilon-ball graph, and (3) Delaunay triangulation. Because the memory consumption of GNNs grows super-linearly with the number of nodes (cells in our case) for dense graphs, we investigated the neighborhood size and chose a proper neighborhood size for computational efficiency. We find that the epsilon-ball neighborhood size grow quadratically with radius in all datasets except for the 3D neural tube dataset (**Supplemental Fig. B.1**). For the 3D neural tube, we use 3D coordinates in spatial graph construction, so we expect the neighborhood size to increase cubically with respect to the

radius. Based on this result, we chose slightly different spatial graphs in our experiments on real data (**Supplemental Table B.1**).

Neural Network Architecture and Training We implement TopoVelo using an encoder-decoder neural network framework. The network takes a graph as input, with nodes representing cells or spots and node features indicating spliced and unspliced counts and spatial coordinates. Edges in the graph represent spatial proximity. The encoder neural network uses graph convolutional layers to estimate the cell state and time of each cell. The decoder network uses graph convolution to estimate the transcription rate of each gene in each cell, then uses the analytical solution to the ODE to estimate the splicing and degradation rate for each gene. The node features are then reconstructed using the ODE solution given time and ODE parameters. **Fig. 3.1c** summarizes the model architecture. Our approach is agnostic to the choice of graph convolution, and in our experiments, we compare spectral graph convolution (GCN)[126] and graph attention (GAT)[127]. Both types of graph convolution compute the representation of a node using its own and its neighbors' features.

The encoder consists of a GCN or GAT layer and a fully connected (FC) layer. It first extracts hidden features from node features $\boldsymbol{\mathcal{O}}$ and graph G. Next, the FC layer maps the hidden features to the mean and standard deviation of a variational posterior. The decoder is a combination of GNN and ODE solution. First, we use a decoder GNN, with a GCN or GAT layer plus an FC layer, to learn a mapping from latent cell states to transcription rates $\boldsymbol{\rho}$. Next, we compute u and s using the analytical solution to the ODEs (3.1)-(3.2), as reported in our previous work[102]. Optionally, we train a graph neural network after the encoder-decoder network for reconstructing u and s has converged to model spatial coordinates as a function of time.

By default, TopoVelo uses a multivariate Gaussian distribution with diagonal covariance matrix as a prior of latent cell state and time, $P(\boldsymbol{\mathcal{Z}}, \boldsymbol{\mathcal{T}})$, i.e. all dimensions of cell state and cell time of all cells are mutually independent. However, when the capture times of cells are known, we can incorporate this information as an informative prior distribution for the latent cell time t. We assume the prior time distribution is a Gaussian with the capture times, e.g. embryonic day 10.5 or 11.5, as the mean and a variance proportional to the time intervals between adjacent experiments.

We train TopoVelo using gradient descent with the ADAM optimizer[174]. Because we reconstruct the node features of the graph but not the edges, we can enable mini-batch training to reduce memory consumption greatly.

Parameter Initialization The graph neural network weights are initialized by a Xavier uniform distribution, while the weights of fully connected layers are initialized by a Xavier normal distribution[196]. ODE parameters are initialized by the steady-state rate parameter estimates, as in our previous work[102]. To properly initialize $q(\boldsymbol{w})$, we first follow scVelo and use a the dynamical model to initialize cell time. Then, we cluster genes by their co-expression and determine if they are repressed using a set of Kolmogorov-Smirnov tests.

We use the proportion of cells in the induction phase as an indicator of the gene being induced. First, we compute a gene-by-gene weighted adjacency matrix $\mathbf{A}$ where $A_{ij} = \frac{1}{2}[corr(\boldsymbol{U}_{:,i}, \boldsymbol{U}_{:,j}) + corr(\boldsymbol{S}_{:,i}, \boldsymbol{S}_{:,j})] + 1$, where $corr(\cdot, \cdot)$ denotes Pearson correlation coefficient and $\boldsymbol{U}_{:,i}, \boldsymbol{S}_{:,i}$ denotes unspliced and spliced counts of the i-th gene across all cells. We apply spectral clustering on genes using $\mathbf{A}$. The number of clusters is determined by using an empirical threshold on the singular values of $\mathbf{A}$ with a Gaussian noise assumption[197]. Let $\tilde{w}_g$ be the proportion of cells in the induction phase and $y_g \in \{1, 2, \ldots, K_y\}$ be the clustering result of the g-th gene. For each cluster y, we perform two Kolmogorov-Smirnov tests between $\mathcal{W}_y = \{w_g : y_g = y\}$ and Dirichlet$(5.0, 5.0)$. The two null hypotheses are

1. $\mathcal{W}_y$ is sampled from Dirichlet(ψ_1, ψ_2) whose CDF is less than CDF of Dirichlet$(5.0, 5.0)$
2. $\mathcal{W}_y$ is sampled from Dirichlet(ψ_1, ψ_2) whose CDF is greater than CDF of Dirichlet$(5.0, 5.0)$

If the first null hypothesis is not rejected, genes in cluster y are likely to be inductive. We set $\psi_1 > \psi_2$ and reinitialize $\mathcal{W}_y$ by sampling from Dirichlet(ψ_1, ψ_2). If the second null hypothesis is not rejected, genes in the cluster y are likely to be repressive and we perform the same sampling but with $\psi_1 < \psi_2$. If both null hypotheses are rejected, we sample $\mathcal{W}_y$ from a mixture of two Dirichlet distributions mentioned in the previous two cases. Such an initialization method prevents TopoVelo from a bad initialization.

3.4.4 Modeling Real Tissue Growth with Cell Velocity

Previous work introduced the concept of RNA velocity, defined as the time derivative of spliced mRNA, $\frac{ds}{dt}$. Since we know the spatial coordinates (x, y) of cells in our spatial transcriptomic data, we can leverage our modeling framework to quantify temporal changes in cell position. In particular, we can infer $(\frac{dx}{dt}, \frac{dy}{dt})$, the time derivative of each cell's spatial coordinates. We refer to the quantity $(\frac{dx}{dt}, \frac{dy}{dt})$ simply as "cell velocity", because it is a change of physical position over time–a velocity in the traditional sense. Here, we used the heuristic method popularized by scVelo[98] to project RNA velocity onto spatial coordinates and obtain an implicit estimate of cell velocity. In brief, the projection method starts with RNA

velocity calculation. Next, it calculates a cell-to-cell transition matrix based on the cosine similarity between the RNA velocity vectors between any two neighboring cells in the transcriptomic space. Finally, it calculates a weighted sum of spatial displacement with weights equal to the transition probability from the previous step. The weighted displacement is considered a project of RNA velocity onto the spatial domain. It reflects the direction of spatial development of cells.

It may be possible in future work to obtain an explicit parametric form for cell velocity as a function of cell state and time, such as with a graph neural network. We consider a spatial decoder as a graph neural network with a GCN or GAT layer and a final fully connected layer. The input to our spatial decoder is a spatial graph, with latent cell state, $\boldsymbol{z}$, latent cell time, t, initial time t_0 and initial location (x_0, y_0) as node features and the output is two-dimensional spatial coordinates. (x_0, y_0) represent the location of a cell at a previous time, t_0. We estimate t_0 and (x_0, y_0) by taking the average time and coordinates of nearest neighbors in the cell state space from all cells within certain radius. After training TopoVelo, we concatenate the latent time, cell state, initial time and initial location into feature vectors and use the given spatial coordinates as target outputs to train a spatial decoder.

3.4.5 Evaluation Metrics

To quantitatively evaluate model performances, we apply both previously published and newly defined evaluation metrics regarding the inferred RNA velocity, cell velocity and cell time.

Time Correlation Given true cell time $T = [t_1, \ldots, t_n]$ and inferred cell time $\hat{T} = [\hat{t}_1, \ldots, \hat{t}_n]$, we define time correlation as the Spearman correlation coefficient between T and $\hat{T}$.

Spatial Velocity Consistency Given a spatial graph and RNA velocity $V = \{\boldsymbol{v_1}, \ldots, \boldsymbol{v_n}\}$ where v_i is a vector of RNA velocity of all genes, we define the spatial velocity consistency as

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{|nbr(i)|}\sum_{j\in nbr(i)} corr(\boldsymbol{v_i}, \boldsymbol{v_j})$$

where $corr(\cdot, \cdot)$ denotes the Pearson correlation coefficient.

Spatial Time Consistency Given a spatial graph with a binary adjacency matrix, $\mathbf{A} \in \{0, 1\}^{n\times n}$, and cell time $T = \{t_1, \ldots, t_n\}$, we define the spatial time consistency as Moran's

I of time over the spatial coordinates:

$$\frac{n}{|\mathbf{A}|_1} \cdot \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} A_{ij}(t_i - \bar{t})(t_j - \bar{t})}{\sum_{i=1}^{n}(t_i - \bar{t})^2}$$

where $\bar{t}$ is the mean cell time.

Cross Boundary Direction Correctness in a Spatial Context In chapter 2, we introduced CBDir[99] and k-step CBDir (k-CBDir) metric to evaluate correctness of cell transition to descendant cell types. While the original CBDir and k-CBDir assume KNN built from gene expression similarity, they are not able to measure the performance of cell velocity prediction. Hence, we adopt the same idea and replace the original KNN graph with a spatial KNN graph to evaluate the correctness of cell velocity in a physical space. We repeat the definition of CBDir here:

$$\text{CBDir} := \frac{1}{|\mathcal{T}|}\sum_{(A,B)\in\mathcal{T}}\frac{1}{|C_A|}\sum_{i\in C_A}\frac{1}{|nbr(c)\cap C_B|}\sum_{c'\in nbr(c)\cap C_B}\frac{\boldsymbol{v_c}^T(\boldsymbol{x_{c'}}-\boldsymbol{x_c})}{\|\boldsymbol{v_c}\|\|\boldsymbol{x_{c'}}-\boldsymbol{x_c}\|}$$

where $nbr(c)$ denotes all spatial neighbors of cell c. For k-CBDir, we also repeat its definition here:

$$k\text{-CBDir} := \frac{1}{|\mathcal{T}|}\sum_{(A,B)\in\mathcal{T}}\frac{1}{|C_A|}\sum_{i\in C_A}\frac{1}{|nbr_k(c)\cap C_B|}\sum_{c'\in nbr_k(c)\cap C_B}(-1)^b\frac{|\boldsymbol{v_c}^T(\boldsymbol{x_{c'}}-\boldsymbol{x_c})|}{\|\boldsymbol{v_c}\|\|\boldsymbol{x_{c'}}-\boldsymbol{x_c}\|}$$

where b indicates whether RNA/cell velocity follows the temporal order, as defined in **Eq. 2.14**.

3.4.6 Interpreting Graph Attention via Cell Influence Score

Graph attention (GAT) applies the attention mechanism to graph-structured data. In the case of TopoVelo, the input is a cell adjacency graph with unspliced and spliced mRNA counts of all genes as node features. We denote $\boldsymbol{o_i}$ as the node feature of cell i. For any cell i in the graph, GAT first computes a set of attention scores between it and all its spatial neighbors, $nbr(i)$, using the following formula:

$$k_{ij} = \frac{\exp\left(\sigma\left(\boldsymbol{a_s}^T\boldsymbol{W_s}\boldsymbol{o_i} + \boldsymbol{a_t}^T\boldsymbol{W_t}\boldsymbol{o_j}\right)\right)}{\sum_{j'\in nbr(i)}\exp\left(\sigma\left(\boldsymbol{a_s}^T\boldsymbol{W_s}\boldsymbol{o_i} + \boldsymbol{a_t}^T\boldsymbol{W_t}\boldsymbol{o_{j'}}\right)\right)}, \forall j \in nbr(i) \tag{3.7}$$

Here, $\sigma(\cdot)$ is a nonlinear activation function, e.g. ReLU; $\boldsymbol{a_s}, \boldsymbol{a_t} \in \mathbb{R}^d$, $\boldsymbol{W_s} \in \mathbb{R}^{m\times d}$ and $\boldsymbol{W_t} \in \mathbb{R}^{m\times d}$ are trainable parameters. For each cell, the total attention score received from

its direct neighbors always sums up to 1, i.e. $\sum_{j \in nbr(i)} k_{ij} = 1$, while the total attention score sent by each cell can be different. Next, node features are projected onto a different vector space using the attention scores and a projection matrix $\boldsymbol{W}$:

$$\boldsymbol{h_i} = \sum_{j \in nbr(i)} k_{ij} \boldsymbol{W} \boldsymbol{o_i} \tag{3.8}$$

Here, $\boldsymbol{h_i} \in \mathbb{R}^m$ denotes the hidden representation of cell i. We can enable multi-head attention by using multiple projection matrices $\boldsymbol{W^{(1)}}, \ldots, \boldsymbol{W^{(m)}}$ and concatenating the output vectors $\boldsymbol{h_i} = [\boldsymbol{h_i^{(1)}}, \ldots, \boldsymbol{h_i^{(m)}}]^T$, where each $\boldsymbol{h_i^{(l)}}$ is computed using separate parameter matrices and **Eq. 3.7, 3.8**.

We can see that k_{ij} can be interpreted as the message passed from a neighboring cell to a target cell. It remains an interesting question whether the graph attention score reflects some aspect of real cell-cell communication. To investigate this, we extend the original attention score computation from direct neighboring pairs to all pairs of cells and calculate the cell influence score from each cell as follows:

$$att_j = \sum_{i=1}^{n} k_{ij} \tag{3.9}$$

In reality, we often use multiple attention heads. In this case, we take the mean across all heads as the final cell influence score. As we will show in the experiment section, the proposed cell influence score generates new insights into cell-cell interaction during tissue development.

3.4.7 Interpreting Results from 3D Neural Tube Development

While 3D spatial transcriptomic technology is developing, most existing spatial transcriptomic technologies only produce 2D slices of tissue sections. Since cells are differentiating in a 3D space, we cannot get a comprehensive view with only a 2D slice. An alternative way to study 3D organoid is to have spatially consecutive sections and sequence these slices using spatial transcriptomics. These slices form a pseudo-3D spatial transcriptomic data. As we will show in **Section 3.5**, we successfully applied TopoVelo to this 3D neural tube dataset. To better interpret cell velocity from TopoVelo, we devised a set of auxiliary computational methods to study 3D neural tube development.

The neural tube has a tubular shape and undergoes bending and rolling when during embryonic development (**Fig. 3.2a**). Its anatomical structure can be described by three axes: (1) anterior-posteror (A-P) axis, (2) dorsal-ventral (D-V) and (3) left-right (L-R) axis.

These three axes are biologically meaningful, but are not orthogonal in the physical space. To better interpret cell velocity, we need a computational method to project cell velocity onto these three axes. We came up with a way to unroll the neural tube first so that the three axes are orthgonal. Next, we calculated projected cell velocity on this transformed coordinate system. Next, we describe our method details.

To obtain a 3D view of the neural tube with A-P, D-V and M-L as orthogonal axes, we unrolled the neural tube by first finding its principal curve[198], a 1D curve passing through 3D data points. We used scFates[199] to calculate 30 anchor points, $\boldsymbol{a_1}, \ldots, \boldsymbol{a_{30}}$, sequentially along the A-P axis. The tool also gives an assignment of each cell to an anchor point. Next, we "unrolled" the neural tube by approximating the principal curve with a spline of line segments, $[(\boldsymbol{a_0}, \boldsymbol{a_1}), (\boldsymbol{a_1}, \boldsymbol{a_2}), \ldots, (\boldsymbol{a_{29}}, \boldsymbol{a_{30}})]$, between anchor points. Here, $\boldsymbol{a_0} = [0, 0, 0]^T$ is the origin. The cell progression in the A-P direction was also approximated by each line segment. The unrolling process was performed recursively. The base step is trivial - we obtained the first transformed anchor point $\boldsymbol{a'_0} = \boldsymbol{a_0}$. For each following segment $(\boldsymbol{a_i}, \boldsymbol{a_{i+1}}), r \geq 1$, we already have previously transformed principal curve $(\boldsymbol{a'_0}, \ldots, \boldsymbol{a'_i})$ and we can recursively transform each segment of the neural tube using the following steps:

1. Consider a coordinate system $\mathcal{C}$ with $\boldsymbol{a_i}$ as the origin and the same x, y and z directions as in the original one. Compute the coordinates of $\boldsymbol{a_{i+1}}$ in $\mathcal{C}$ and transform it to spherical coordinates. Let (r_i, ϕ_i, θ_i) be the spherical coordinates. The spherical transformation is:

$$r = \sqrt{x^2 + y^2 + z^2} \tag{3.10}$$

$$\phi = \arccos\left(\frac{z}{r}\right) \tag{3.11}$$

$$\theta = \arccos\left(\frac{x}{\sqrt{x^2 + y^2}}\right) \cdot \text{sign}(y) \tag{3.12}$$

   where sign$(\cdot)$ is $+1$ for non-negative values and -1 for negative values. Obtain the transformed anchor point $\boldsymbol{a'_{i+1}} = \boldsymbol{a'_i} + [0,\ \|\boldsymbol{a_{i+1}} - \boldsymbol{a_i}\|,\ 0]^T$ and add it to the transformed principal curve.

2. For all cells assigned to anchor $\boldsymbol{a_i}$, compute the coordinates in $\mathcal{C}$ and transform them to spherical coordinates using Eq.3.10-3.12. For a cell j, denote $(\Delta r_j, \Delta\phi_j, \Delta\theta_j)$ as the spherical coordinates.

3. Since ϕ_i and θ_i are also the rotation angles to align the direction of $(\boldsymbol{a_i}, \boldsymbol{a_{i+1}})$ with $(\boldsymbol{a'_{i-1}}, \boldsymbol{a'_i})$, the spherical coordinate of any cell j with respect to $\boldsymbol{a'_i}$ is $(\Delta r, \Delta\phi, \Delta\theta) =$

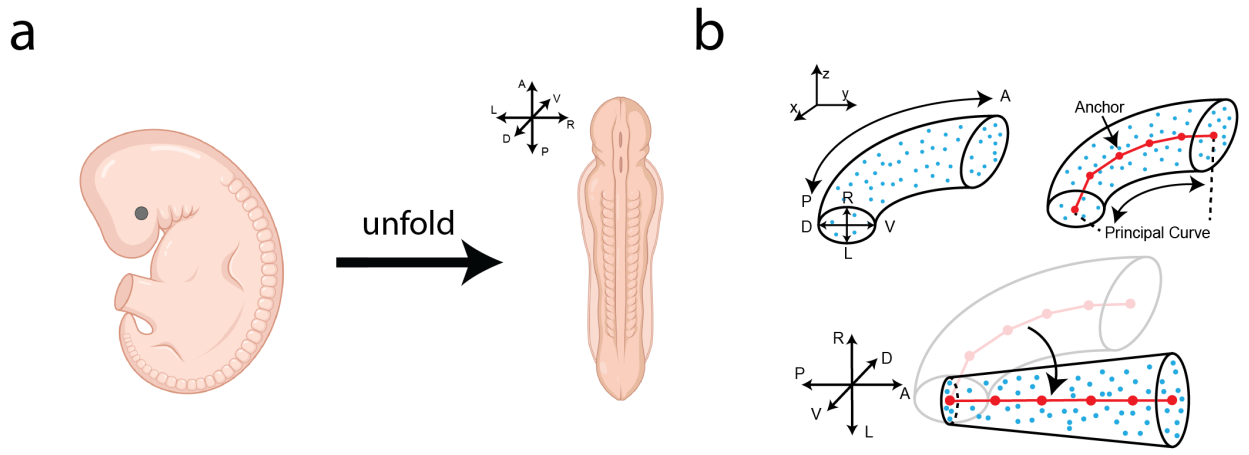

Figure 3.2: **Illustration of Neural Tube Unfolding. (a)** Animation of a mouse embryo with a folded neural tube (left) and an unfolded neural tube (right). **(b)** Proposed algorithm to unfold a neural tube.

$(\Delta r_j, \frac{\pi}{2} - \phi_i + \Delta\phi_j, \frac{\pi}{2} - \theta_i + \Delta\theta_j)$. We can transform it back to cartesian coordinates using:

$$\Delta x = \Delta r \sin(\Delta\phi)\cos(\Delta\theta)$$
$$\Delta y = \Delta r \sin(\Delta\phi)\sin(\Delta\theta)$$
$$\Delta z = \Delta r \cos(\Delta\phi)$$

4. Notice that $(\Delta x, \Delta y, \Delta z)$ is just the coordinates relative to $\boldsymbol{a_i}$, but not the transformed anchor point, $\boldsymbol{a_i'}$. We need to translate all cells assigned to anchor $\boldsymbol{a_i}$ by $\boldsymbol{a_i} - \boldsymbol{a_i'}$ to get final results:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \boldsymbol{a_i} - \boldsymbol{a_i'} + \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix}$$

Fig. 3.2b illustrates the unfolding process.

After transformation, each cell has a 3D coordinate in the unrolled neural tube. Next, we project RNA velocity onto the unrolled 3D space and perform a divergence analysis. In mathematics and physics, divergence is used to describe the net flow of a vector field out of a point in space. In a 3D space, given a vector field $\overrightarrow{F} = (v_x, v_y, v_z)$, the divergence of $\overrightarrow{F}$, denoted as $\nabla \cdot \overrightarrow{F} = \frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z}$. We consider a 1D divergence along the anterior-posterior

axis of the neural tube:

$$div(V_{AP}) = \frac{\partial V_{AP}}{\partial x_{AP}}$$

where V_{AP} is the cell velocity in the direction of A-P axis and x_{AP} is the unrolled spatial coordinate on the unrolled neural tube. Divergence represents the total flux out of a point in space. According to the literature, cells at closure points migrate outward, so we expect a positive divergence at these points, whereas divergence should be negative where cells from two closure points converge. Since we do not have an analytical expression for cell velocity, we apply an algorithm that approximates divergence using a sliding window method. The algorithm consists of the following steps:

1. Sort cells based on their A-P coordinates.
2. Discretize the A-P axis into windows of length w and step size of w.
3. Calculate the average cell velocity in each window as the approximate cell velocity of the grid point. Denote $\bar{v}_i$ as the cell velocity of the i-th grid point, $i = 0, 1, \ldots, n-1$.
4. For the i-th grid point ($i > 1$), its divergence is approximated as $\bar{v}_i - \bar{v}_{i-1}$.
5. Apply a 1D Gaussian filter with bandwidth=$0.25w$ to filter out local noise.

3.4.8 Data Preprocessing

We followed scVelo preprocessing pipeline in general with certain modifications depending on different dataset attributes. The standard preprocessing for RNA velocity has the following steps: (1) preliminary gene filtering by total unspliced and spliced RNA count number, (2) cellwise normalization, (3) variable gene selection and (4) smoothing of count numbers on a KNN graph. We performed the standard preprocessing pipeline for the Slide-seq brain cortex and all stereo-seq datasets. In addition, we selected spatially variable genes in the Slide-tags mouse brain and the 3D neural tube datasets. We report the details in **Supplemental Table B.2**. My colleague Dr. Justin Kun Lee performed cell culturing and Dr. Jialin Liu performed image registration and visualization. The following paragraphs of section 3.4.8 describes their methodology for completeness of this dissertation and full credit is given to Dr. Justin Kun Lee and Dr. Jialin Liu.

iPSC Maintenance iPS cells (WTC11) were maintained in feeder-free conditions in mTeSR1 (STEMCELL, 85850) on plates coated with Matrigel Growth Factor Reduced Basement Membrane Matrix (Corning, 354230). The cells were passaged every 3-4 days using Accutase (Gibco, A1110501) and seeded with ROCK inhibitor Y-27632 (Stemcell, NC1678100).

Embryoid Body Differentiation and Maintenance For embryoid body (EB) differentiation, the EB formation media (DMEM/F-12 (Gibco, 11320033), 20% (v/v) Knockout Serum Replacement (Gibco, 10828028), 1% MEM Non-essential amino acid solution (Gibco, 11140050), 0.5% GlutaMAX (Gibco, 35050061) supplemented with β-mercaptoethanol (MP Biomedicals, 194705) and 1% penicillin-streptomycin (Gibco, 15140122)) was used following the STEMCELL Aggrewell 400 protocol. The AggreWell 400 24-well plate (STEMCELL, 34415) was coated with anti-adherence rinsing solution (STEMCELL, 07010). The iPSCs were dissociated with accutase and seeded 1.2 x 10^6cells per well in EB formation media. After 24 hours, 50% of the media was replaced with fresh EB formation media. After 48 hours of seeding, the EBs were harvested using 37-um reversible strainers and moved to an ultra-low attachment 6-well plate (Corning Costar, 07-200-601) in Essential 6 (E6) media (Gibco, A1516401). The media change was done every 48 hours. After 21 days of seeding, the EBs were flash-frozen in E6 media and kept in liquid nitrogen until further analysis.

Spatial Transcriptomics To perform Slide-Seq, EBs were thawed on ice, and O.C.T. compound (Fisher Scientific, 4585) was gently added over the EBs, filling the embedding mold (1 cm diameter) before snap freezing in dry ice. 10-micron EB sections were placed on 3mm CurioSeeker tiles for processing. EBs were permeabilized, released mRNA was captured on the spatially barcoded beads, and cDNA synthesis was performed per the manufacturer's protocol. cDNA was then amplified and used for library preparation following the CurioSeeker protocol.

Final library quality was assessed using the LabChip GXII HT (PerkinElmer) and libraries were quantified by Qubit (ThermoFisher). Pooled libraries were used for paired-end sequencing according to the manufacturer's protocol (Illumina NovaSeqX Plus). BclConvert Software (Illumina) was used to generate de-multiplexed Fastq files and the CurioSeeker Pipeline (Curio Biosciences) was used to align reads, generate feature-spot matrices, perform clustering and gene expression analysis, and place spots in spatial context on the slide image.

EB Cluster Annotation We performed deconvolution analysis using Robust Cell Type Decomposition (RCTD) to annotate the EB clusters[200]. For the reference data, we used scRNA-seq data on D21 EBs, and the cell types were deconvoluted based on the Seurat cluster resolution 0.1 [201]. An RCTD object was created using 200 minimum UMI per bead, and default parameters were used for the rest of the analysis.To identify which spots belong to each EB, we performed hierarchical clustering using the function *hclust* with the *ward.D2* method in *R*. The number of groups was set to 9. These hierarchical clusters were

used when identifying the cell type proportion of each EB in **Supplementary Fig. B.2**.

H&E Image Registration For registering Curio-seq beads to the H&E image, we hand-picked a set of anchor points based on matching shapes, performed a piece-wise linear transformation, and aligned the image to the bead locations in pixels using the *fitgeotform2d* function in MATLAB. Since bead locations are simply the centroids of each bead, we cropped out square image patches with a length of 10 µm around these centroids to represent the morphology overlapping each bead location. We then adopted the *ggimg* package[202] to visualize these images patches in a 2D plot.

3.5 Experiments

3.5.1 Simulation Study

We investigated the performance of TopoVelo using both simulated and real data. To simulate cells from an entire tissue, we (1) sampled random starting positions for an initial set of cells at $t = 0$, (2) simulated their future locations using a defined tissue growth pattern, (3) defined a spatial correlation pattern for the transcription rates ρ, (4) calculated u and s using the analytical ODE solution and (5) added noise and simulated sparsity to match the properties of the real spatial transcriptomic data (**Supplemental Section B.2**). We simulated two datasets with simple types of tissue growth patterns, layered growth and radial growth (**Fig. 3.3a, Supplemental Table B.3**). These mimic two different types of biologically plausible patterns–the brain cortex grows layer by layer, while other structures grow from the inside to the outside.

To quantitatively evaluate our method, we computed several metrics and compared our performance with those from previous RNA velocity methods. We compared with (1) scVelo[98], a widely used RNA velocity approach that fits each gene separately and assumes a single lineage, (2) VeloVAE[102], our previous method that fits a single shared latent time value for each cell and can model branching trajectories, (3) cellDancer[101], a deep learning model for inferring cell-specific RNA velocity, (4) DeepVelo[100], a deep learning model leveraging graph neural networks yet not utilizing spatial information, (5) VeloVI[104], a VAE model fitting each gene separately and (6) STT[123], an RNA velocity method incorporating spatial information. Among these methods, only STT takes spatial information into account. Unlike TopoVelo, which directly models spatial interactions inside the RNA velocity model, STT assumes cell development stems from two kernels calculated from velocity and spatial coordinates separately.

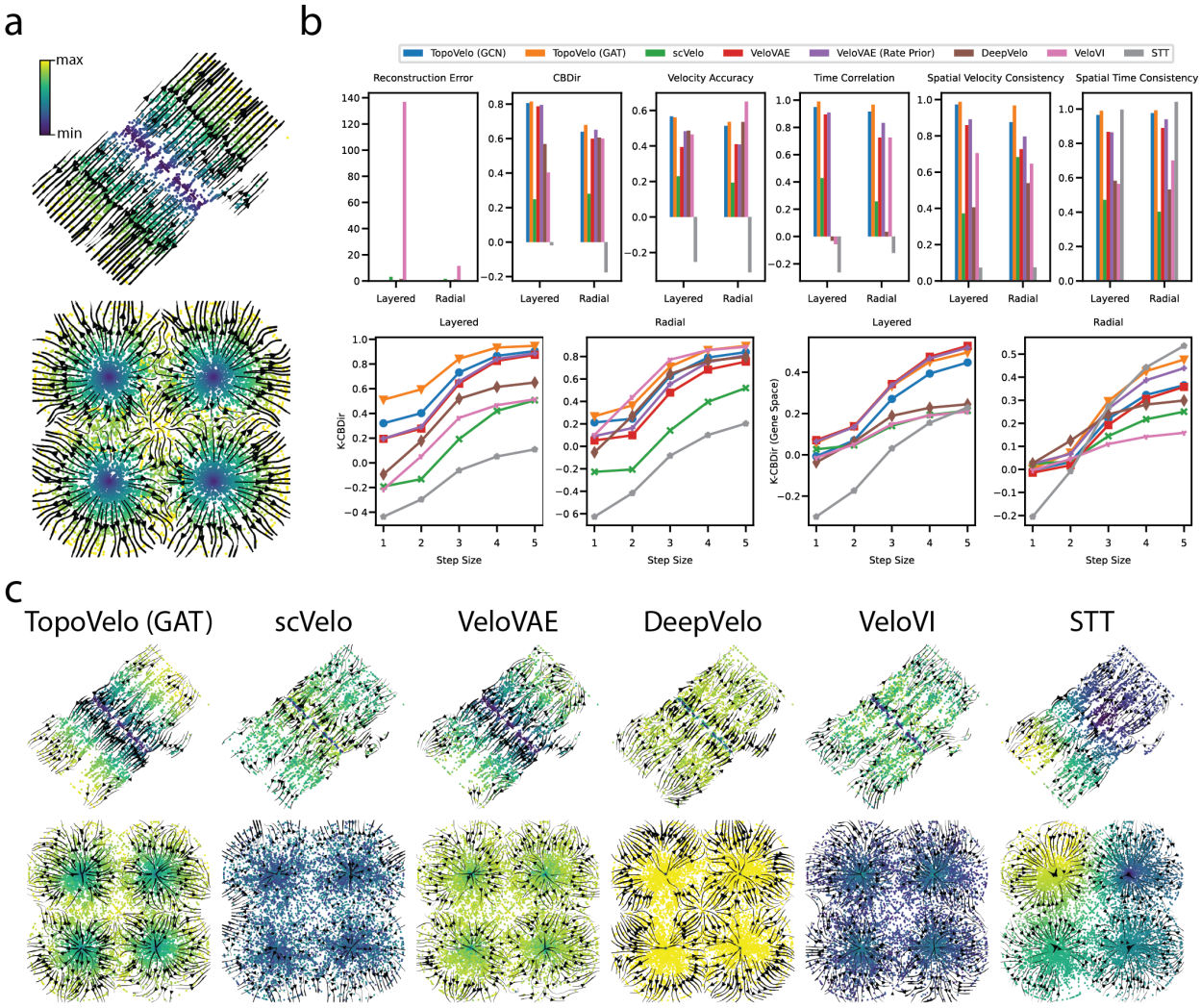

Figure 3.3: **TopoVelo Accurately Models Spatial Dynamics in Simulated Datasets.** **(a)** Simulated datasets with bidirectional(top) and radial(bottom) growth patterns. The true velocity stream is plotted over spatial coordinates colored by the true time values. Note that these stream plots indicate cell velocity in physical coordinates, not RNA velocity in abstract UMAP coordinates. **(b)** Quantitative performance on Simulated Datasets. The top row shows six metrics used for benchmarking. Higher values are better for all metrics except for reconstruction error. The bottom row contains line plots showing k-CBDir using k-step neighbors with k up to five. Note that we computed two versions of k-CBDir, one using spatial nearest neighbors and one using nearest neighbors in gene expression space. **(c)** Comparison of inferred cell velocity. Streamlines with arrow indicate the direction of cell velocity in the physical space. Coordinates are colored by inferred time values.

Evaluation metrics include cross-boundary direction correctness (CBDir)[99], Spearman correlation of cell time, spatial velocity consistency and spatial time consistency. For evaluating cell velocities, we first employed cross-boundary direction correctness (CBDir)[99], a

metric that measures the coherence of velocity flow between progenitor and descendant cell types. We also introduced a new metric called k-CBDir by extending CBDir to include not only immediate neighbors but also k-step neighbors with k up to five. Additionally, for simulated datasets, we computed the average cosine similarity between inferred spatial velocity and the true cell velocity, which we refer to as "velocity accuracy" (**Fig. 3.3b**). Moreover, we calculated spatial time consistency and spatial velocity consistency as defined in **Section 3.4.5**. These two spatial consistency metrics assess whether cell time and velocity vary smoothly among spatially adjacent cells instead of fluctuating rapidly over space. We think these are reasonable evaluation metrics, because methods that neglect spatial coordinates often produce spatially chaotic time and velocity. Thus, lack of spatial smoothness is a clear drawback of these methods. However, it is important to clarify that we neither expect nor require that the true velocity and time patterns are perfectly smooth over space. TopoVelo fits a separate set of cell state and rate parameters for each cell, and cells that are immediate neighbors can have sharply varying parameters. This allows TopoVelo to model tissues with multiple compartments that interact at sharp boundaries. For example, in the two simulated tissue topologies we evaluated, there are sharp boundaries in spatial velocity, and TopoVelo accurately infers these.

Our results show that TopoVelo achieves significantly better performance in terms of CBDir, velocity accuracy, time correlation, spatial velocity consistency, and spatial time consistency (**Fig. 3.3b**). Among all six methods compared, TopoVelo has the strongest and most consistent velocity flow. The time inference from TopoVelo is also the closest to the ground truth. VeloVAE, DeepVelo and VeloVI all showed less accuracy in terms of time inference and spatial time smoothness compared to TopoVelo. Finally, we used the simulated data to assess the accuracy of rate parameter estimation (**Supplemental Fig. B.3**). TopoVelo and VeloVAE achieved top performance in estimating spatially-varying, cell-specific transcription rates. Surprisingly, scVelo performed the best at estimating splicing rates even though it showed significantly worse performance than other methods in most other metrics (time estimation, velocity estimation, etc.). VeloVI also performed consistently well at estimating rate parameters. This indicates that incorporating spatial coordinates can substantially improve the estimation of cell times and velocities. We next visualized cell velocity using velocity stream plots on spatial coordinates colored by inferred latent cell time (**Fig. 3.3c**). In addition, TopoVelo has the strongest and most consistent velocity flow. The time inference from TopoVelo is also the closest to the ground truth, while the time from VeloVAE is mostly accurate yet more chaotic spatially.

3.5.2 Benchmark Study on Real Data

As a real validation, we applied TopoVelo to real spatial transcriptomic datasets generated with the Slide-Seq v2, Stereo-Seq, and 10x Visium technologies. Slide-Seq measures transcriptome-wide mRNA expression levels at 10 μm resolution, while Stereo-Seq measures transcriptome-wide mRNA expression at sub-μm resolution. These technologies are ideal initial candidates for spatial velocity analysis because they have cellular resolution and capture both spliced and unspliced transcripts. We further chose a mouse cerebral cortex Slide-seq dataset from embryonic day 15 (E15) and mouse gut and lung Stereo-seq datasets from E13.5. The Slide-Seq and Stereo-seq datasets were not annotated with ground truth for the cell developmental stage, so we relied prior knowledge from existing literature to verify our results. In addition, we found a recently-published human thymus 10X Visium dataset that does have ground-truth labels. Each spatial transcriptomic spot in the thymus dataset is annotated with a scalar indicating its position along the cortex-medulla axis (CMA)[1]. The CMA value was calculated from a number of manually curated points in a histology image and corresponds to the direction in which cells migrate and differentiate in the thymus.

These real datasets exhibit spatial developmental patterns consistent with those designed in our simulation study. During the development of the mouse cerebral cortex, neurons integrate into cortical layers in a birthdate-dependent manner and follow a radial axis starting in the ventricular zone (VZ) and progressing through the subventricular zone (SVZ), intermediate zone (IZ), and cortical plate (CP)[203]. Embryonic mouse lung and gut form 3D tubular structures during development. During early embryonic development (E9.5 to E14.5), the developing gut and lung initially form pseudostratified epithelial tissue surrounded by splanchnic mesenchyme. From E9.5 to E14.5, the cells of the epithelium and mesenchyme rapidly proliferate, resulting in gut elongation and increase in the luminal area and girth of each tube[204, 205]. Finally, the thymus consists of the medulla and cortex regions, forming individual compartments[1, 206, 207]. Precursor cells called thymocytes enter the medulla region of the thymus through blood vessels and migrate from medulla into the cortex region as they differentiate into $CD4^+$ and $CD8^+$ cells[1, 206].

We downloaded the authors' Slide-Seq u and s counts and ran the published SAW pipeline to obtain u and s from Stereo-Seq data. In the absence of cell segmentation information, we binned the Stereo-Seq data at 20 μm resolution. Additionally, the Stereo-Seq data contained cluster annotations only at the level of tissues (e.g., gut and lung), so we sub-clustered gut and lung spots using Leiden community detection and manually annotated clusters as epithelial or mesenchymal using previously published marker genes. For the Visium thymus dataset, we ran kallisto and bustools[56] on the original fastq files to obtain u and s counts. We followed the same preprocessing steps used by previous RNA velocity methods: selecting

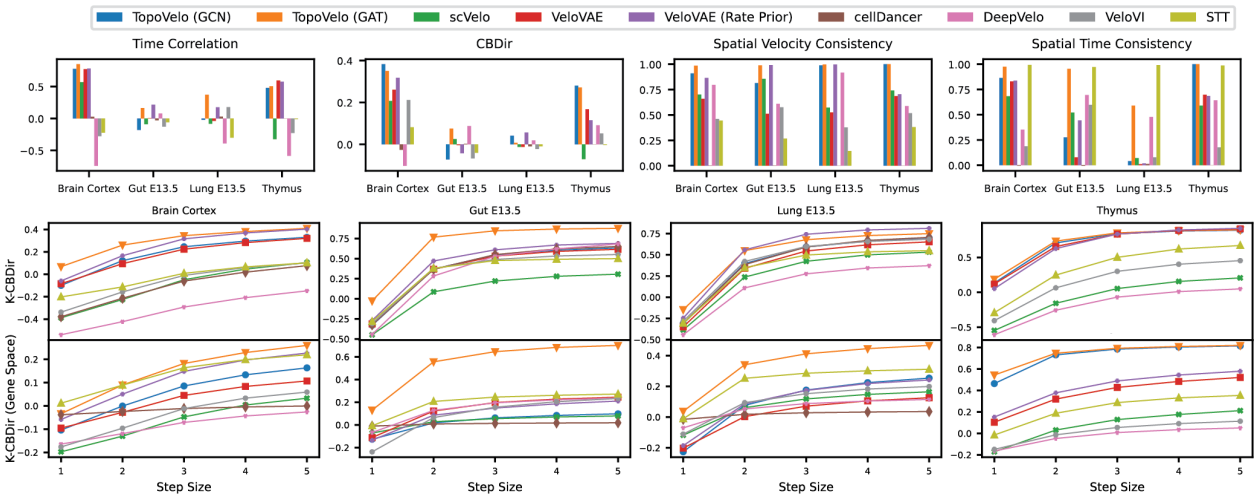

Figure 3.4: **Quantitative Comparison of Model Performance on Real Datasets. Top**: bar plots comparing methods across four benchmark datasets. Each color represents a method and each bar group represents a dataset. **Bottom**: line plot showing k-Step CBDir metric. Each color represents a method.

highly variable genes, normalizing by library size, and smoothing u and s among nearest neighbors[98].

Although we lack detailed ground truth labels in the real Slide-seq and Stereo-seq data, we know the general spatial trend of tissue growth, and we know that inferred time and velocity should generally be spatially smooth. Additionally, we know that the cortex forms from the VZ to the upper layers (right to left in the Slide-Seq section) and that gut and lung tubes develop from the inside out. Thus, for the cortex dataset we used the x coordinate as a rough approximation of the expected temporal order. For the gut and lung datasets, we roughly approximated the expected temporal order by identifying the tissue border using Delaunay triangulation, then calculating the distance of each cell from the nearest border cell. For the thymus dataset, the CMA values provide ground truth about cell developmental time; the cells with low CMA values should be the least differentiated, because cell migration and differentiation from the medulla to the cortex is well supported by literature[1, 206, 207].

As shown in **Fig. 3.4**, TopoVelo with graph attention achieves the highest spatial time consistency, CBDir, spatial velocity consistency, and time correlation. We see a particularly significant advantage in terms of the spatial time consistency, where only TopoVelo and STT achieved high performance. This demonstrates the necessity of including spatial information. Meanwhile, we observe STT achieved a much lower velocity consistency while TopoVelo consistently performed the best across all four datasets. Unlike STT, which models RNA velocity and spatial relations separately, TopoVelo directly incorporates spatial dependency

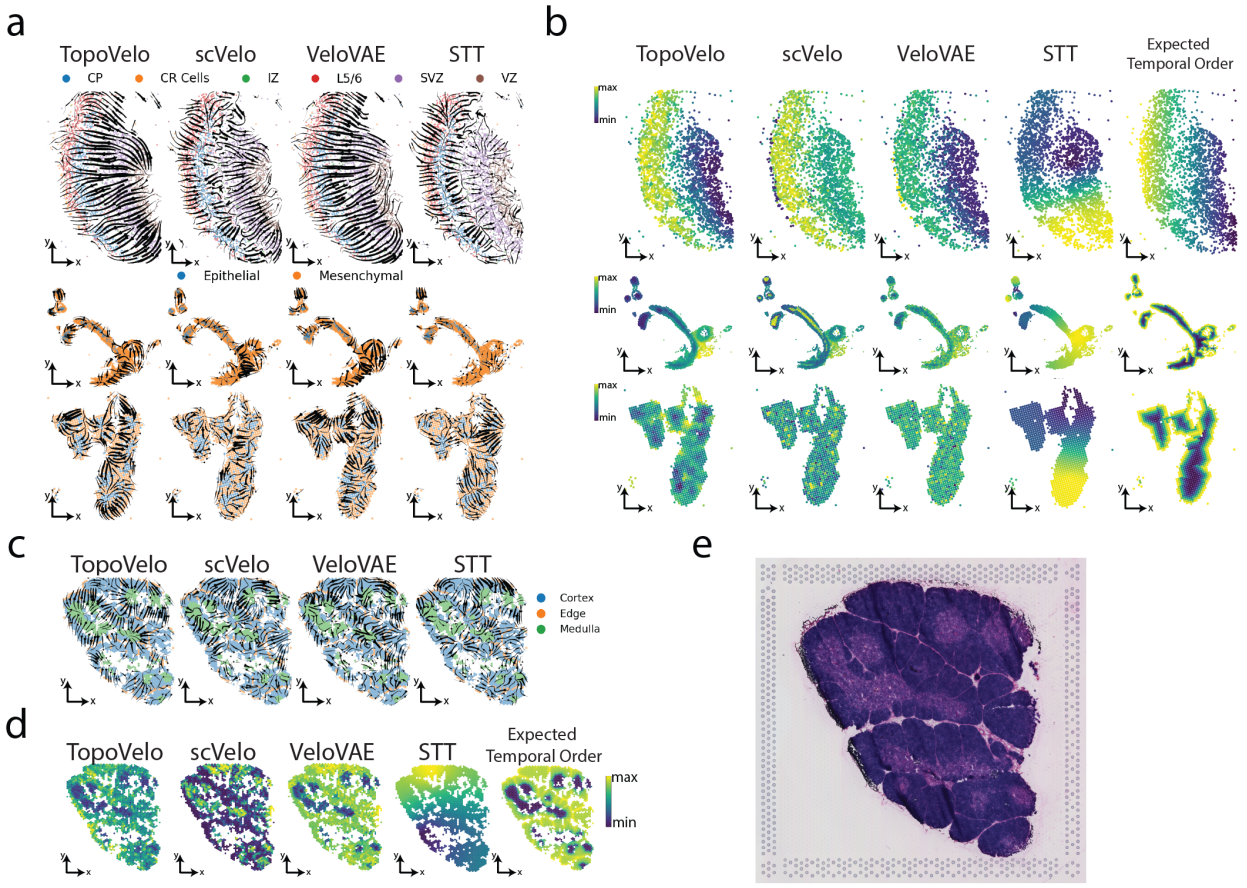

Figure 3.5: **Qualitative Comparison of TopoVelo and Existing RNA Velocity Methods. (a)** Velocity stream plots shown on spatial (physical) coordinates. **(b)** Cell coordinates colored by inferred latent cell time. The last column entitled "Expected Temporal Order" shows a rough approximation of the expected time based on prior knowledge that cortex grows from ventricular zone to layers of excitatory neurons and gut and lung form from the inside out. **(c)** Velocity stream plots on the thymus tissue. **(d)** Thymus cell coordinates colored by inferred latent cell time. The last column entitled "Expected Temporal Order" is the ground truth by the CMA score. **(e)** Morphology image of the thymus[1].

in the RNA splicing dynamics. For k-CBDir, TopoVelo also achieved consistently better performance than the other methods at all time steps.

We also qualitatively evaluated the results by visualizing velocity and latent time on tissue coordinates (**Fig. 3.5a, b, c, d**, **Supplemental Fig. B.4, B.5**). In the cerebral cortex, both TopoVelo and VeloVAE correctly captured the cell migration trend from the VZ zone to CP in the stream plots. However, TopoVelo produced more accurate cell time in the more differentiated layers including L5/6 and CP (**Fig. 3.5a, b**, first row). In contrast, scVelo not only showed a reversed migration direction in L5/6, CP and CR cells in the streamline plots, but also misidentified CR cells as the cell type of origin. In both streamline and latent time

plots from the gut and lung data, TopoVelo exhibited a clear spatial pattern of outward radial growth. Although the developmental relation between mesenchymal and epithelial cells in early mouse gut and lung is not completely understood, previous literature concludes that from E9.5 to E14.5, pseudostratified intestinal epithelium and surrounding splanchnic mesenchyme rapidly proliferate, resulting in tube elongation and increase in both luminal area and girth[204, 205]. This is consistent with our spatial velocity results, which predict that the tubes composed of epithelial cells (seen in cross-section in our 2D datasets) and surrounded by mesenchymal cells are growing outward. For the thymus dataset, we adopted the negative CMA score as the ground truth for cell time. TopoVelo correctly inferred the temporal order and cell migration from medulla to the cortex regions. The histological differences also corroborates our findings (**Fig. 3.5e**).

To test robustness of our method, we conducted three additional experiments. First, we verified that TopoVelo works on low-resolution spatial data (such as 10X Visium) with a second simulation study. We mimicked the effect of cell mixing by first simulating spatial unspliced and spliced RNA data as we did in the simulation study (**Section 3.5.1**). Next, we summed up unspliced and spliced counts of spatially adjacent cells to simulate a low-resolution dataset. We trained TopoVelo separately on two low-resolution datasets with layered and radial growth patterns. TopoVelo still correctly inferred the latent time and cell velocity (**Supplemental Fig. B.6**). This simulation result also aligns with our analysis of 10X Visium data from human thymus, supporting the ability of TopoVelo to model low-resolution spatial transcriptomic data.

Next, we tested the sensitivity of our method to neighborhood size on four real datasets reported in **Fig. 3.5**: E15 mouse brain cortex, E13.5 mouse intestine and lung and three-month post-natal human thymus. We built KNN and ball graphs on spatial coordinates with different parameters and trained TopoVelo using these graphs. After training, we evaluated the results both quantitatively and qualitatively. TopoVelo has generally stable performance on all datasets in terms of spatial velocity, time consistency, and K-CBDir (**Supplemental Fig. B.7, B.8**). However, the Stereo-seq E13.5 mouse gut dataset showed the most narrow range of robust results. We think this might be caused by the geometry of the gut tissue, which is narrow in the lateral dimension but much longer in the (orthogonal) axial direction, making it more sensitive to the size of the graph neighborhood. In addition, qualitative results of cell velocity and inferred cell time also show that TopoVelo was able to infer biologically meaningful results under different graph choices (**Supplemental Fig. B.9-B.12**).

We additionally applied TopoVelo to a developing mouse embryo at E13.5 profiled with the 10X Visium platform. A total of ten slices of the embryo were taken serially approximately

every 1 mm along the craniocaudal axis[208]. We showed a total of five representative slices that span most of the range of the embryo (**Supplemental Fig. B.13, B.14**).

3.5.3 TopoVelo Models Embryonic Mouse Brain Development from Slide-Tags Data

We next applied TopoVelo to Slide-Tags data from the developing mouse brain[110]. Slide-tags marks nuclei with barcodes from known spatial positions, then extracts the nuclei for droplet-based single-nucleus sequencing. Slide-tags produces spatial data at true single-cell resolution, because each transcript can be linked to a single nucleus of known position. Additionally, because the protocol relies on droplet-based single-nucleus sequencing, it detects transcripts with high sensitivity compared to other methods that capture transcripts *in situ*.

We downloaded a Slide-tags dataset from embryonic (E14.5) mouse brain and calculated unspliced and spliced transcript counts. Because Slide-tags data has the same format as 10X scRNA, we simply used the standard *velocyto* approach to obtain unspliced and spliced transcript counts. We built a spatial graph by connecting all cells within a fixed radius of 50 µm, rather than a k-nearest neighbor graph, due to variable cell density across spatial positions. We then used TopoVelo to estimate rate parameters for the cells in the dataset.

TopoVelo inferred the direction of cell differentiation in the developing forebrain (**Fig. 3.6a**, upper left), predicting that radial glia develop into neurons from inside to outside. In general, there is not enough information in a single snapshot spatial transcriptomic dataset to definitively establish that cell migration is occurring. The cell velocity vectors inferred by TopoVelo indicate the direction of cell maturation and sometimes also indicate the direction of cell migration. However, if we know (from biological prior knowledge) that cell migration is occurring in a particular context, our spatial velocity analysis can also potentially reveal the direction and rate of migration. This is because often a cell migrates in the direction of its future gene expression state in the cell state space. This is not guaranteed to be true under every possible circumstance, but it happens in many developmental processes.

For example, the development of mouse cerebral cortex is well-studied, and involves migration of neural progenitor cells that form the layers of the cortex in an inside-out fashion. In fact, previous studies have used live-cell microscopy and cell tracking to directly measure cell migration speeds in this system. Thus, in this case it is reasonable to interpret the cell velocity vectors inferred by TopoVelo as indicating the direction of both cell differentiation and cell migration. We were curious to investigate how the "cell velocity" estimated by TopoVelo compares to experimental cell migration rates from live-cell imaging in mouse cortex.

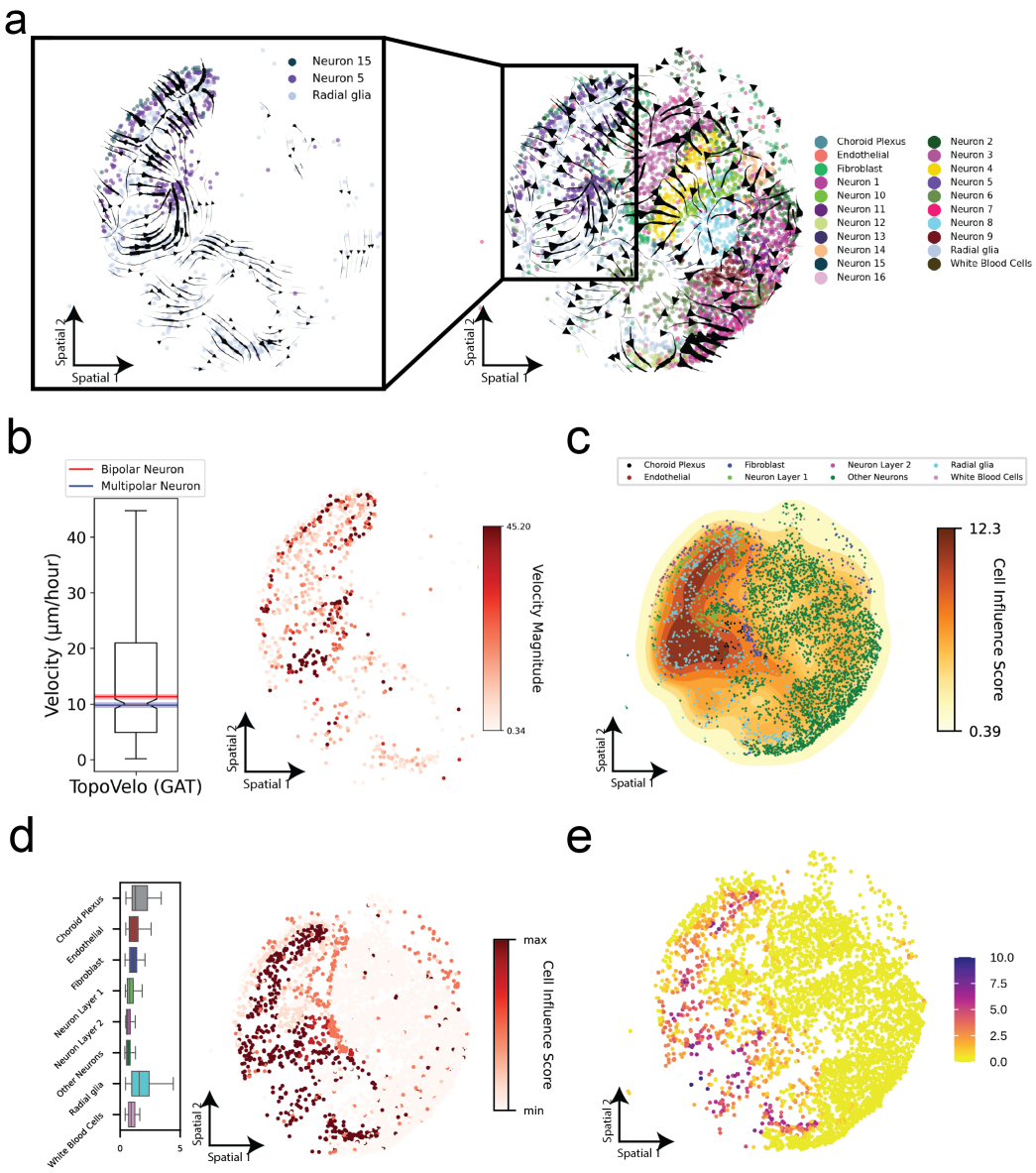

Figure 3.6: **TopoVelo analysis of mouse neural tube closure.** **(a)** Schematic of registering whole mouse embryo Slide-seq data. Each dot represents a slide-seq spot and is colored by its cell type. Red represents the brain and neural tube. Grey represents all other cell types. **(b)** Quiver plot of the cell velocity of the brain and neural tube cells in 3D from TopoVelo. Each dot represent a cell with color indicating the brain or neural tube region. **(c)** Illustration of 3D geometric transformation of the brain and neural tube. **(d)** 3D scatter plot of the brain and neural tube with the principal curve passing through. **(e)** Heatmap plots showing marker gene expression in different brain subregions. **(f)** 3D scatter plot of the original (left) and transformed (right) brain and neural tube. Cells are colored by different subregions, namely forebrain (green), midbrain (red) and hindbrain (blue). **(g)** Top view of the transformed brain and neural tube with putative closure points. Cells are colored by a heatmap indicating the magnitude of cell velocity along the anterior (red) - posterior (blue) axis.

We used TopoVelo to estimate cell velocity in physical units using the known cell-cell distances (in units of μm) from the spatial transcriptomic data. We further followed the standard assumption used in previous RNA velocity papers that the induction/repression cycle of a gene takes 20 hrs[170]. Using these units for time and space measurements, our model predicts a median cell speed of 10 μm/hr within the cortex(**Fig. 3.6b**). Remarkably, this estimate accords well with previous measurements of neuron migration rates from live-cell imaging (9.8 ± 0.4 μm/hr for multipolar neurons and 11.3 ± 0.4 μm/hr for bipolar neurons)[209].

We further investigated the spatial relationships among cell states learned by TopoVelo. To do this, we used the graph attention weights learned by the neural network encoder to evaluate the total attention from each cell to any other cell (Methods). We refer to this value as the cell influence score, because it quantifies how much each cell influences the predicted state of other cells.

Interestingly, we found that the cell influence scores vary systematically by spatial position and by cell type. In particular, radial glia in the deeper portions of the developing cortex region have the highest influence scores, and the score continuously decreases toward the outside of the cortex(**Fig. 3.6c**). Fibroblasts lining the ventricle and choroid plexus cells also have high influence scores(**Fig. 3.6d**). We hypothesized that high influence scores may indicate cells that play important roles in tissue development through ligand-receptor signaling. To test this hypothesis, we used our recently developed CytoSignal[210] method to identify the likely positions at which significant ligand-receptor protein-protein interactions occur in the tissue[125]. Remarkably, CytoSignal indicates that the spatial pattern of the cells with high influence scores closely matches the spatial positions with the largest amount of contact-dependent ligand-receptor signaling(**Fig. 3.6e**). In particular, the spatial locations near radial glia and choroid plexus have among the highest number of significant contact-dependent signaling interactions compared to other tissue positions. The fibroblasts located near the ventricle similarly participate in a large number of diffusion-dependent signaling interactions, providing a possible explanation for their correspondingly high influence scores. These results are especially striking because TopoVelo did not use any information about ligand-receptor interactions during inference.

3.5.4 TopoVelo Reveals Signatures of Mouse Neural Tube Closure

Most spatial transcriptomic measurement techniques are inherently two-dimensional (operating on thin tissue sections), but it is possible to reconstruct complex tissue structures in 3D using serial sections. Such 3D serial section datasets open the exciting possibility of mapping

the temporal dynamics of cell fate transitions that give rise to the spatially intricate, curved structures of the mammalian embryo.

To investigate this possibility, we used TopoVelo to analyze a Slide-Seq dataset from the mouse embryo at day 9 of embryonic development (E9.0)[211]. Serial sagittal sections spanning the whole mouse embryo were cut at 20 μm intervals, then profiled with Slide-Seq (**Fig. 3.7a**). The serial sections were then registered to each other in three dimensions, giving transcriptome-wide gene expression measurements at 10 μm positions spanning the whole embryo. We extracted the brain and neural tube from the embryo and built a graph by connecting all cells within 50 μm of each other. We then used TopoVelo to model the cells with spatially coupled differential equations.

TopoVelo predicted cell velocity results for each position within the 3D volume. We visualized the velocity results using a 3D quiver plot (**Fig. 3.7b**), revealing the complexity of the predicted velocity trends. However, these results are difficult to interpret because the brain and neural tube form a complex, nonlinear shape–and in particular, the anterior-to-posterior (A-P) axis is bent into a U-shaped curve. To aid in interpreting the cell velocity results, we fit an elastic principal curve along the A-P axis, then "unrolled" the brain and neural tube along the principal curve (**Fig. 3.7c, d, f**). We then calculated the velocity component along each of the coordinate axes, including A-P, dorsal-ventral (D-V), and right-left (R-L). We also used marker gene expression patterns to annotate the boundaries of the forebrain, midbrain, and hindbrain (**Fig. 3.7e**).

The sequence of events that lead to neural tube closure in mice is well documented[212]. Between E8.5 and E9.0, neural tube closure begins at three distinct closure points–closure 1 (C1), closure 2 (C2), and closure 3 (C3). C1 begins in the hindbrain and propagates bidirectionally along the A-P axis like two zippers. Similarly, C2 begins in the midbrain and propagates bidirectionally along the A-P axis. C3 begins at the anterior end of the forebrain and propagates in the posterior direction until it meets C2. As C2 propagates in the posterior direction, it meets C1 propagating in the anterior direction.

Remarkably, the A-P component of the cell velocity predicted by TopoVelo matches this known sequence of events. First, we quantitatively determined the closure points by calculating the divergence of cell velocity along the A-P axis . The divergence of cell velocity is the net velocity flow into or out of a position in the tissue. Hence, a positive divergence means that cells are migrating away from that position, while a negative divergence means the opposite. Next, we plotted divergence versus cells' location on the A-P axis and found three peaks with the highest net velocity flow, close to where we expect C1, C2, and C3 to begin based on the literature (**Fig. 3.7g, Supplemental Fig. B.15**). Thus, our spatial velocity analysis allows us to annotate neural tube closure points in a data-driven fashion.

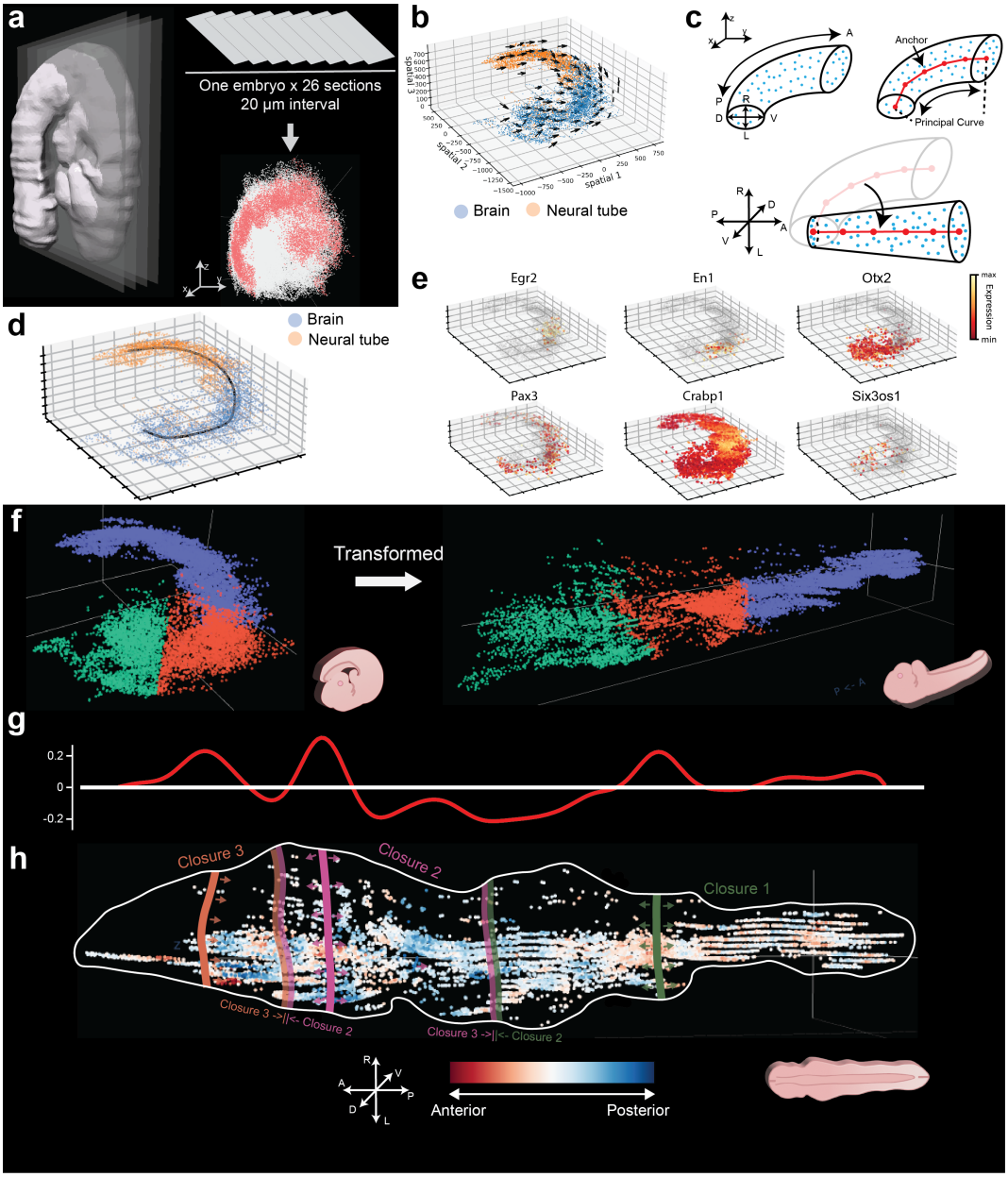

Figure 3.7: **TopoVelo Analysis of Mouse Neural Tube Closure.** **(a)** Schematic of registering whole mouse embryo Slide-seq data. Each dot represents a slide-seq spot and is colored by its cell type. Red represents the brain and neural tube. Grey represents all other cell types. **(b)** Quiver plot of the cell velocity of the brain and neural tube cells in 3D from TopoVelo. Each dot represent a cell with color indicating the brain or neural tube region. **(c)** Illustration of 3D geometric transformation of the brain and neural tube. **(d)** 3D scatter plot of the brain and neural tube with the principal curve passing through. **(e)** Heatmap plots showing marker gene expression in different brain subregions. **(f)** 3D scatter plot of the original (left) and transformed (right) brain and neural tube. Cells are colored by different subregions, namely forebrain (green), midbrain (red) and hindbrain (blue). **(g)** Line plot showing the net flow into or out of a position on the A-P axis. **(h)** Top view of the transformed brain and neural tube with putative closure points. Cells are colored by a heatmap indicating the magnitude of cell velocity along the anterior (red) - posterior (blue) axis.

The cell velocity calculated by TopoVelo also reveals signatures likely associated with neural tube closures intersecting (**Fig.3.7h**). In particular, cell velocity proceeds in the posterior direction from the anterior end of the forebrain until it reaches a segment of cells with velocity in the anterior direction, consistent with the posterior propagation of C3 meeting the anterior propagation of C2. Similarly, cell velocity changes direction in the midbrain, consistent with bidirectional propagation of C2. The cells in the midbrain with posterior velocity, in turn, meet the cells in the hindbrain with anterior velocity, consistent with the posterior propagation of C2 meeting the anterior propagation of C1. Again, the divergence of the cell velocity along the A-P axis shows trends consistent with the neural tube closure points meeting each other: there are local minima with negative divergence at the spatial positions where we would expect the closures to meet each other based on the literature. In short, TopoVelo allowed us to infer complex spatial development patterns in a curved 3D embryo, recapitulating key features of neural tube closure.

3.5.5 TopoVelo Maps Spatial Patterns of Early Differentiation in Human Embryoid Bodies

In vitro models of human differentiation provide a tremendous opportunity to understand fundamental biology and model disease. In particular, 3D self-organizing cell culture models called organoids have been shown to recapitulate key aspects of early human differentiation. As with cell differentiation in vivo, spatial location, and microenvironment play a crucial role in cell fate transition in vitro. However, previous studies have primarily investigated in vitro cell differentiation using dissociated single-cell transcriptomic data. Our lack of understanding of how spatial context influences cell fate specification likely underlies two key limitations of current organoid models: lack of reproducibility and lack of complete maturation compared to in vivo processes. Thus, combining spatial transcriptomic profiling and TopoVelo modeling provide a very promising route to improving and fully utilizing in vitro model systems.

We chose human embryoid bodies (EBs) as a model system to investigate early human differentiation in spatial context. Starting from induced pluripotent stem cells, EBs form 3D spherical structures that mimic the early human embryo, segregating into trophoectoderm, primitive endoderm, and early mesoderm[213]. EBs also recapitulate important spatial properties of developmental processes such as symmetry breaking, asymmetric gene expression, and approximate axis formation and elongation[214]. Another important property of EBs is their ability to generate many cell types–including early members of nearly all human cell lineages–within just 21 days[201, 215]. Thus, EBs provide an opportunity to study the

role of spatial context in the differentiation of many human cell types simultaneously. Additionally, modeling EB differentiation is important because EBs are the starting point for a wide variety of organoid protocols including liver[216], cardiac muscle[217], neuron[218], and blood[219].

We adapted a published protocol to flash-freeze multiple human EBs in a single block, allowing cryo-sectioning and spatial transcriptomics. After growing EBs for 21 days, we used a cryostat to obtain 10 μm frozen sections, each containing multiple EBs in a single field of view. We sectioned onto the capture surface for Curio Seeker, a commercial version of Slide-seq. Curio measures transcriptome-wide polyadenylated gene expression using 10 μm barcoded beads. The spatial position of each barcoded bead is known, providing spatially resolved gene expression at near-cellular resolution. We performed spatial transcriptomics and hematoxylin and eosin (H&E) staining on adjacent sections (**Fig.3.8a**). Within the 9 mm^2 Curio capture area, we were able to measure the transcriptomes and histology of multiple EBs.

Using the RCTD algorithm[200], we mapped the cell types from a previous human EB scRNA dataset onto our spatial transcriptomic data. We identified multiple transcriptionally distinct cell types, including pluripotent cells, endoderm, ectoderm, mesoderm, as well as early endothelial, neural crest, and neural cells (**Fig.3.8a**). We observed relatively few pluripotent cells, likely because we grew EBs for 21 days; previous studies found little expression of pluripotency markers in EBs beyond day 12[213]. The cell types showed clear spatial arrangement within each EB: cells from the same germ layer were generally positioned near each other and pluripotent cells were enriched in the cores of the EBs. Additionally, the cell types were not uniformly distributed among EBs, but rather spatially correlated. That is, the cell type composition varied considerably among EBs, with significant differences in relative proportion of each cell type within each EB. In particular, each EB tended to contain predominantly either endoderm and mesoderm or ectoderm-derived cells expressing neural crest and neuron markers (**Supplemental Fig. 6**). We also observed cell-free voids in the EBs, likely corresponding to the fluid-filled cavitated structures previously reported in late-stage EBs[213].

We next performed spatial velocity analysis on the EB data using TopoVelo (**Fig. 3.8b**). Overall, the direction of cell differentiation is predicted to go from inside to outside. This prediction matches well with the cluster annotations, which indicate that pluripotent and early ectoderm cells are enriched near the cores of the EBs, while more differentiated neurons, ectoderm, and mesoderm are enriched near the surface. These findings are also consistent with previous live-cell imaging data that differentiated cells emerge at surface sites on the outside of pluripotent cells[220].

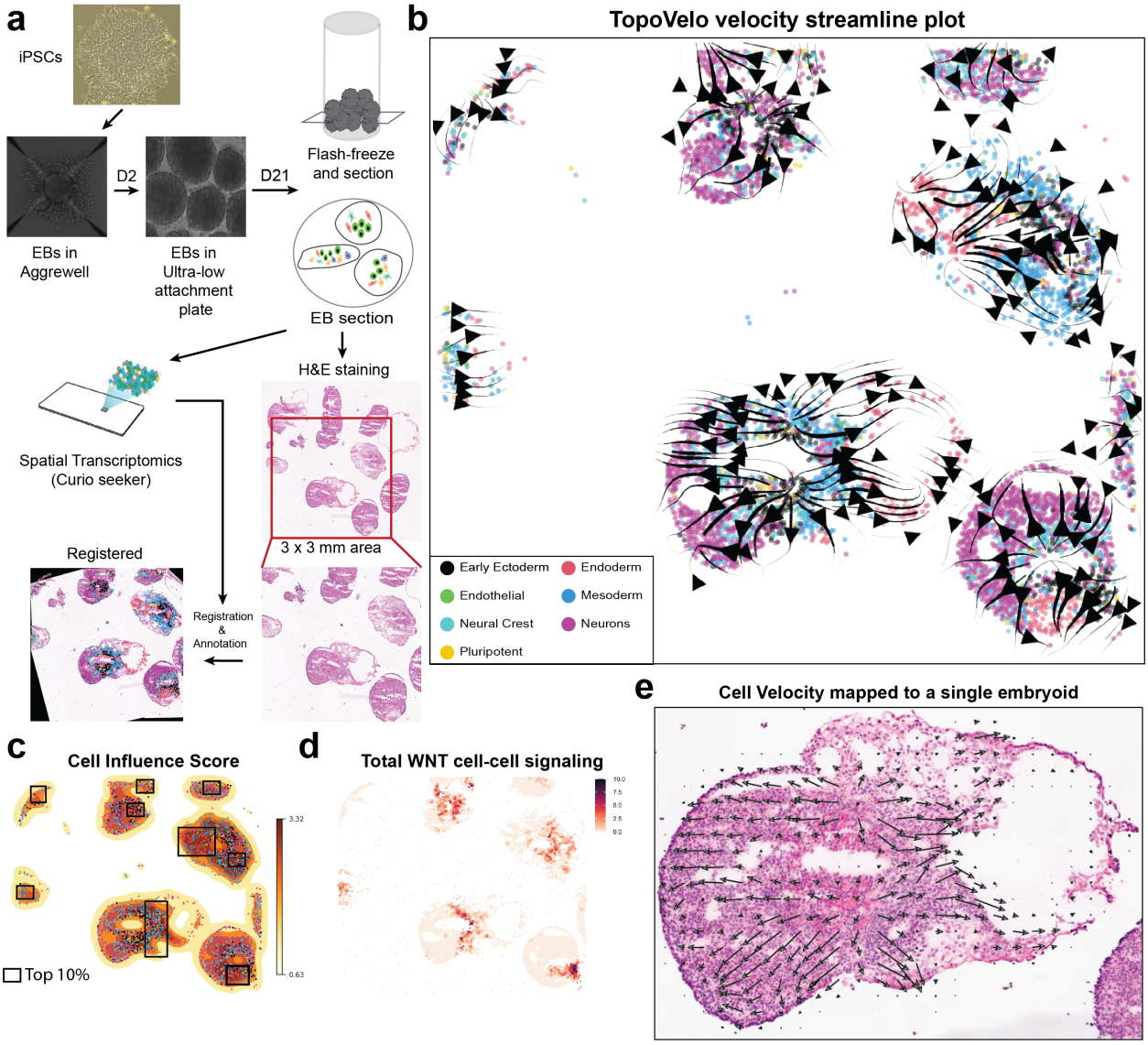

Figure 3.8: **TopoVelo Analysis of Human Embryoid Bodies.** **(a)** Schematic representation of the embryoid bodies (EBs) differentiation and spatial transcriptomics. **(b)** Cell velocity streamline plot from TopoVelo. Cells are colored by annotated cell types. **(c)** Heatmap showing the cell influence score. Cells are colored by annotated cell types. **(d)** Heatmap showing the number of significant cell-cell WNT signaling interactions inferred by CytoSignal. **(e)** Zoomed-in view of the cell velocity streamline plot mapped to the H&E of a single embryoid.

We further investigated the spatial relationships among cells learned by TopoVelo. We calculated cell influence scores using the graph attention weights and visualized the scores on spatial coordinates (**Fig. 3.8c**). Interestingly, the influence scores are concentrated in particular areas of each EB. We next looked at the spatial expression of ligands and receptors to see whether the influential cells participate in signaling interactions. Indeed, we found

a relationship between the spatial pattern of cell influence scores and the total amount of diffusion-dependent signaling interactions identified by CytoSignal[210] (**Fig. 3.8d**), with a Spearman correlation of 0.14 ($p < 2.2 \times 10^{-16}$). Additionally, WNT signaling is highly active in the spatial positions near influential cells (**Fig. 3.8c**). Taken together, these results suggest that TopoVelo has identified cell niches within EBs that may influence the fate transitions of their neighbors by secreting diffusible ligands such as WNT.

Finally, an exciting aspect of mapping cell differentiation in tissue context is the opportunity to link molecular and imaging measurements of the same tissue locations. To investigate this possibility, we used the H&E staining data from EB experiment to identify morphological changes during differentiation. We registered the H&E data to the spatial transcriptomic data (**Section 3.4.8**) and overlayed the cell velocity onto the histological image (**Fig. 3.8e**). This revealed a relationship between histology and spatial velocity. In particular, cells in the middle of the embryoid (primarily early ectoderm cells) show the brightest Eosin signal (pink color). This may indicate increased secretory activity in these cells, because Eosin stains extracellular matrix and cytoplasm. In contrast, more differentiated cells further away from the middle of the embryoid show less Eosin signal.

3.6 Discussion

In this chapter, we consider the problem of modeling gene expression dynamics across an entire population of cells with known spatial locations using spatial transcriptomic data. We present TopoVelo, a variational Bayesian approach for jointly inferring the temporal and spatial dynamics of RNA splicing using graph learning techniques and spatial transcriptomic data. We demonstrate its efficacy by evaluating the method on both simulated and real datasets and calculating quantitative metrics. To our knowledge, TopoVelo is the first model that can use spatial transcriptomic data to link space and time in this way. Although the concept of velocity in space is well known in classical physics, it has not been quantitatively studied in computational biology due to the absence of appropriate measurement data and a theoretical foundation. Cell velocity opens the door to investigating the spatial mechanisms of gene expression. We believe that our work is only the first step in this exciting direction, and there are numerous questions for future research to address.

First, although we use spatial position to build a spatial graph, we do not explicitly model gene expression as a function of spatial position. Future work could incorporate domain knowledge about tissue growth and mechanistic models describing the spatial movement of cells. Second, there is significant room for improving the differential equation model of gene expression. For example, our current approach does not account for cell-specific variations

in splicing, degradation rates, or transcriptional variance. Modifying these assumptions to better capture gene expression dynamics is another direction for future research. The spatial mechanism of gene expression also involves cell-cell communication through ligand-receptor binding. Incorporating this interaction information directly into our ODE formulation and graph learning methods would be of high biological interest. Finally, we are excited to apply TopoVelo to gain a more comprehensive understanding of cell differentiation across a wide range of tissue environments. We anticipate that TopoVelo will be useful for studying both normal and aberrant cell differentiation in vivo and in vitro, deepening our understanding of how spatial context shapes cell fate.

CHAPTER 4

Inferring Causal Mechanism of Gene Expression

4.1 Motivation

Genes form a complex biological network with unknown interconnections, known as the gene regulatory network. For example, regulatory gene expression produces transcription factors that bind to specific regulatory regions on DNA, thereby controlling the expression of downstream genes. This is considered a causal mechanism for gene expression and is of high interest to many biological studies in disease research and drug discovery. Recent developments in sequencing technology [49, 50, 42, 43, 44, 45, 46, 47] have enabled the mass production of single-cell RNA sequencing data with interventions. These technologies either treat cells with different chemicals or use specific enzymes such as CRISPR-Cas9 to suppress or enhance the expression of certain genes and measure the gene expression outcomes under such perturbations. Consequently, new algorithms are needed to better identify gene regulatory relationships using this new type of transcriptomic data. In this chapter, we study the problem of causal discovery from interventional data using differentiable methods.

Discovering causal relationships is a challenging task that arises in many scientific domains. Often, both observational and interventional experiments can be performed to measure the effects of underlying causal mechanisms. Nonetheless, identifying causal relationships remains a difficult problem in large, complex systems.

Classical methods for this problem date back to the PC algorithm[137, 140] and infer causal relationships by searching over a discrete space of directed acyclic graphs (DAGs), performing statistical tests and iteratively removing cycles. However, these discrete search approaches do not scale to causal graphs with thousands of nodes and datasets with hundreds of thousands of measurements. Later, score-based methods were proposed[221, 222, 223, 224, 225, 226]. The main idea behind these methods is to perform greedy search over the DAG space and maximize a score function. Due to the discrete nature and large search space,

classical score-based methods have high time complexity and poor scalability[221], are unable to handle cycles[147] and are sensitive to initial conditions.

Recently, a number of differentiable methods emerged as alternatives to discrete search[147, 148, 149, 151, 152, 154, 153]. These methods optimize a score function by parameterizing DAGs and maximizing the data likelihood under a structural equation model (SEM). While differentiable methods outperform discrete search approaches and scale to large and complex datasets, they still suffer from several limitations. First, differentiable methods approximately enforce the acyclicity constraint by calculating a surrogate penalty function, whose computation is expensive and does not guarantee that the final graph is strictly acyclic. Second, while differentiable methods often come with theoretical guarantees about identifiability in terms of a graph equivalence class, they lack a probabilistic interpretation. This limitation prevents reasoning about uncertainty or incorporating prior knowledge. Finally, most differentiable methods consider only interventions that target specific, known nodes in a causal graph. In many real-world settings, however, the precise targets of an intervention are unknown. For example, in biological systems, chemical treatments can affect multiple genes, and the precise mechanism of action for a given chemical is often unknown. Thus, jointly identifying causal relationships and intervention targets under untargeted interventions is of practical importance but remains relatively unexplored.

Another related stream of work uses approximate Bayesian inference to learn causal graphs. Bayesian causal inference methods offer principled approaches for quantifying uncertainty and incorporating structured priors, but scale poorly and lack the explicit identifiability results of score-based methods.

In this chapter, we propose **A**mortized **B**ayesian **C**ausal **D**iscovery of **E**xtended **F**actor **G**raphs (ABCDEFG). Our key insight lies in formulating a generative model for DAGs that enforces acyclicity by construction, allowing scalable approximate Bayesian inference without the calculation of an expensive acyclicity penalty. Additionally, our framework unifies previous Bayesian and differentiable methods, allowing us to reason about both the distributional properties and the identifiability of the causal graphs inferred by ABCDEFG.

4.2 Technical Contributions

We make the following technical contributions:

1. We propose a new parametric model for DAGs that is acyclic by construction.

2. We incorporate probabilistic circuits into the generative model, allowing flexible modeling of complex joint distributions over causal edges.

3. We establish a Bayesian framework for causal discovery and derive new theoretical results connecting our Bayesian framework to the identifiability guarantees of score-based methods.

4. We develop a variational Bayesian method for causal discovery and demonstrate its superior performance on causal discovery tasks.

5. We extend our Bayesian framework to include interventional nodes and model interventions with unknown targets.

6. We apply our method to a large-scale real dataset of cellular response to growth factor treatment to infer gene-gene and treatment-gene relationships.

4.3 Causal Discovery

The basic setting for causal discovery is an unknown graph connecting a set of variables to form a random field. Each variable has associated observations. For example, in genetic causal discovery, genes are nodes in a causal graph and gene expression levels (RNA count numbers) are observed values. Score-based methods tend to design a score function, e.g. the likelihood of observing the data given a graph and distributional assumptions, and search for a graph that maximizes the score. Most previous works consider the set of DAGs due to desired mathematical properties such as identifiability as well as convenience. This dissertation continues this track of research and mainly focuses on learning genetic DAGs from transcriptomic data with perturbation. Thus, we introduce previous theoretical work of differentiable causal discovery as the background for our proposed method.

4.3.1 Problem Setup

We consider a system of n causally related random variables, $\{X_i : i \in [n]\}$. Assume that observations of X_i are generated from a structural equation model (SEM) with a DAG, G^*. In our application, we also assume that the causal graph is acyclic and has a low-rank structure. In addition, we have a set of interventions represented by a set $\mathcal{I} = \{I_1, \ldots, I_{n^{\mathcal{I}}}\}$. Each I_k is a set of nodes targeted by the k-th intervention, and $I_1 = \emptyset$. Denote $\boldsymbol{\pi_i}$ as the set of parents of i in G^*. Under the Markov property assumption, the joint distribution of $\boldsymbol{X}$ under the k-th intervention can be factorized as follows:

$$p^{(k)}(\boldsymbol{X}) = \prod_{i=1}^{n} p^{(k)}(X_i | \boldsymbol{X}_{\boldsymbol{\pi_i}}). \tag{4.1}$$

Here, $p^{(k)}(X_i|\boldsymbol{X}_{\boldsymbol{\pi_i}}) = p^{(1)}(X_i|\boldsymbol{X}_{\boldsymbol{\pi_i}}),\ \forall i \notin I_k$. Given observations of X_i under interventions, our goal is to discover the true causal graph, G^*.

4.3.2 Differentiable Causal Discovery

NOTEARS Zheng et al.[147] replaced combinatorial search with a continuous optimization problem and named the method NOTEARS, short for "**N**on-combinatorial **O**ptimization via **T**race **E**xponential and **A**ugmented lag**R**angian for **S**tructure learning". Given a set of random variables $X_i : i \in [d]$, the authors assumed a linear SEM: $X_j = \boldsymbol{w_j}^T \boldsymbol{X} + z_j$ where $\boldsymbol{X} = [X_1, \ldots, X_d]^T$ and z_j is noise. The weights $\boldsymbol{w_j}$ are columns of a weighted adjacency matrix $\boldsymbol{W}$. Then, they replace the combinatorial constraint of acyclicity with an equality constraint $h(\boldsymbol{W}) = 0$ so that the problem becomes:

$$\min_{\boldsymbol{W}:h(\boldsymbol{W})=0} F(\boldsymbol{W}) + \frac{\rho}{2} h^2(\boldsymbol{W}) \tag{4.2}$$

Here, they propose a constraint $h(\boldsymbol{W}) = \operatorname{Tr}(\exp(W \circ W)) = 0$, where $\circ$ denotes the Hadamard product. Such an optimization can be solved by augmented Lagrangian:

$$\max_{\alpha} \min_{\boldsymbol{W}} F(\boldsymbol{W}) + \frac{\rho}{2} h^2(\boldsymbol{W}) + \alpha h(\boldsymbol{W}) \tag{4.3}$$

The exact algorithm uses gradient-based iterative optimization. A follow-up work[148] reduced computational cost and designed a polynomial regression loss tailored to gene expression data.

Differentiable Causal Discovery with Interventions (DCDI). Later, a method called DCDI[149] extended this continuous optimization framework to include interventional data with both known and unknown interventions and proved the identifiability up to an $\mathcal{I}$-Markov equivalence class[145], a subset of a Markov equivalence class. Here, we introduce some key concepts from the work as our method builds on this work. Given a set of causally related random variables $\boldsymbol{X} = \{X_1, \ldots, X_n\}$ with a causal graph G. A fundamental assumption of a causal DAG underlying $\boldsymbol{X}$ is the Markov property, which leads to a factorization of the joint distribution. We denote $\boldsymbol{\pi_i}$ as the set of all parents of i in G. Here, we present some key results from previous works. For a complete review of relevant concepts, please refer to **Supplemental Section C.2.1**.

Definition 4.1 (Causally Factorized Joint Distribution[149])**.** *Let $G = (V, E)$ be a causal DAG with n nodes and $\mathcal{I}^* = \{I_k : k \in [l]\}$ be a set of interventions. We define $\boldsymbol{\mathcal{M}}_{\mathcal{I}^*}(G)$ as*

the set of joint distributions factorized according to the Markov property, i.e. $\boldsymbol{\mathcal{M}}_{\mathcal{I}^*}(G) := \{\{p^{(k)} : k \in [n^{\mathcal{I}^*}]\} : p^{(k)}(\boldsymbol{X}) = \prod_{i=1}^{n} p^{(k)}(X_i|\boldsymbol{X}_{\boldsymbol{\pi_i}})\}$

By convention, $I_1 = \emptyset$ represents a pure observational setting. Based on the definition above, Brouillard et al.[149] defined a type of equivalence relation called $\mathcal{I}$-Markov equivalence relation to describe DAG equivalence under interventions.

Definition 4.2 ($\mathcal{I}$-Markov Equivalence[145])**.** *Two DAGs* G_1 *and* G_2 *are* $\mathcal{I}$*-Markov equivalence if and only if* $\boldsymbol{\mathcal{M}}_{\mathcal{I}}(G_1) = \boldsymbol{\mathcal{M}}_{\mathcal{I}}(G_2)$*. We denote by* $\mathcal{I}$*-MEC*(G) *as the set of all DAGs which are* $\mathcal{I}$*-Markov equivalent to* G*.*

Yang et al.[149] also showed that by assuming faithfulness, we can prove $\mathcal{I}$-Markov equivalence by showing graphs augmented with intervention nodes share the same structures. This is presented in the following theorem.

Theorem 4.3 (Sufficent and Necessary Conditions for $\mathcal{I}$-Markov Equivalence[145])**.** *Two DAGs* G_1 *and* G_2 *belong to the same* $\mathcal{I}$*-Markov Equivalence Class (*$\mathcal{I}$*-MEC) if and only if their* $\mathcal{I}$*-DAGs have the same skeleton and v-structures.*

We use the notation $\simeq_{\mathcal{I}}$ to denote $\mathcal{I}$-Markov equivalence relation. Since we consider the set of f-DAGs, the causal relations between i and j are passed through latent factors. Denote $\boldsymbol{\pi}_{\boldsymbol{i}}^{\boldsymbol{D}}$ as the set of parents of a vertex i(node or factor) in the graph D. Next, we use a continuous random variable $\boldsymbol{Z} = \{Z_1, \ldots, Z_m\}$ to represent the factors. Then, we have a class of joint distributions of $\boldsymbol{X}$ and $\boldsymbol{Z}$ produced by an f-DAG.

Definition 4.4 (Family of Distributions associated with an f-DAG)**.** *Let* $D = (V, F, E)$ *be an f-DAG with* n *nodes and* m *factors and* m *factors and* $\mathcal{I}^* = \{I_1, \ldots, I_k\}$ *be a set of interventions.. Then,* $\boldsymbol{\mathcal{M}}_{\mathcal{I}^*}(D)$ *is defined as the set of probabilistic models with the following form:*

$$\boldsymbol{\mathcal{M}}_{\mathcal{I}^*}(D) = \left\{ \{p^{(k)}(\boldsymbol{X}, \boldsymbol{Z}) : k \in [n^{\mathcal{I}^*}]\} : p^{(k)}(\boldsymbol{X}, \boldsymbol{Z}) = \prod_{i=1}^{n} p^{(k)}(X_i|\boldsymbol{Z}_{\boldsymbol{\pi}_{\boldsymbol{i}}^{\boldsymbol{D}}}) \prod_{j=1}^{m} p^{(k)}(Z_j|\boldsymbol{X}_{\boldsymbol{\pi}_{\boldsymbol{j}}^{\boldsymbol{D}}}) \right\} \tag{4.4}$$

where $p^{(k)}(X_i|\boldsymbol{Z}_{\boldsymbol{\pi}_{\boldsymbol{i}}^{\boldsymbol{D}}}) \neq p^{(1)}(X_i|\boldsymbol{Z}_{\boldsymbol{\pi}_{\boldsymbol{i}}^{\boldsymbol{D}}})$ *if and only if* $i \in I_k$ *and* $p^{(k)}(Z_j|\boldsymbol{X}_{\boldsymbol{\pi}_{\boldsymbol{j}}^{\boldsymbol{D}}}) \neq p^{(1)}(Z_j|\boldsymbol{X}_{\boldsymbol{\pi}_{\boldsymbol{j}}^{\boldsymbol{D}}})$ *if and only if* $j \in I_k$*.*

The authors of DCDI dervied a nice property of score-based differentiable methods – identifiability, under moderate assumptions. This is an important result as it applies to all score-based differentiable causal discovery methods.

Theorem 4.5 (Identifiability of Differentiable Causal Discovery[149]). *Let* $\boldsymbol{X} = \{X_1, \ldots, X_n\}$ *be a set of causally related random variables with a causal DAG* $G^* = (V, E)$ *and* $\mathcal{I}^* = \{I_k : k \in [n^{\mathcal{I}^*}]\}$ *be a set of interventions with* $I_1 = \emptyset$. *Assume the following:*

1. *The set of distributions from our parametric models contains the ground truth interventional distributions:* $\{p^{(k)}(\boldsymbol{X}) : k \in [n^{\mathcal{I}^*}]\} \in \boldsymbol{\mathcal{F}}_{\mathcal{I}^*}(G^*)$ *where* $\boldsymbol{\mathcal{F}}_{\mathcal{I}^*}(G^*) = \{\{f^{(k)}(\boldsymbol{X}|G^*; \boldsymbol{\Phi})\} : \boldsymbol{\Phi} \in \Omega(\boldsymbol{\Phi})\}$.

2. *Denote* $\perp\!\!\!\perp_{G^*}$ *as the d-separation relation in* G^*. *$\mathcal{I}$-faithfulness contains the following two conditions.*

   (a) *For any disjoint set* $A, B, C \subset V$, $\boldsymbol{X_A} \perp\!\!\!\perp \boldsymbol{X_B}|\boldsymbol{X_C} \implies A \perp\!\!\!\perp_{G^*} B|C$

   (b) *For any disjoint sets* $A, C \subset V$ *and* $k \in [n^{\mathcal{I}^*}]$, $p^{(k)}(\boldsymbol{X_A}|\boldsymbol{X_C}) = p^{(1)}(\boldsymbol{X_A}|\boldsymbol{X_C}) \implies A \perp\!\!\!\perp_{G^{*\mathcal{I}^*}} \xi_k|C$

3. $\forall G, I, \boldsymbol{\Phi}, f^{(k)}(\boldsymbol{X}|G, I; \boldsymbol{\Phi}) > 0$.

4. $\forall k \in [n^{\mathcal{I}^*}]$, $\left|\mathbb{E}_{p^{(k)}(\boldsymbol{X})}\left[\log p^{(k)}(\boldsymbol{X})\right]\right| < +\infty$.

Define the score function as

$$S_{\mathcal{I}^*}(G) = \sup_{\boldsymbol{\Phi}} \sum_{k=1}^{n^{\mathcal{I}^*}} \mathbb{E}_{p^{(k)}(\boldsymbol{X})}\left[\log f^{(k)}(\boldsymbol{X}|G; \boldsymbol{\Phi})\right] - \lambda|G|$$

Then, with a small enough $\lambda > 0$, *we have* $S_{\mathcal{I}^*}(G^*) > S_{\mathcal{I}^*}(G)$.

Although DCDI alleviates the problem of high searching time complexity, it still suffers from high computational cost due the computation of the acyclic constraint. Calculating the trace of matrix exponential requires $O(d^3)$ complexity where d is the number of nodes in a causal graph. In addition, computing the likelihood also requires $O(d^2)$ time complexity. The human genetic causal graph involves more than 1500 transcription factors[227], so it remains a problem to apply these algorithms to large-scale causal graphs.

Factor Directed Acyclic Graphs (f-DAGs) Our goal is to build a generative model for DAGs and ultimately a Bayesian framework for inferring causal DAGs. To do this, we start with a type of graph called a factor DAG (f-DAG). To further accelerate causal discovery for large-scale causal graphs, Lopez et al.[151] deployed the low-rank property of graph adjacency matrices and proposed a causal discovery method, DCD-FG, based on the concept of a factor graph, for modeling causal relations. An f-DAG is a DAG with two types of vertices – nodes and factors – and edges only from nodes to factors or from factors to nodes. They factorize

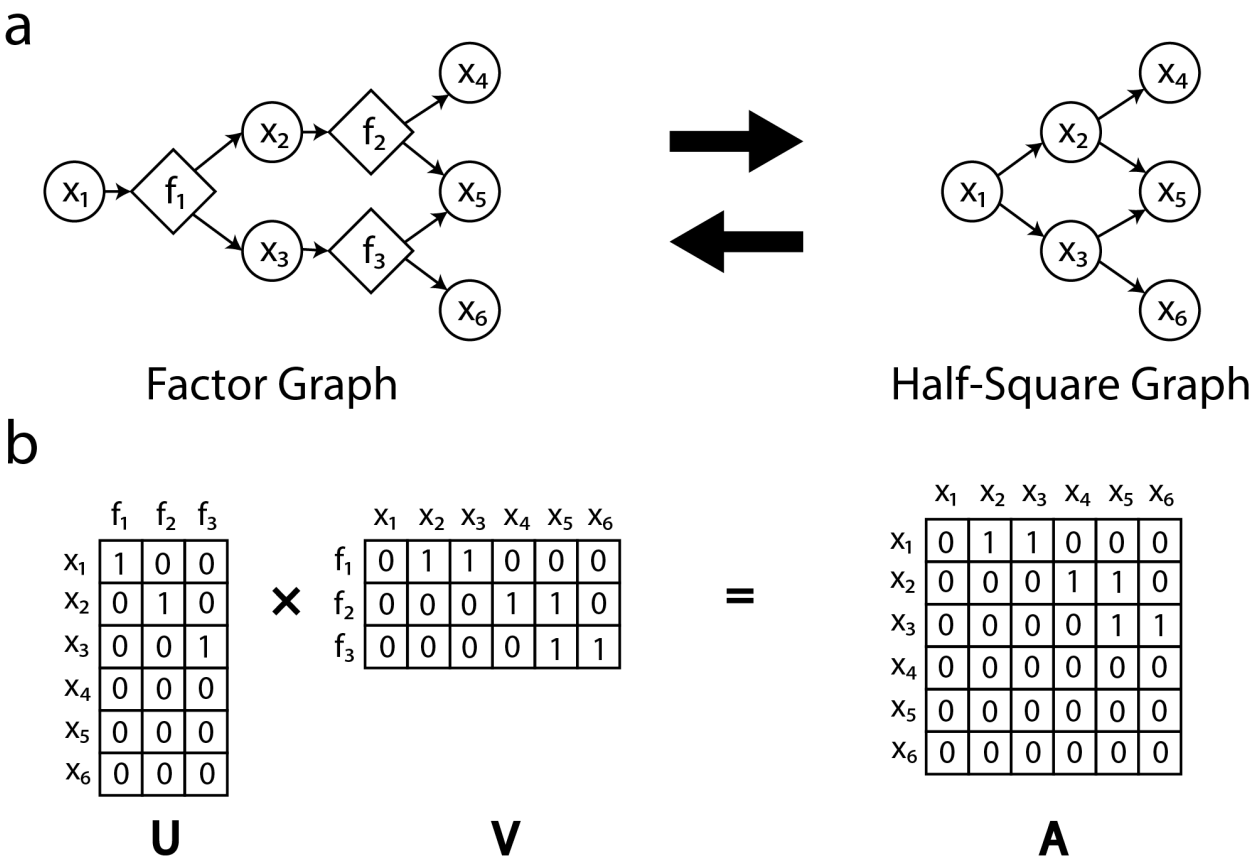

Figure 4.1: **Illustration of f-DAG. (a)** Graph representation of an f-DAG and its corresponding half-square node graph. Circular shapes denote nodes and diamond shapes denote factors. **(b)** Algebraic representation of an f-DAG and its corresponding half-square node graph. The cross on left hand side denotes matrix Boolean product. x_1 - x_6 denote nodes and f_1 - f_3 denote factors. **U**, **V** and **A** denote a node-to-factor, factor-to-node and node-to-node adjacency matrix.

a binary adjacency matrix $\mathbf{A}$ into a Boolean product of two matrices $\mathbf{U} \in \mathbb{R}^{d \times m}, \mathbf{V} \in \mathbb{R}^{m \times d}$ where $m \ll d$. $\mathbf{U}$ is called a node-to-factor matrix and $\mathbf{V}$ a factor-to-node matrix. Here, we give a formal definition.

Definition 4.6 (Factor Directed Acyclic Graph (f-DAG)[151])**.** *Given a set of nodes, V, and factors, F, a factor directed acyclic graph (f-DAG), denoted as (V, F, E), is a directed acyclic graph $(V \cup F, E)$ where edges $E \subseteq \{(i, j) : i \in V, j \in F \text{ or } i \in F, j \in V\}$.*

An f-DAG is a DAG with two different types of vertices, nodes and factors. All edges connect two vertices of different types. Alternatively, if we represent an f-DAG using an adjacency matrix $\mathbf{A}$, we can use $\mathbf{U}$ and $\mathbf{V}$ to represent node-to-factor and factor-to-node adjacency matrices, respectively. Then, we have $\mathbf{A} = \mathbf{U} \circ \mathbf{V}$ where $\circ$ denotes the matrix

Boolean product. Furthermore, we can condense an f-DAG into a node-only graph as defined below.

Definition 4.7 (Half-Square Graph[151]). *Given an f-DAG, $D = (V, F, E)$, its half-square node graph is defined as $D^2[V] = (V, \{(i, j) : \exists f \in F, (i, f), (f, g) \in E\})$, and half-square factor graph is defined as $D^2[F] = (F, \{(f, g) : \exists i \in V, (f, i), (i, g) \in D\})$.*

A half-square graph essentially keeps all dependency relations between nodes in the original factor graph. The factors can be interpreted as intermediate nodes on the paths between causally-related observations. **Fig. 4.1** illustrates the relation between an f-DAG and its half-square graph. We also note that the mapping from the set of f-DAGs to half-square graphs is an surjection.

If we assume the true gene regulatory network as an f-DAG with n nodes and m factors, an equivalent acyclicity constraint becomes

$$\mathrm{Tr}(\exp(\mathbf{VU})) = m$$

The trace of matrix exponential can be approximated by the spectral norm[148]. The factor graph model reduces computational complexity of both likelihood evaluation and acyclic constraint calculation to $O(md)$.

Existing differentiable approaches have the advantage of identifiability guarantee and computational efficiency. However, there are a number of limitations:

1. The acyclicity constraint is enforced softly via augmented Lagrangian. Graphs inferred from previous methods could still have cycles that are removed post hoc by pruning.

2. Existing differentiable sampling methods for graphs are over-simplified and cannot effectively model graphs with non-independent edges.

3. Existing Bayesian causal discovery methods lack a theoretical guarantee of graph identifiability.

4. No existing method can identify the targets of soft interventions with indirect effects.

To address these limitations, we developed a new method name Amortized Bayesian Causal Discovery of Extended Factor Graphs (ABCDEFG). Our method have multiple components as described in the next section.

4.4 Methods

4.4.1 Probabilistic Modeling of f-DAGs

Generative Model for f-DAGs A key insight behind our approach is a generative process for sampling f-DAGs that guarantees acyclicity by construction. This property avoids the computationally expensive acyclicity penalties required by existing differentiable causal discovery methods, always yields acyclic graphs, and serves as the foundation for a probabilistic treatment of causal f-DAG inference. Consider a set of n nodes, $\{v_i : i \in [n]\}$, and m factors, $\{f_j : j \in [m]\}$. We can construct an f-DAG in two steps: (1) form a topological ordering of nodes and factors together and (2) determine the node-to-factor or factor-to-node edge connection (**Fig. 4.2**). The first step determines edge *direction*, as we stipulate that edges can only point towards nodes or factors with higher rank in the ordering. Because nodes in an f-DAG are connected only via factors (there are no direct node-to-node connections), it is unnecessary to explicitly model the relative ordering between nodes. Instead, the m factors partition all nodes into $m + 1$ subsets in the topological ordering and each node is randomly inserted into one partition. This can be modeled as n categorical distributions with $m + 1$ categories, denoted as $\boldsymbol{Y} = \{Y_i : i \in [n]\}$. The second step determines edge *existence*, regardless of direction. These edge connection probabilities are related to a joint distribution of all edge connections. We use a binary matrix $\boldsymbol{B} \in \{0, 1\}^{n \times m}$ to represent edge connections. Thus, $\boldsymbol{Y}$ contains all the direction information and $\boldsymbol{B}$ contains all the connection information. Hence, $\boldsymbol{Y}$ and $\boldsymbol{B}$ uniquely determine an f-DAG, and we can generate an f-DAG by sampling $\boldsymbol{Y}$ and $\boldsymbol{B}$.

Modeling Joint PMFs with Sum-Product Networks (SPNs) Using the above generative process, we can infer a causal DAG by optimizing a score function with respect to $\boldsymbol{Y}$ and $\boldsymbol{B}$. But what is the best way to do this, given that $\boldsymbol{Y}$ and $\boldsymbol{B}$ are discrete? One possibility is the Gumbel softmax trick[228], often applied due to its simplicity. For $\boldsymbol{Y}$, we can parameterize each Y_i with logits $\boldsymbol{\theta_i}$ and sample Y_i using Gumbel softmax. Similarly we can treat each edge in $\boldsymbol{B}$ as a Bernoulli random variable and sample from Gumbel softmax. However, such a naive approach treats all edges as independent and neglects possible correlation between edges.

A more general approach is to model the joint distribution of edges in $\boldsymbol{B}$ using a sum-product network[229]. Two naive ways to sample a binary vector $\mathbf{b} \in \{0, 1\}^d$ are (1) sample from a single categorical distribution over all binary vectors or (2) sample each entry independently from a Bernoulli distribution. The former involves 2^d categories, which is impractical for large d, while the latter neglects dependency between any two entries and

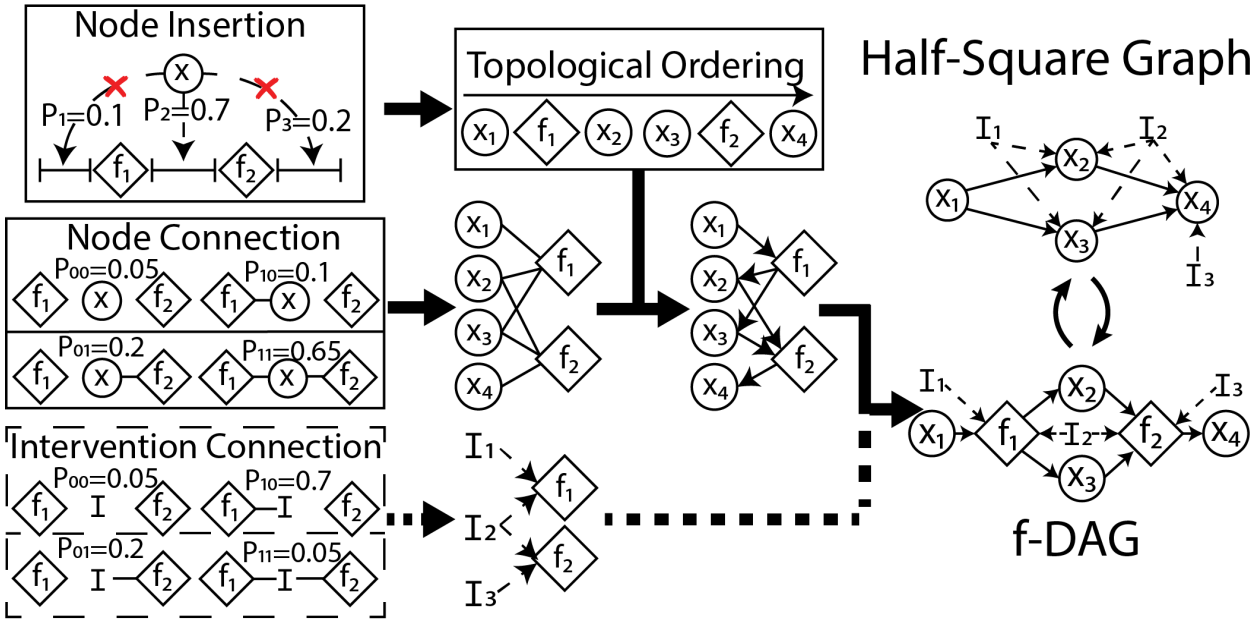

Figure 4.2: **Construction of an f-DAG.** Dashed lines and blocks indicate optional steps for extended f-DAGs with unknown intervention targets. **(a)** Generative process of f-DAGs. Generation proceeds from left to right. **Left**: Random sampling of node partition (top), undirected node-to-factor graph (middle) and optionally intervention-to-node bipartite graph (bottom). Note the sampling steps are repeated independently for each node. **Middle**: Sampling Results. Random node insertion results in a partial ordering of node and factors. Connection sampling specifies undirected edge connection. **Right**: Formation of an f-DAG by combining the information from partial ordering, undirected edge connection and optional intervention-to-node connection.

lacks expressiveness. In contrast, SPNs provide an appealing parametric model for $\boldsymbol{B}$ due to their balance between model complexity and expressiveness.

Let $\boldsymbol{B} = [B_1, \ldots, B_d]^T \in \{0,1\}^d$ be a random binary vector. We applied and extended the algorithm by Shih and Ermon[230] to construct an SPN to model the joint distribution of $\boldsymbol{B}$. The construction of an SPN is analogous to building a neural network by sequentially adding layers. Each layer contains one type of computation nodes: (1) input node, (2) product node and (3) sum node and acts as a function of input as shown in **Fig. 4.3a**. The SPN starts with singletons $\{b_1\}, \ldots, \{b_d\}$ as an initial partition. Each b_i is passed to two input nodes outputting 0 and 1 respectively. Next, each product layer merges the partitions from the previous layer by creating all combinations of bit sequences for each merge. When the number of sequences from a merge exceeds a threshold, w, a sum layer is added to filter out sequences from the previous layer while keeping the same number of partitions. The merge filter process continues until a single partition remains. The final output is a single bit sequence sampled from a joint PMF of bits (**Fig. 4.3b**). Thus, the SPN can also be

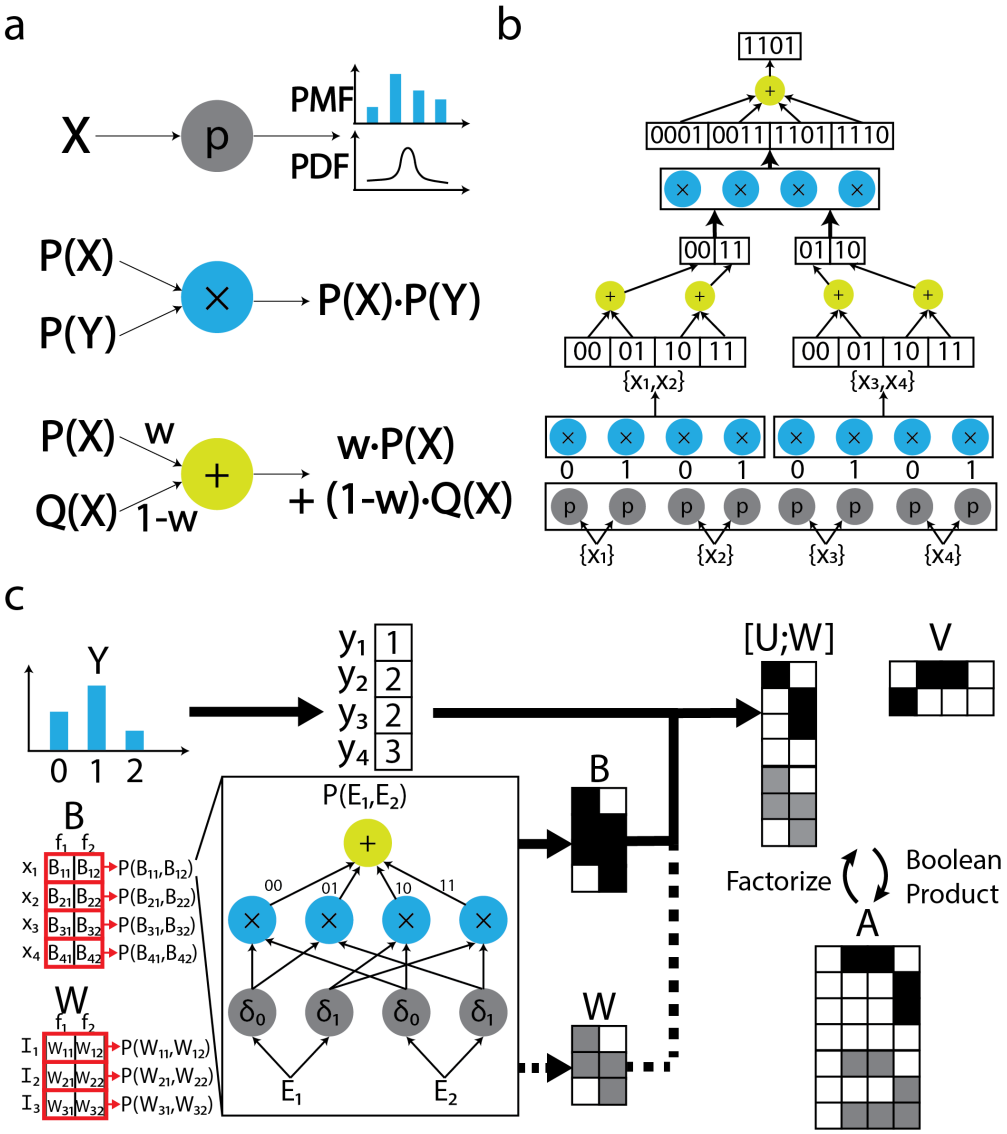

Figure 4.3: **Overview of the SPN-FG Model.** **(a)** Building blocks of a sum-product network (SPN). **Top**: An input node encodes a PMF or PDF given an input value x. **Middle**: A product node generates a product of input distributions as the output. **Bottom**: A sum node generates a mixture of input distributions as the output. **(b)** An Example of SPN Architecture. Assume inputs are random bits $x_1 \dots x_4$. Input nodes generate both 0's and 1's for each bit. Next, a product layer merges x_1, x_2 and x_3, x_4 by generating all 2-bit sequences for $\{x_1, x_2\}$ and $\{x_3, x_4\}$ respectively. Then, a sum layer downsamples inputs. Finally, a product and a sum layer merge $x_1, \dots, x_4$ together and output a 4-bit sequence. **(c)** Illustration of the SPN-FG Model. **Left**: Parametric models for random quantities specifying an f-DAG. Node-factor partial ordering is determined by a Categorical random variable $\boldsymbol{Y}$. Undirected node-to-factor and directed intervention-to-node connections are determined by random binary matrices $\mathbf{B}$ and $\mathbf{W}$. Each row of $\mathbf{B}$ and $\mathbf{W}$ follows a joint PMF parameterized by an SPN. **Middle**: Sampled partition values $y_1, \dots, y_4$ and binary matrices $\mathbf{B}$ and $\mathbf{W}$. **Right**: Matrix form of f-DAG-to-DAG conversion. Node-to-factor and factor-to-node matrices are determined by the partition $\boldsymbol{Y}$ and connection matrix $\mathbf{B}$.

interpreted as a deep mixture model whose trainable parameters are the mixture weights of all sum nodes.

The original algorithm by Shih and Ermon[230] only works when d is a power of two due to recursively halving the partitions, but we extended it to the general case. To do this, we divide d into powers of two based on its binary representation: $d = \sum_{i=0}^{k} b_i \times 2^i$. Next, for each $b_i = 1$, we build an SPN modeling joint PMF of 2^i bits. Finally, we apply a product and sum unit to merge the outputs from each SPN together. The number of parameters in an SPN with a maximum width of w for an f-DAG with m factors and n nodes scales as $\boldsymbol{\Theta}\left(\frac{mnw^2}{\log w}\right)$, achieving a balance between model size and model expressiveness (**Supplemental Section C.1**).

Now we present the SPN-FG model. SPN-FG is a parametric probabilistic model for f-DAGs. It consists of two parts, node partition, $\boldsymbol{Y}$, and node connection matrix, $\boldsymbol{B}$. $\boldsymbol{Y} = \{Y_1, Y_2, \ldots, Y_n\}$ where n is the number of nodes and Y_i is a categorial random variable parameterized by $\boldsymbol{\theta_i}$. $\boldsymbol{B}$ is a binary matrix whose row or column distributions (PMF on binary random row or column vectors) are modeled by separate SPNs. Thus, we can easily construct a DAG by sampling Y_i independently and $\boldsymbol{B}$ from a joint PMF encoded by SPNs. Suppose the sample space of Y_i is $\{0, 1, \ldots, m\}$ where m is the number of factors.

Next, $\mathbf{U}$ and $\mathbf{V}$ can be determined by the following rule:

$$U_{ij} = \begin{cases} 1 & B_{ij} = 1 \text{ and } Y_i < j \\ 0 & \text{else} \end{cases}, \quad i \in [n], j \in [m] \tag{4.5}$$

$$V_{ij} = \begin{cases} 1 & B_{ij} = 1 \text{ and } Y_i \geq j \\ 0 & \text{else} \end{cases}, \quad i \in [n], j \in [m] \tag{4.6}$$

Once $\mathbf{U}$ and $\mathbf{V}$ are determined, the f-DAG is determined. **Fig. 4.3c** illustrates the generative process of the SPN-FG model.

Extended f-DAG The previous f-DAG model by Lopez et al.[151] is limited to known intervention targets. We further extend it for identifying unknown intervention targets. In this extended f-DAG model, we model the effect of each intervention on target nodes via factors. This is a natural model for interventions whose exact targets are unknown, such as drugs that affect a biological pathway.

Suppose $\mathcal{I} = \{I_1, \ldots, I_{n^{\mathcal{I}}}\}$ is a set of unknown intervention targets. We use a $n^{\mathcal{I}}$-by-m binary matrix $\mathbf{W}$ where W_{kj} represents whether the k-th intervention targets the j-th factor. Therefore, we define an extended f-DAG as follows:

Definition 4.8 (Extended f-DAG)**.** *Let $D = (V, F, E)$ be an f-DAG and $\mathcal{I} = \{I_1, \ldots, I_{n^\mathcal{I}}\}$ be a set of interventions. Let $\Xi = \{\xi_k, k \in [n^\mathcal{I}]\}$ be $n^\mathcal{I}$ nodes corresponding to the $n^\mathcal{I}$ interventions. An extended f-DAG is defined as an f-DAG $D^\mathcal{I} = (V \cup \Xi, F, E \cup E^\mathcal{I})$ where $E^\mathcal{I} \subseteq \{(\xi_k, l) : l \in F\}$, i.e. set of edges from intervention nodes to factors.*

The SPN-FG model can be directly extended to include unknown interventions as additional nodes. Construction of an extended f-DAG is exactly the same as that of a normal f-DAG, except that we add edges from intervention nodes to non-intervention nodes based on their causal relations (**Fig. 4.2**, dashed lines). We introduce a binary intervention-to-factor matrix $\boldsymbol{W}$ to represent the causal relation from interventions to downstream nodes (**Fig. 4.3**c, dashed lines).

4.4.2 Bayesian Causal Discovery of DAGs

Let $\boldsymbol{\mathcal{G}}$ be the set of all DAGs. Consider a generative process where a DAG is first sampled from a prior, $p(G)$ with support on $\boldsymbol{\mathcal{G}}$, and a generative model $p^{(k)}(\boldsymbol{X}|G)$ under the k-th intervention. Given empirical observations, we can obtain a MAP estimate of the causal graph as $G^* = \arg\max_{G \in \boldsymbol{\mathcal{G}}} \sum_{k=1}^{n^\mathcal{I}} p^{(k)}(G|\boldsymbol{X})$.

Because $|\boldsymbol{\mathcal{G}}|$ is super-exponential in n[141], searching through the discrete space is computationally inefficient for large n. Instead, we resort to continuous optimization. As the true posterior is often intractable, we apply variational Bayes using a parameterized variational distribution $q(G; \boldsymbol{\Lambda})$. In this way, we are able to find G^* by optimizing a KL divergence: $G^* = \arg\min_{G \in \boldsymbol{\mathcal{G}}} \sum_{k=1}^{n^\mathcal{I}} KL(q(G; \boldsymbol{\Lambda}) || p^{(k)}(G|\boldsymbol{X}, I_k))$. In fact, we can minimize the KL-divergence by maximizing the evidence lower bound (ELBO),

$$
\begin{aligned}
q^*(G) = \arg\max_{q(G;\boldsymbol{\Lambda})} & \sum_{k=1}^{n^\mathcal{I}} \mathbb{E}_{p^{(k)}(\boldsymbol{X}|G^*)} \left[\mathbb{E}_{q(G;\boldsymbol{\Lambda})} \left[\log p_\Theta^{(k)}(\boldsymbol{X}|G) \right] \right] \\
& - KL\left(q(G; \boldsymbol{\Lambda}) || p(G)\right). \qquad (4.7)
\end{aligned}
$$

This ELBO objective is directly connected to autoencoding variational Bayes[171]. A slight difference here is the data distribution. Because we consider the data being generated by a single true causal graph, the data we sample are essentially biased and the expectation is over $p^{(k)}(\boldsymbol{X}|G^*)$ instead of $p^{(k)}(\boldsymbol{X})$. The optimal solution to this optimization problem is $q^*(G) = p_\Theta^{(k)}(G|\boldsymbol{X})$, assuming enough capacity of the variational family.

As mentioned in section 4.3.2, we can narrow down the search space by considering f-DAGs as a reasonable low-rank approximation of the true causal DAG. Although we propose SPN-FG as the major graph model in this work, the Bayesian framework is general to all

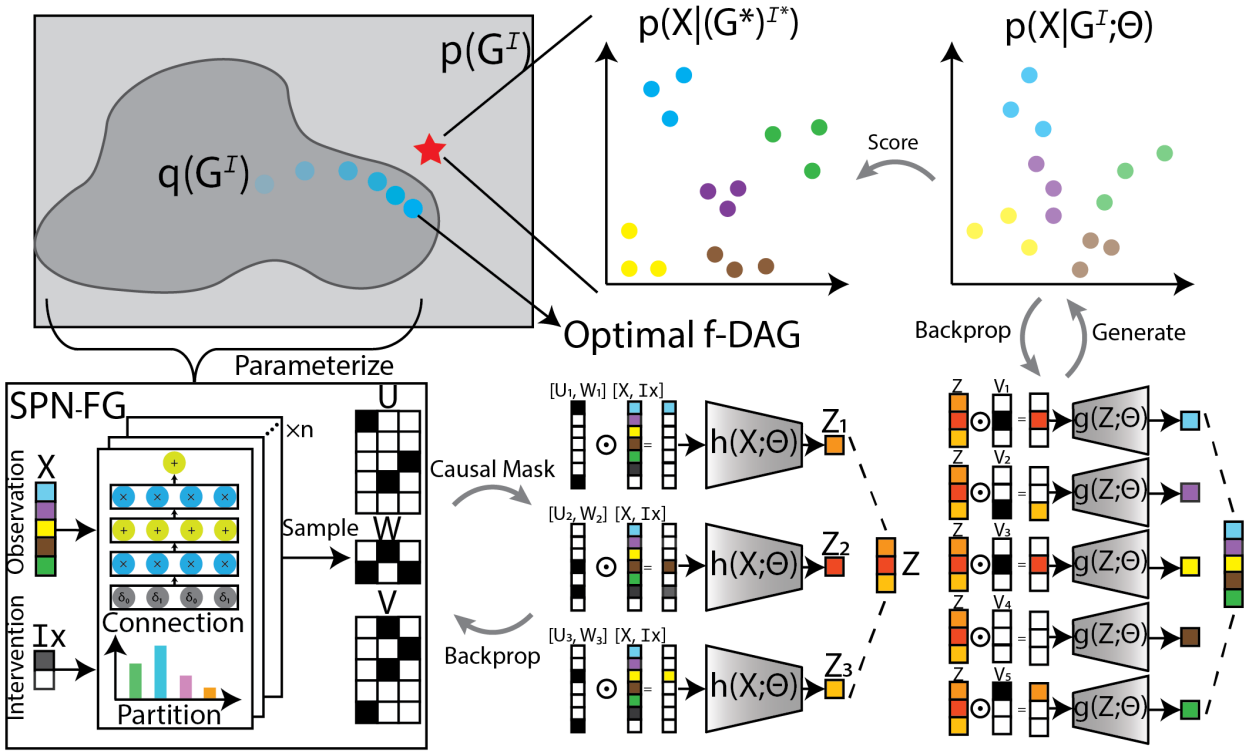

Figure 4.4: **Overview of ABCDEFG. Top Left:** Bayesian framework. A prior p with a support of all DAGs and a variational distribution with a support of f-DAGs. The red star represents the ground-truth DAG and light blue dots with increasing transparency show an optimization process w.r.t. the variational distribution. **Top Right:** Real vs. generated data distribution. **Bottom:** ABCDEFG model architecture. Binary matrices $\mathbf{U}, \mathbf{V}, \mathbf{W}$ are sampled from a parametric f-DAG model such as SPN-FG. Next, observations are masked by sampled causal relations (under Hadamard product, $\odot$) and fed to a VAE model fitting data distribution. Arrows show direction of data flow and back propagation.

parametric DAG models.

4.4.3 Amortized Bayesian Causal Discovery of Extended Factor Graphs

Model Description With the problem setup in **Section 4.4.2**, we now formally introduce our method, Amortized Bayesian Causal Discovery of Extended Factor Graphs (ABCDEFG). Given a set of random variables $\boldsymbol{X} = \{X_i : i \in [n]\}$ generated via a causal graph G^*, we apply a Bayesian method by estimating $p(G|\boldsymbol{X}, I^*)$ via optimization as described in **Section**

4.4.2:

$$q^*(G) = \arg\min_{G \in \boldsymbol{\mathcal{G}}} \sum_{k=1}^{n^{\mathcal{I}^*}} KL(q^{(k)}(G; \boldsymbol{\Lambda}) || p^{(k)}(G | \boldsymbol{X}, I_k)).$$

From our derivation (**Supplemental Section C.2**), the objective is equivalent to maximizing the ELBO (**Eq. 4.7**). The key to convert discrete search into continuous optimization is to create a differentiable parametric model for DAGs and estimate the ELBO using Monte Carlo sampling. In this work, we assume the true causal graph is or can be approximated by an f-DAG. Thus, we apply either the canonical (independent Bernoulli sampled by Gumbel softmax) or the SPN-FG model to parameterize $q(G; \boldsymbol{\Lambda})$.

The model architecture (bottom panel of **Fig. 4.4**) consists of an SPN-FG model for f-DAGs and a VAE for data distribution fitting. We sample f-DAGs from an SPN-FG model. The output is a node-to-factor matrix $\mathbf{U}$ and a factor-to-node matrix $\mathbf{V}$. Next, we model the data distribution under the causal graph as $p(\boldsymbol{X}) = \int_{\boldsymbol{Z}} \prod_{i=1}^{n} f(Z_j | \boldsymbol{X}_{\boldsymbol{\pi_j}}) g(X_i | \boldsymbol{Z}_{\boldsymbol{\pi_i}}) d\boldsymbol{Z}$. Here, $\boldsymbol{\pi_i}$ and $\boldsymbol{\pi_j}$ are the parent nodes and factors in the f-DAG. Instead of using separate encoding and decoding functions to obtain the posterior of each Z_j and conditional likelihood of each X_i, we follow the practice of Lopez et al.[151] and amortize all conditional distributions into a single encoding and decoding feed-forward neural network. Causal relations are injected into the VAE via masking operations $\mathbf{U_j} \odot \boldsymbol{X}$ and $\mathbf{V_i} \odot \boldsymbol{Z}$, where $\odot$ denotes the Hadamard product.

When the intervention targets are unknown, the causal discovery problem can be treated as recovering an extended f-DAG where each intervention corresponds to an intervention node. Equivalently, our SPN-FG model can be extended to include an intervention-to-factor matrix $\mathbf{W} \in \{0, 1\}^{k \times m}$. The causal mask operation becomes $[\mathbf{U_j} \odot \boldsymbol{X}; \mathbf{W_j} \odot \boldsymbol{I}]$ where $\boldsymbol{I}$ is a one-hot encoding of the intervention. We can apply the same optimization approach to jointly infer the causal graph and intervention targets. An additional advantage of extended f-DAG is including additional intervention information such as the dosage of a chemical treatment and combinations of interventions.

Identifiability Previous score-based methods have derived identifiability guarantees, such as identifying the true causal graph up to an equivalence class, whereas Bayesian approaches have largely focused on statistical inference and not investigated identifiability. Because ABCDEFG combines aspects of Bayesian and score-based methods, we seek to connect these approaches by investigating identifiability. We obtain theoretical guarantees for ABCDEFG by relating our approximate posterior distribution with the score function optimized by previous methods. Based on previous theory (**Supplementary Section C.2.1**), we derive

new results about the identifiability of f-DAGs.

Theorem 4.9 (Identifiability via ELBO maximization)**.** *Let $\boldsymbol{X} = \{X_1, \ldots, X_n\}$ be a set of causally related random variables with a causal DAG G^* and $\mathcal{I}^* = \{I_k : k \in [n^{\mathcal{I}}]\}$ be a set of interventions with $I_1 = \emptyset$. Let $\boldsymbol{\mathcal{G}}$ be a subset of all causal DAGs and $q^*(G)$ be an optimal graph distribution from the optimization problem:*

$$\sup_{q(G;\boldsymbol{\Lambda}):supp(q)\subseteq\boldsymbol{\mathcal{G}}} \boldsymbol{\mathcal{L}}(q(G;\boldsymbol{\Lambda}))$$

where

$$\boldsymbol{\mathcal{L}}(q(G;\boldsymbol{\Lambda})) = \mathbb{E}_{q(G;\boldsymbol{\Lambda})}\left[S_{\mathcal{I}^*}(G)\right] - \beta KL(q(G;\boldsymbol{\Lambda})||p(G))$$
$$S_{\mathcal{I}^*}(G) = \sup_{\boldsymbol{\Phi}} \sum_{k=1}^{n^{\mathcal{I}^*}} \mathbb{E}_{p^{(k)}(\boldsymbol{X})}\left[\log f^{(k)}(\boldsymbol{X}|G;\boldsymbol{\Phi})\right] - \lambda|G|$$

If $G^* \in \boldsymbol{\mathcal{G}}$, then, under the same assumptions as in Thm. C.12[149], namely sufficient capacity, $\mathcal{I}$-faithfulness, positivity and finite entropy, for small enough $\beta > 0$ and $\lambda > 0$, $\hat{G} = \arg\max_G q^*(G)$ is $\mathcal{I}^*$-Markov equivalent to G^*.

We present a proof sketch here. Please refer to **Supplemental Section C.2.2, C.2.3** for details and a related proof of $\mathcal{I}^*$-Markov equivalence for interventions with unknown targets.

Proof. We prove this theorem by contradiction. Suppose $\exists \hat{G} = \arg\max_G q^*(G)$ that is not $\mathcal{I}^*$-Markov equivalent to G^*. We can create another distribution q' such that $q'(\hat{G}) - q^*(\hat{G}) = q^*(G^*) - q'(G^*) = \epsilon > 0$ and for any other graph G, $q'(G) = q^*(G)$. From algebraic calculation, we have

$$\boldsymbol{\mathcal{L}}(q') - \boldsymbol{\mathcal{L}}(q^*) = \epsilon\left(S_{\mathcal{I}^*}(G^*) - S_{\mathcal{I}^*}(\hat{G})\right) + \beta\Delta.$$

Because $S_{\mathcal{I}^*}(G^*) - S_{\mathcal{I}}(\hat{G}) > 0$, $\exists \beta > 0$ such that $\boldsymbol{\mathcal{L}}(q') - \boldsymbol{\mathcal{L}}(q^*) > 0$. Then, we have a contradiction about q^* being an optimal solution to the optimization problem. □

Furthermore, when the true causal graph can be factorized as an f-DAG, our method is able to identify the f-DAG with additional assumptions. Denote $par(\cdot; D)$ and $chd(\cdot; D)$ as the set of parent and child nodes in any graph D.

Definition 4.10. *Let $G = (V, E)$ be any graph, $\forall f \in V$, the set of unique parents and children of f are defined as $P_f(G) := \{i : i \in par(f; G), chd(i; G) = \{f\}\}$ and $C_f(G) := \{j : j \in chd(f; G), par(j; G) = \{f\}\}$.*

With the above definition, we define a subset of f-DAGs:

Definition 4.11. *Let $\mathcal{I}$ be a set of interventions. $\boldsymbol{\mathcal{D}}_m \subseteq \{(V, F, E) : |F| = m\}$ is defined as the set of f-DAGs with m factors and the following properties:*

1. *$\forall f \in F$, $P_f(D) \neq \emptyset$ and $C_f(D) \neq \emptyset$.*
2. *$\forall f_1, f_2 \in F$ such that $f_1 \neq f_2, f_1 \to f_2$ in $(D^{\mathcal{I}})^2[F]$, $|P_{f_1}(D^{\mathcal{I}})| > 1$ (inclusively) or $|P_{f_2}(D^{\mathcal{I}})| > 1$.*
3. *$\forall f \in F$, $|par(f; D^{\mathcal{I}})| > 1$, there is at most one factor $g \in par(f; (D^{\mathcal{I}})^2[F])$ such that $|par(g; D)| = 1$.*

Intuitively, these properties ensure that factors in the f-DAG have enough distinct upstream and downstream nodes to uniquely identify them. The first property means factors must be functional–passing causal relations from source to target and have a unique parent and child to be distinguished from others. The second and third conditions set a lower limit of v-structures and avoid paths with consecutive non-collider factors. Based on the definition, we further define the resulting half-square graphs.

Definition 4.12. *Given a set of f-DAGs $\boldsymbol{\mathcal{D}}_m$ defined as in Def. 4.11, $\boldsymbol{\mathcal{G}}_m$ is defined as the set of all DAGs having a rank-m f-DAG factorization:*

$$\boldsymbol{\mathcal{G}}_m := \{G : \exists D \in \boldsymbol{\mathcal{D}}_m, G = D^2[V]\}.$$

With Def. 4.11 and 4.12, we have the following lemma.

Lemma 4.13. *Let $D_1 = (V, F, E_1)$ and $D_2 = (V, F, E_2)$ be two f-DAGs on the same set of nodes, V, and factors, F, and $\mathcal{I}$ be a set of interventions. Let $\Xi = \{\xi_k : k \in [n^{\mathcal{I}}]\}$ be intervention nodes. In addition, suppose $D_1, D_2 \in \boldsymbol{\mathcal{D}}_m$ defined as in Def. 4.11. Then, under a permutation of factors, we have $D_1 \simeq_{\mathcal{I}} D_2 \iff D_1^2[V] \simeq_{\mathcal{I}} D_2^2[V]$.*

We present a proof sketch here and include all details in **Supplemental Section C.2.4**.

Proof. By Theorem C.9[145], we convert the proof of $\mathcal{I}$-Markov equivalence to proof of equivalent graph structure. The general strategy is proof by contradiction under a discussion of different graph structures. We first show the forward direction. Suppose $D_1 \simeq_{\mathcal{I}} D_2$ but $D_1^2[V] \not\simeq_{\mathcal{I}} D_2^2[V]$. There must be a mismatch in skeleton or v-structure. The former case implies a mismatch in skeleton between f-DAGs, while the latter implies a v-structure mismatch between f-DAGs. Hence, we have contradiction in both cases.

For the reverse direction, the main idea is to show that there is a bijection $\phi : F \to F$ such that $par(f; D_1) = par(\phi(f); D_2)$ and $chd(f; D_1) = chd(\phi(f); D_2)$. In fact, we partition F into three subsets: (1) $F_1 := \{f : f \in F, |par(f; D)| = |chd(f; D)| = 1\}$, (2) $F_2 := \{f : f \in F, |par(f; D)| = 1, |chd(f; D)| > 1\}$ and (3) $F_3 : \{f : f \in F, |par(f; D)| > 1$ and prove a bijection between each set of factors. We introduced Proposition C.20-C.22 for the proof of each partition and apply proof by contradiction. Finally, bijection implies all undirected edge connections and v-structures should match between D_1 and D_2. □

Finally, we conclude our theory with a theorem on the identifiability of f-DAGs.

Theorem 4.14 (Identifiability of the f-DAG via ELBO maximization)**.** *Let $\{X_i : i \in [n]\}$ be a set of causally related random variables whose joint distribution follows a causal graphical model with a causal DAG G^* and $\mathcal{I}^* = \{I_k : k \in [n^{\mathcal{I}^*}]\}$ be a set of interventions where $I_1 = \emptyset$. Suppose $G^* = (D^*)^2[V] \in \boldsymbol{\mathcal{G}}_m$. Let $q(G; \boldsymbol{\Lambda})$ be a variational distribution on $\boldsymbol{\mathcal{G}}_m$ and q^* be a PMF on $\boldsymbol{\mathcal{G}}_m$ maximizing the ELBO objective as in theorem 4.9. Then, under the same assumptions as in Theorem C.12[149], namely sufficient capacity, $\mathcal{I}$-faithfulness, positivity and finite entropy, and for sufficiently small $\lambda > 0$ and $\beta > 0$, $\hat{D} = \arg\max_D q^*(D^2[V]; \boldsymbol{\Lambda})$ is $\mathcal{I}^*$-Markov equivalent to D^* under a permutation of factors.*

The theorem is a direct result of theorem 4.9 and lemma C.19. By Theorem 4.9, $\hat{G} \simeq_{\mathcal{I}^*} G^*$. Since $\hat{G}, G^* \in \boldsymbol{\mathcal{G}}_m$, $\exists \hat{D}, D^*$ such that $\hat{G} = \hat{D}^2[V]$ and $G^* = (D^*)^2[V]$. By lemma 4.13, $\hat{D} \simeq_{\mathcal{I}^*} D^*$ under a permutation of factors.

4.5 Experiments

4.5.1 Simulation Study

We next evaluated our algorithm on simulated data. We simulated data largely based on the approach of a previous study[151], but we extended their simulations in a few ways. First, we found that the simulated graphs often had many disconnected nodes (no incoming or outgoing edges), so we ensured at least 75% of nodes were connected. Second, we explored the effects of correlations between edge probabilities, which our approach explicitly models but previous approaches do not, by constructing an SPN and then sampling from the joint distribution of edges. Third, we simulated interventions with unknown targets. Finally, we simulated both hard and soft interventions. Additional results and simulation details are included **Supplemental Section C.3.1, C.3.2**. To evaluate our method, we benchmarked ABCDEFG on 24 datasets and compared with four existing SOTA methods: DCDI[149], DCDFG[151], ENCO[154] and SDCD[153]. The 24 datasets include eight types of SEMs

– a combination of (1) linear vs. non-linear causal effects, (2) independent vs. jointly distributed edge probabilities, and (3) hard vs. soft interventions. Each simulated graph includes 100 nodes and 10 factors. We simulated three separate graphs for each type of SEM. Similar to previous studies, we report Structural Hamming Distance (SHD), precision, and recall of edge prediction. We used consistent hyperparameter setting for ABCDEFG across all simulations (**Supplemental Section C.3.3**) and managed to train ABCDEFG on all datasets with moderate time and memory consumption (**Supplemental Section C.3.4**). ABCDEFG performed on par with SOTA methods on linear SEMs, though SDCD showed consistently strong performance in the linear case (**Fig. 4.5a, b, e, f**). ABCDEFG significantly outperformed all other approaches in graphs with nonlinear causal effects and edge probabilities that are jointly distributed and sampled from an SPN (**Fig. 4.5b, f**). We note that ABCDEFG performed significantly better than other methods on nonlinear SEM with soft interventions and the true causal graph sampled from an SPN-FG model with correlated edges (**Fig. 4.5a, b**, diamond and cross markers).

We next evaluated how ABCDEFG performs using interventions with unknown targets, a key advantage of our approach. To test target identification, we generated causal graphs but did not use the intervention target information during inference. SDCD, ENCO, and DCDFG cannot incorporate interventions with unknown targets. Although DCDI can in principle identify both causal relations and unknown intervention targets, it can only infer targets of hard (perfect) interventions but not soft (imperfect) ones. Thus, we ran DCDI on twelve datasets with hard interventions and unknown targets and compared it to ABCDEFG. In addition to SHD, precision, and recall of the causal graph, we evaluated the accuracy of the intervention-to-node graph (**Fig. 4.5 c, d, g, h**). Overall, the accuracy of inferred node-to-node relationships was lower here compared to interventions with known targets, indicating that causal inference is more challenging under unknown interventions. ABCDEFG achieved better node-to-node causal relation inference then DCDI on all four simulation types. We also observed that ABCDEFG with SPN-FG performed better than the basic model when the true causal graph is sampled from an SPN-FG model and a similar result hold for the basic model. ABCDEFG inferred the intervention targets more accurately than the node-to-node causal relationships and achieved relatively high precision and recall, particularly for SPN-simulated graphs. Noticeably, in both linear and non-linear SEMs with SPN-simulated graphs and soft interventions, ABCDEFG achieved over 80% precision and recall.

Finally, we investigated DAG constraints. ABCDEFG is free of any DAG constraint because the basic and SPN-FG models both are guaranteed to generate DAGs, while other methods binarize and prune weighted adjacency matrices to produce DAGs. We took the binarized adjacency matrices without pruning and found that ABCDEFG was the only

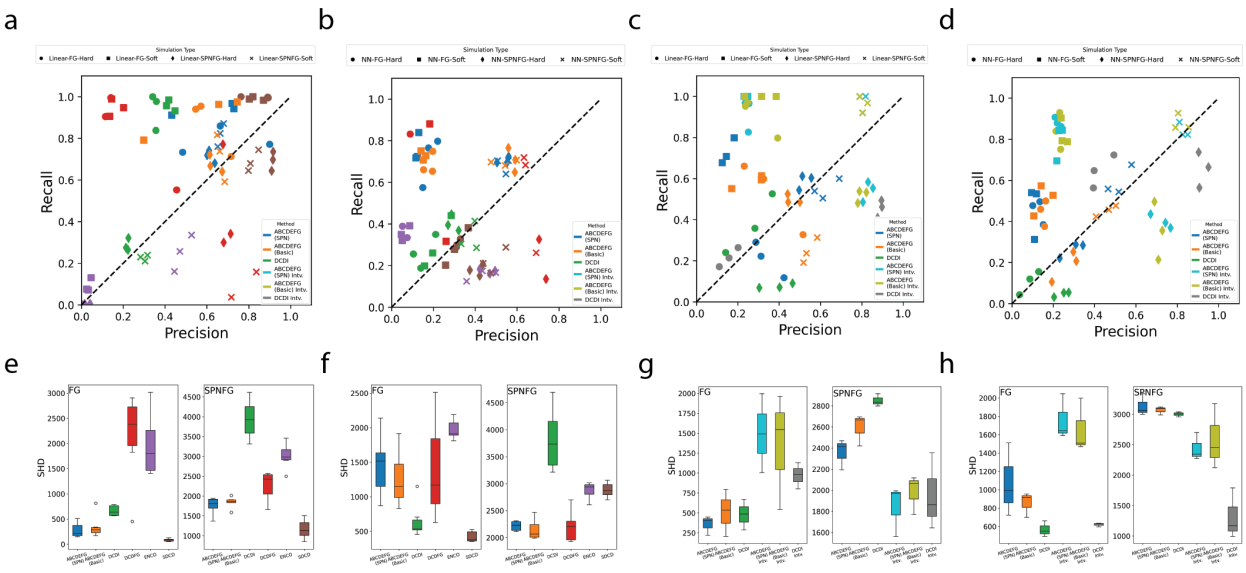

Figure 4.5: **Benchmarking Results on Interventional Data.** The scatter plots show precision vs. recall on datasets with **(a)** linear targeted, **(b)** non-linear targeted, **(c)** linear untargeted and **(d)** non-linear untargeted SEMs. Markers represent simulation types and colors represent methods. The box plots show the SHD on datasets with **(e)** linear targeted, **(f)** non-linear targeted, **(g)** linear untargeted and **(h)** non-linear untargeted SEMs. Methods starting with "Intv." refer to results on intervention target identification.

method generating DAGs in all 24 simulated datasets (**Fig. 4.6a**). Notice that DCDFG, which also assumes a low-rank causal graph, is the second-best method in terms of fulfilling the DAG constraint. To further investigate these inferred causal graphs, we evaluated a acyclicity distance, defined as the minimum number of edge deletion operations to convert a directed graph into an acyclic one. Because exact calculation of the distance is NP-hard[231], we resort to existing works that give an upper and symptotic lower bound[232, 233] and devised an algorithm to calculate an exact though loose lower bound (**Supplemental Section C.4**). As **Fig. 4.6b** shows, unpruned graphs from DCDI, ENCO and SDCD all have many loops, as high as hundreds, while ABCDEFG is completely free of loops. Thus, we conclude that ABCDEFG allows inference of true DAGs.

4.5.2 Application to Real Cellular Perturbation Screen.

We applied our model to data from a large-scale perturbation screen in which cells were treated with 46 combinations of 14 growth factors[51]. Growth factors are biomolecules that induce significant molecular changes through signaling pathways and are used to steer cells toward desired cell types in the dish. Though some downstream targets of growth factors are known, the targets are highly context-specific and largely unknown. The raw data contains

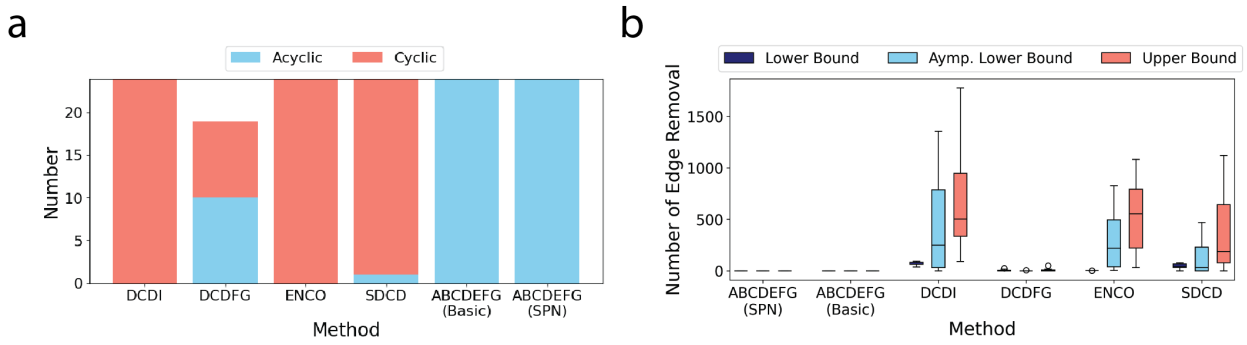

Figure 4.6: **Evaluation of the DAG Constraint. (a)** Bar plot showing the proportion of acyclic vs. cyclic graphs inferred by each method. **(b)** Box plot showing upper and lower bound of minimum number of edge deletion operations to convert cyclic graphs into acyclic ones.

gene expression counts for 34,469 genes in 31,475 cells. Following standard preprocessing steps for this type of data, we extracted the 1,000 most highly variable genes for causal graph inference (**Supplemental Section C.5**). We used 10 factors in our model. The total number of edges recovered from the gene-to-gene half-graph is 99,167. The number of edges from perturbations to factors (from the intervention-to-factor matrix $\boldsymbol{W}$) ranges from 1 (IWP2) to 7 (BMP4, FGF8 and SAG). The average number of incoming and outgoing edges across factors is 63 and 122, respectively. There is no ground truth available for the gene-gene and intervention-gene relationships in this type of real data. So to test whether our model could reveal biological insights, we did similar assessments described by Lopez et al.[151] in the context of interventions with known targets. From the graph recovered by ABCDEFG, we selected five growth factors (BMP4, BMP7, Chiron, FGF2, and FGF4) and visualized their targets (**Fig. 4.7**).

We conducted gene ontology enrichment analysis to investigate the functions of the genes upstream and downstream of each factor. Reassuringly, the growth factors have causal edges to factors whose connected genes play a role in the respective signaling pathways. For example, FGF2 and FGF4 are upstream of factor one, whose genes are enriched in terms that include response to fibroblast growth factor. Similarly, BMP4 and BMP7 are upstream of factor two, whose genes are enriched for response to BMP. Furthermore, growth factors within the same family connect to similar factors: BMP4 and BMP7 both have connections with factor one, two, three and nine, while FGF4 and FGF7 are both connected with factor one, two, three and five. We also found that other factors are enriched in terms like neuron differentiation, forebrain development, and chemical synaptic transmission, which relate to neural system development. We also inspected the individual upstream and

downstream genes identified by the model. Specifically, our model identified upstream genes like the transcription factor HES1, EBF3, EOMES and LMX1A, and the signaling pathway inhibitor TFPI. These genes either initialize or inhibit the initialization of biological processes or pathways. In contrast, many of the downstream genes in our graph encode proteins which serve as functional units at the end of biological pathways: NEFM and MBP encode structural proteins of brain cells, and SAT1 and ADCY2 are key enzymes.

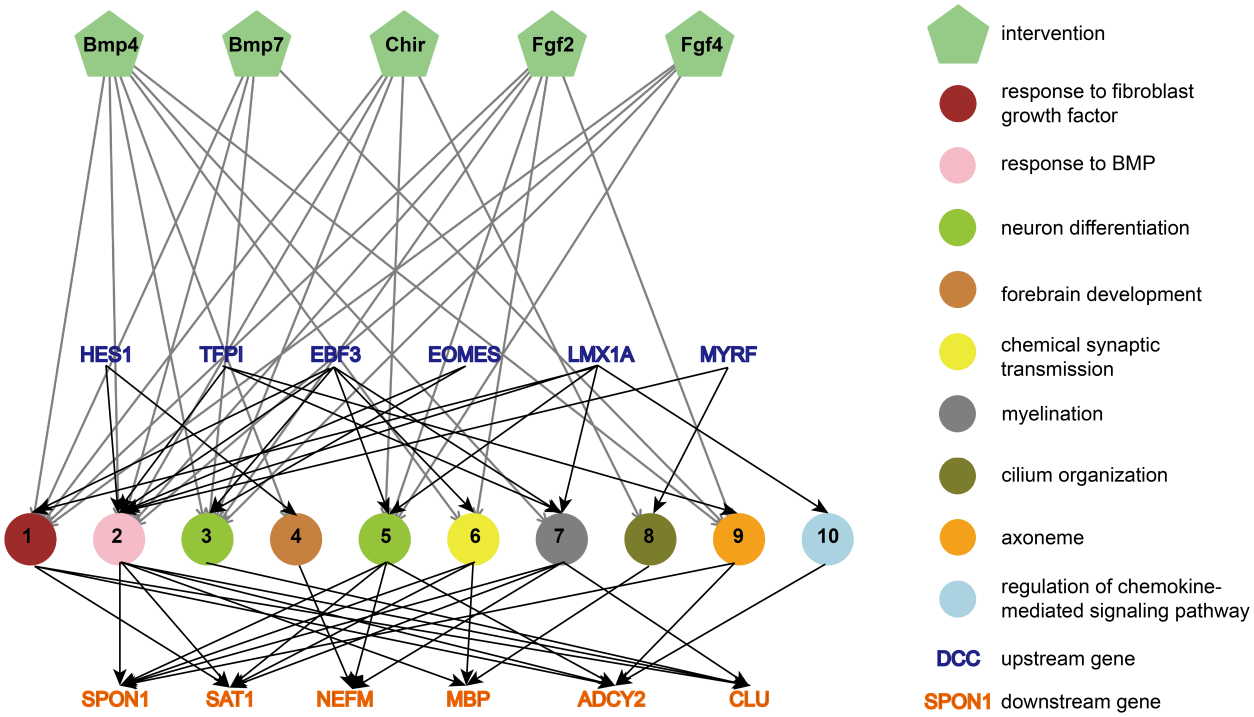

Figure 4.7: **Extended Factor Graph Inferred from Growth Factor Screen.** Inferred causal edges among interventions with unknown targets (growth factors; pentagons), factors (circles), and genes (text) are shown. Factor colors indicate gene ontology terms enriched in the upstream (blue) and downstream (orange) genes. Edges from interventions to factors are shown in gray arrows, and edges between genes and factors are shown in black arrows.

4.6 Discussion

In this chapter, we presented ABCDEFG, a differentiable Bayesian method for causal discovery from large-scale, high-dimensional interventional data. We unified theoretical results from two lines of work — score-based differentiable methods and Bayesian methods — by offering a Bayesian view of the identifiability theorem and extending it to factor DAGs. At the algorithmic level, we designed a generative process for DAGs without calculating a surrogate DAG constraint function and developed a new parametric model, SPN-FG, which is

both expressive and scalable to large systems. We also introduced the extended factor DAG to model interventions with unknown targets and demonstrated its application to chemical target identification from single-cell perturbation data.

ABCDEFG achieved top performance on various simulation datasets, especially those with soft interventions. Moreover, it is the only method that can infer interventional targets given soft perturbations with unknown targets. Although SOTA methods such as SDCD outperformed ABCDEFG on certain types of simulated data, we believe our method has more practical applications in gene regulatory network discovery with large-scale single-cell perturbation data. Of course, our basic assumptions are still simplified compared to real cases. For example, the low-rank assumption may not hold in practice, and the number of factors is still empirically determined with no general guidance. The identifiability guarantee is also subject to additional assumptions and applies only in the infinite data regime. In addition, our method does not incorporate information about chemical perturbations, such as molecular properties. We expect future work to address these problems.

CHAPTER 5

Summary and Future Work

5.1 Summary of the Dissertation

The cell is such a mesmerizing yet complex subject to study. For hundreds of years, humans have only discovered the tip of the iceberg. This is largely due to the lack of measurement technologies capable of profiling cells in detail. As a vital product of cellular activity, messenger RNA (mRNA) is a key molecule in understanding cell fate decisions. It serves as an intermediary between DNA and proteins, reflecting the dynamic state of cellular processes. Single-cell RNA sequencing (scRNA-seq), along with spatial transcriptomics and genetic or chemical screening, offers valuable quantitative measures that allow researchers to precisely study the behavior and function of individual cells during development. These technologies provide a high-resolution view of the transcriptome, revealing the heterogeneity of cell populations and their unique responses to environmental signals. By integrating scRNA-seq with spatial transcriptomics, scientists can also map gene expression to specific locations within tissues, uncovering spatial patterns that define cellular interactions and tissue organization. Furthermore, genetic and chemical screening complement these approaches by enabling the identification of key regulators and pathways involved in cellular processes. Collectively, these advanced tools facilitate a comprehensive understanding of cellular dynamics, enabling breakthroughs in developmental biology, disease research, and therapeutic development.

The advent of new computational methods and machine learning technologies is revolutionizing our understanding of gene expression mechanisms. As single-cell RNA sequencing and spatial transcriptomics generate vast amounts of complex data, these advanced computational approaches become essential for extracting meaningful insights. Statistical methods generate reliable results for hypothesis testing, while machine learning algorithms can identify patterns and relationships in gene expression data that would be impossible to discern manually, enabling the discovery of cellular dynamics and regulatory relationships. One of

the key advantages of machine learning is its ability to handle high-dimensional datasets, accurately clustering cells into distinct types and states based on their gene expression profiles. This capability is crucial for unraveling cellular heterogeneity and understanding how different cell types contribute to development and disease.

As we delve deeper, we realize that single-cell transcriptomic data contain richer information than just cellular heterogeneity. Understanding how cells differ is just the first step, and a fundamental understanding requires discovery of underlying mechanisms. In addition, cellular identity information is static, but cell development is a spatiotemporal process. Thus, it is important to extract dynamical information from massive amounts of data.

This dissertation presents a series of works aimed at computationally aiding the scientific discovery of gene expression dynamics from single-cell transcriptomic data. We first examined the RNA velocity inference problem, a significant topic in computational biology because it focuses on uncovering hidden dynamic information in the data. We establish gene expression as a temporal process and treat the problem of RNA velocity inference as a latent variable problem. To uncover latent cell developmental time and mRNA transcription, splicing, and degradation rates from single-cell RNA sequencing data, we propose a Bayesian method, VeloVAE, and improve it with a variety of extensions. We designed a benchmark study with both quantitative and qualitative evaluations and compared VeloVAE with state-of-the-art (SOTA) methods. Our method overcomes the limitations of existing RNA velocity methods, including providing a unified cell developmental time, modeling early induction and late repression of gene expression, capturing branching effects in multi-lineage systems, and handling discrete counts. The application of VeloVAE to complex, large-scale datasets, such as the mouse brain and the whole embryo, reveals cell differentiation trajectories.

Temporal information is important, but it is only a piece of the puzzle. Cell development is a spatiotemporal process, so the spatial aspect also has significant implications for cell development. With the advancement of spatial transcriptomics, single-cell RNA sequencing with spatial information has become available. Thus, we further investigated the spatial mechanisms of gene expression. In particular, we focus on the problem of jointly modeling RNA splicing and spatial dynamics by using a spatially coupled RNA velocity model. Based on this framework, we developed TopoVelo, a variational graph autoencoder model, to perform Bayesian estimation of RNA velocity and predict cell velocity in physical space from spatial transcriptomic data. We validated our method on both simulated and real datasets. Remarkably, our predicted spatial cell velocity in mouse brain tissue closely aligns with previous quantitative measurements of neuron cell migration velocity. In addition, we were the first to apply TopoVelo to a 3D developing mouse neural tube and successfully annotated three closure points, consistent with findings reported in previous literature. Finally, we

analyzed human embryoid data cultivated by our wet lab colleagues. We applied TopoVelo to identify the cell migration path from pluripotent stem cells to more differentiated cells and observed clear morphological changes during this process. Through these efforts, we demonstrated the existence of a spatial mechanism underlying gene expression dynamics: cells differentiate and interact as they migrate. Spatial interactions influence the speed of gene expression and contribute to cell fate decisions.

Last but not least, gene expression is also controlled by a causal mechanism known as the gene regulatory network (GRN). This implies that the current RNA velocity model, even with both temporal and spatial mechanisms, is insufficient. To understand the causal mechanism, we separately consider this problem as an instance of a machine learning problem known as causal discovery. Classical methods based on statistical testing and discrete graph searching are not directly applicable to high-dimensional, large-scale data. However, with single-cell perturbation data available, we can identify the causal graph with much higher accuracy. Thus, we developed a new method, Amortized Bayesian Causal Discovery of Extended Factor Graph (ABCDEFG). Our method combines the advantages of Bayesian methods and score-based differentiable methods and is guaranteed to identify the causal graph up to a smaller equivalence class. In addition, we designed a probabilistic model for directed acyclic graphs that guarantees acyclicity by construction and proposed the sum-product network as a generative model for graphs. We demonstrated their efficacy on both simulated and real datasets. Furthermore, we extended the factor DAG model to include soft and unknown interventions. This is particularly important for biological discovery because many experiments involve chemical screening, which does not directly target specific genes. For gene causal relation discovery, our method offers the advantage of modeling gene pathways via a factor graph and the capability of identifying unknown target genes under chemical perturbations.

Overall, this dissertation presents three distinct yet connected projects that consistently utilize a variational Bayesian framework. Through the careful design of deep generative models, we incorporate both statistical power and biological mechanisms into these models. This design choice further allows us to combine the temporal, spatial, and causal mechanisms of gene expression in a unified approach in the future, leading to a more accurate and fundamental understanding of cell development.

5.2 Contribution and Broader Impact

This dissertation aims to apply machine learning techniques to facilitate scientific discovery in computational biology. Compared with existing research, our work contributes to this

field in several key aspects.

First, we address new types of computational problems that are essential for understanding cell fate transitions and have emerged from recent advancements in sequencing technology. The proposed Bayesian framework is sufficiently general to uncover diverse hidden information about complex dynamical systems from static snapshot data.

Next, we develop several variational autoencoder (VAE) models within this framework and apply them to extract the temporal, spatial, and causal mechanisms underlying gene expression. Specifically, we strive to integrate the strengths of two approaches: (1) the interpretability and domain knowledge provided by mathematical modeling, and (2) the versatility and expressiveness of deep learning. My research offers new insights into designing deep generative models that better serve scientific purposes.

Finally, this dissertation has the potential to make an impact on the scientific community. By applying computational methods to real single-cell transcriptomic data, we demonstrate how machine learning techniques, combined with domain knowledge, can uncover new biological insights or validate existing findings. While current computational methods remain far from solving many challenging problems in cell biology, this work takes a step toward bridging the gap and advancing large-scale scientific discovery. We have developed three open-source software packages for the scientific community and sincerely hope that they will contribute to a deeper understanding of cellular systems.

5.3 Future Work

Our work lies at the intersection of machine learning and computational biology. While our methods have demonstrated strong performance and provided valuable biological insights into gene expression mechanisms, we are well aware of their limitations and have identified several directions for future work.

Temporal Dynamics with RNA Velocity Current RNA velocity methods are limited in a number of ways. First, the current ODE model for RNA splicing does not capture arbitrary dynamics, particularly nonlinear dynamics. In fact, most methods treat gene expression in all cells as samples from a linear system. While VeloVAE models complex dynamics by considering a mixture of ODEs, it is still an approximation of a more complex nonlinear system. Therefore, it is worthwhile to explore more realistic ODE models. Another promising direction is to consider the Laplace domain. Recently, a new type of deep learning technique called neural Laplace has provided a potential way to model temporal dynamics [234, 235]. Such approaches have the advantage of being once-for-all, as they allow us to

know the entire time domain given information in the Laplace domain.

Next, the discrete VeloVAE model takes a first step to modeling RNA velocity directly on raw count data. This is a more realistic model for gene expression since the production of RNA is a discrete system. Thus, it remains an interesting problem to extend current ODE formulation to a stochastic differential equation (SDE) or a diffusion process. We expect such approach can better account for the random nature of biochemical reactions.

RNA velocity only considers the biochemical reaction from DNA to RNA. However, gene expression occurs through a chain of biochemical reactions at different levels, including genetic, transcriptomic, epigenetic, and proteomic levels. Another important future direction is to extend RNA velocity to multiple modalities. In fact, some existing works have initiated this direction at the epigenetic[169] and proteomic[168] levels, but no method currently includes all modalities in a single run. We envision both a sequencing technology and a computational method that uncover the ultimate temporal mechanism of gene expression across all modalities.

On the other side of this field, we see incredibly rapid development in AI and machine learning. New methods continue to emerge, and we believe that some have the potential to influence future work on the RNA velocity problem. One interesting question is how to model stochastic biochemical reactions using diffusion models. A diffusion model[236, 7, 237, 238, 239] is a type of deep generative model inspired by the diffusion process of molecules. The forward and backward processes correspond to the noising and denoising of input data. To model stochastic RNA velocity, these processes need to be replaced by a stochastic process described by the chemical master equation. However, no algorithms have been developed to recover both the temporal information and reaction rates using such a diffusion process. We believe that a new type of deep generative model will emerge to study the stochastic process of gene expression and ultimately lead us to model a stochastic trajectory of cell differentiation.

Another major topic is foundation models. Given the success of large language models (LLMs), we are beginning to realize the power of transformers, powered by deep neural networks, to automatically extract relevant information and perform inference. A number of foundation models for single-cell transcriptomics have also emerged, based on massive parallel training on existing scRNA-seq datasets[240, 241, 242, 243]. However, none of these foundation models can mechanistically model the biochemical processes of DNA transcription, RNA splicing, degradation, epi-genetic modification, and translation. These mechanisms are critical to our understanding of real biology, so it remains an open question whether foundation models can learn these mechanisms from data. Of course, most current scRNA-seq data do not distinguish between unspliced and spliced RNA, but we envision that future work

can combine foundation models with RNA velocity, or even multimodal velocity models.

Cell Development in Space Uncovering the relationship between RNA splicing dynamics and cell development in space provides a pivotal path toward a fundamental understanding of disease. Many diseases manifest as tissue sections begin to exhibit unusual physical features, while the root of this morphological change lies in abnormalities in gene expression. TopoVelo allows us to connect the temporal dynamics of RNA splicing with the spatial development of cells. We can further improve the method in several ways. First, the ODE model in TopoVelo does not directly account for spatial information. One future direction is to include spatial variation in RNA molecule quantity and extend the time-domain ODE to a partial differential equation (PDE). Another important spatial context to incorporate into the spatial mechanism is cell-cell communication via ligand-receptor binding.

In addition, spatial transcriptomic data are often generated alongside a histological image of the tissue. In TopoVelo, we manually registered spatial coordinates to the image. Future methods could explore either a multimodal deep generative model or a transfer learning approach to connect RNA velocity with cell images. This direction has the potential to associate cell transcriptomic states with disease.

TopoVelo applies graph neural networks to learn spatially-aware cell representations. Since the focus is to understand the mechanism of RNA splicing dynamics and cell development in space, we considered only the two most popular graph learning techniques, GCN and GAT, and did not extensively investigate other graph learning techniques. Future work can create standardized benchmark datasets and evaluation metrics for graph representation learning methods. Scalability is another important subject of research, as TopoVelo does not apply any graph sampling techniques to handle large graphs with millions of nodes or dense graphs with hundreds of neighbors per cell.

Causal Discovery of Gene Regulatory Network In the fourth chapter, we propose SPN-FG, a new factor DAG model, and ABCDEFG, a variational Bayesian causal discovery method based on several previous works. We believe there are many promising directions for improving causal discovery methods. First, the current theory regarding the identifiability of differentiable methods is limited to the infinite-data regime and relies on many assumptions. We believe it is important to first investigate (1) identifiability results in finite-data regimes and (2) the validity of these assumptions in real scenarios, such as single-cell perturbation experiments. Next, most existing differentiable models do not handle cyclic graphs. It is of both theoretical and practical interest to develop differentiable methods for cyclic graphs with identifiability guarantees.

Another interesting direction is to explore the connection between causal relationships and attention. We observe that predicting gene expression from masked inputs in our VAE model is algebraically similar to self-supervised learning. Additionally, a probabilistic adjacency matrix representing a gene regulatory network (GRN) resembles attention scores. Thus, it would be of great interest for future studies to investigate these two methods, which stem from different applications. In fact, a previous work has initiated this direction[244]. With suffcient understanding of these two related machine learning methods, we could adapt transformers for large-scale causal discovery. More broadly, we would like to understand whether foundation models can (1) discover causal relationships from big data and (2) learn causally meaningful representations and make decisions based on logical rules. While existing chain-of-thought[245] techniques generate a sequence of language descriptions of the model's internal "thinking," they are primarily based on prompting. We would like to explore whether causal relationships can fundamentally change foundation model architectures, create a new training paradigm, and achieve better overall performance.

Multi-domain Extension While the three directions above represent future directions for individual research areas, the temporal, spatial, and causal mechanisms are not isolated in a more realistic cell model. Hence, it is of great importance to combine them. In terms of temporal and spatial domains, we can build upon TopoVelo and, as we mentioned earlier, consider both temporal and spatial variations in RNA splicing dynamics. Techniques such as neural operator learning and domain knowledge about PDEs and cell-cell communication can be incorporated. Another promising direction is to integrate the gene regulatory network (GRN) into the RNA velocity model. Currently, all RNA velocity methods assume independence among genes for simplicity. However, the existence of a gene regulatory network implies that this assumption is incorrect. The causal relationships between genes make the ODE model not gene-wise but rather a vector-form ODE: $\frac{d\boldsymbol{u}}{dt} = \boldsymbol{\alpha} - \mathbf{B}\boldsymbol{u}, \frac{d\boldsymbol{s}}{dt} = \mathbf{B}\boldsymbol{u} - \mathbf{C}\boldsymbol{s}$. Here, we show an example of a extension of the original linear ODE model[89]. α is a 1D vector of transcription rates, $\mathbf{B}$ and $\mathbf{C}$ are non-diagonal matrices with quantitative causal relations between any two genes. Jointly inferring all mechanisms from single-cell data is more challenging, but once achieved, it will provide a more accurate and comprehensive view of genetic activities during cell differentiation. With a sufficient understanding of gene expression mechanisms, we will be able to monitor cell states and take timely action to prevent or treat diseases. While RNA velocity describes the dynamic behavior of RNA splicing in cell populations, it is typically used to describe the natural process of cell development. A potential application of RNA velocity is to directly control gene expression so that we can precisely guide cell development toward an ideal state. This involves studying gene expres-

sion dynamics with actuation, which could potentially enable cell reprogramming and define a new type of cell therapy for future medical treatments.

5.4 Final Remarks

With rapid advancements in machine learning (ML) and artificial intelligence (AI), along with the explosion of large-scale single-cell transcriptomic data, scientists today are taking a different approach to discovering gene expression mechanisms. Although we are still exploring the mysteries of the cell system, we sincerely hope that the use of AI/ML will accelerate scientific discovery and lead to better treatments for diseases.

APPENDIX A

Supplementary Materials for Chapter 2

A.1 Intractability of RNA Velocity Inference

Gene-Shared Latent Times It is shown in scVelo[98] that given the u and s values of a single gene, cell time can be inferred using an EM algorithm. However, their approach results in different cell times for different genes. Instead, we would like to develop an EM algorithm to infer the unique cell time, which is shared across all genes. We denote time as $\boldsymbol{t}$ and the ODE parameters as $\boldsymbol{\theta}$. Following the standard EM algorithm, we obtain the E-step at the $(j+1)$-th iteration:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta};\boldsymbol{\theta}^{(j)}) &= \mathbb{E}_{p(\boldsymbol{t}|\boldsymbol{X};\boldsymbol{\theta}^{(j)})}\left[\ln p(\boldsymbol{X}|\boldsymbol{t};\boldsymbol{\theta})\right] \\ &= \sum_{i=1}^{N} \mathbb{E}_{p\left(t^{(i)}|\boldsymbol{x}^{(i)};\boldsymbol{\theta}^{(j)}\right)}\left[\ln p(\boldsymbol{x}^{(i)}|t^{(i)};\boldsymbol{\theta})\right]\end{aligned}$$

Here, we make the assumption that t_i and $\boldsymbol{x_i}$ $(i = 1, 2, \ldots, N)$ are mutually independent. First, without computing the exact form, we can show that the posterior is intractable.

$$p(t|\boldsymbol{x};\boldsymbol{\theta}) = \frac{p(\boldsymbol{x}|t;\boldsymbol{\theta})p(t)}{\int_{t'=-\infty}^{+\infty} p(\boldsymbol{x}|t';\boldsymbol{\theta})p(t')dt'}$$

It's natural to assume that the time prior $p(t)$ is uniform in $[0, T]$. However, VeloVAE assumes a Gaussian prior $\mathcal{N}(\mu_0, \sigma_0^2)$ because the support of a uniform distribution is not $\mathbb{R}$ and the KL divergence might be undefined in some cases. For the purpose of analysis, we choose the uniform prior here. Later in the case of unshared latent time, we will see that with certain approximations, using a uniform prior results in the same algorithm as scVelo.

In addition, we assume the covariance matrix of u and s of all genes is diagonal, i.e.

$\boldsymbol{\Sigma} = \text{diag}(\sigma_{u,1}^2, \ldots, \sigma_{u,G}^2, \sigma_{s,1}^2, \ldots, \sigma_{s,G}^2)$. Since $p(\boldsymbol{x}|t'; \boldsymbol{\theta})$ is Gaussian, we have

$$p(t|\boldsymbol{x}; \boldsymbol{\theta}) = \frac{\frac{1}{(2\pi)^G |\boldsymbol{\Sigma}|^{\frac{1}{2}}} e^{-d(t)^2} \cdot \frac{1}{T} I_{\{t \in [0,T]\}}}{\int_{t'=-\infty}^{+\infty} \frac{1}{(2\pi)^G |\boldsymbol{\Sigma}|^{\frac{1}{2}}} e^{-d(t')^2} \cdot \frac{1}{T} I_{\{t' \in [0,T]\}} dt'} = \frac{e^{-d(t)^2} I_{\{t \in [0,T]\}}}{\int_{t'=0}^{T} e^{-d(t')^2} dt'} \tag{A.1}$$

$$\text{where } d(t)^2 := \sum_{g=1}^{G} \frac{1}{2\sigma_{u,g}^2} (u_g - \hat{u}_g(t))^2 + \frac{1}{2\sigma_{s,g}^2} (s_g - \hat{s}_g(t))^2 \tag{A.2}$$

Now consider the integral in the denominator. We know that both $\hat{u}(t)$ and $\hat{s}(t)$ have at least one exponential term involving t. Without exact calculation, we know that the denominator will involve the integral of $\exp(\exp(-ct))$ with some constant c and this cannot be expressed by any elementary function. Let $C(\boldsymbol{\theta})$ be the constant equal to the integral in the denominator. Therefore, the total likelihood function is

$$\mathcal{L}(\boldsymbol{\theta}; \boldsymbol{\theta^{(j)}}) = \sum_{i=1}^{N} \int_t \frac{e^{-d\left(t^{(i)}; \boldsymbol{\theta^{(j)}}\right)^2}}{C^{(i)}(\boldsymbol{\theta^{(j)}})} \left[-G \ln(2\pi) - \frac{1}{2} \ln(|\boldsymbol{\Sigma}|) - d(t^{(i)}; \boldsymbol{\theta})^2 \right]$$

Because $C^{(i)}(\boldsymbol{\theta^{(j)}})$ is intractable, it's hard to directly optimize the total likelihood function. Hence, the EM algorithm cannot be easily applied.

In addition, if we pick a Gaussian prior, we would end up with the same result with slight difference in the form of $d(t)$:

$$d(t)^2 := \frac{(t - t_0)^2}{2\sigma_0^2} + \sum_{g=1}^{G} \frac{1}{2\sigma_{u,g}^2} (u_g - \hat{u}_g(t))^2 + \frac{1}{2\sigma_{s,g}^2} (s_g - \hat{s}_g(t))^2$$

Unshared Latent Times Now let's consider the special case of $G = 1$, which is just the local gene fitting in scVelo. The M-step in scVelo[98] is simply to minimize the sample mean square error:

$$\boldsymbol{\theta^{(j+1)}} = \arg\min_{\boldsymbol{\theta}} \frac{1}{N} \sum_{i=1}^{N} \left[\left(u^{(i)} - \hat{u}^{(i)}(t)\right)^2 + \left(s^{(i)} - \hat{s}^{(i)}(t)\right)^2 \right] \tag{A.3}$$

where $\hat{u}, \hat{s}$ are predictions by the learned kinetic function of the gene. First, we need to assume $C^{(i)}(\boldsymbol{\theta^{(j)}})$ is a constant for all i. With this approximation, the M-step becomes

$$\begin{aligned}
&\max_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{\theta^{(j)}}) \\
=&\max_{\boldsymbol{\theta}} \sum_{i=1}^{N} \int_{t^{(i)}=0}^{T} \frac{e^{-d\left(t^{(i)};\boldsymbol{\theta^{(j)}}\right)^2}}{C^{(i)}(\boldsymbol{\theta^{(j)}})} \left[-c_\sigma - d(t^{(i)};\boldsymbol{\theta})^2\right] dt^{(i)} \\
=&\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} \int_{t^{(i)}=0}^{T} \frac{e^{-d(t^{(i)};\boldsymbol{\theta^{(j)}})^2}}{C^{(i)}(\boldsymbol{\theta^{(j)}})} d(t^{(i)};\boldsymbol{\theta})^2 dt^{(i)} \qquad \text{(A.4)} \\
\approx&\min_{\boldsymbol{\theta}} \frac{1}{C(\boldsymbol{\theta^{(j)}})} \sum_{i=1}^{N} \int_{t^{(i)}=0}^{T} d(t^{(i)};\boldsymbol{\theta})^2 e^{-d\left(t^{(i)};\boldsymbol{\theta^{(j)}}\right)^2} dt^{(i)} \qquad \text{(A.5)}
\end{aligned}$$

We further assume $\sigma_u = \sigma_s = \sigma$, so $d(t)^2 = \frac{1}{2\sigma^2}\left[(u-\hat{u}(t))^2 + (s-\hat{s}(t))^2\right]$. We assume that $\sigma \approx 0$ and $d(t;\boldsymbol{\theta})$ has a global minimum $t_0 \in [0,T]$. As σ approaches 0, $d(t;\boldsymbol{\theta^{(j)}})$ approaches infinity. Following the analysis by Li et al.[246], we apply the Laplace's method to approximate the integral:

$$\int_{t^{(i)}=0}^{T} d(t^{(i)};\boldsymbol{\theta})^2 e^{-d\left(t^{(i)};\boldsymbol{\theta^{(j)}}\right)^2} dt^{(i)} \approx \sqrt{\frac{2\pi}{2d''(t_0;\boldsymbol{\theta^{(j)}})d(t_0;\boldsymbol{\theta^{(j)}}) + 2d'(t_0;\boldsymbol{\theta^{(j)}})^2}} e^{-d\left(t_0;\boldsymbol{\theta^{(j)}}\right)^2} d(t;\boldsymbol{\theta})^2 \qquad \text{(A.6)}$$

$$\propto d(t;\boldsymbol{\theta})^2 \qquad \text{(A.7)}$$

Using (A.5) and (A.7), we obtain the final result:

$$\arg\max_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{\theta^{(j)}}) = \arg\min_{\boldsymbol{\theta}} \frac{1}{N} \sum_{i=1}^{N} \left[\left(u^{(i)} - \hat{u}^{(i)}(t)\right)^2 + \left(s^{(i)} - \hat{s}^{(i)}(t)\right)^2\right]$$

Therefore, minimizing the mean square error is equivalent to maximizing the total likelihood function under all the assumptions and approximations we made above.

A.2 Supplementary Results

In a preliminary benchmark study, we compared scVelo (SOTA), basic model (VAE with fixed rates), and VeloVAE (VAE with latent cell state and variable rates). We include the basic model to assess the importance of the mixture of ODEs. The metrics we use are (1) MSE = Mean Squared Error; (2) k_t = Time correlation; and (3) k_t(Info.) = Time correlation under informative prior. These metrics are well justified. MSE is widely used for assessing how

well the models fit the observed data. The limitations of the single-cell data itself preclude ground truth for the cell times. However, the inferred times should at least be correlated with the cell capture times when available (usually on the order of days). In addition, we evaluated the model performance under an informative time prior, shown in the column of k_t(INFO.). Our results show that (1) using a variable transcription rate with dependency on cell state improves data reconstruction and (2) incorporating known experimental time into the Bayesian framework improves time correlation significantly (**Table A.1**).

Table A.1: **Performance on scRNA-seq Datasets.**

Dataset	Method	MSE	k_t	k_t(Info.)
Pancreas	scVelo	2.107	N/A	N/A
	Basic Model	6.815	N/A	N/A
	VeloVAE	**0.823**	N/A	N/A
Dentate Gyrus 1	scVelo	0.670	N/A	N/A
	Basic Model	0.574	N/A	N/A
	VeloVAE	**0.243**	N/A	N/A
Mouse Brain 1	scVelo	10.160	N/A	N/A
	Basic Model	10.431	N/A	N/A
	VeloVAE	**1.886**	N/A	N/A
Erythroid Lineage	scVelo	0.873	-0.707	N/A
	Basic Model	0.246	**0.802**	0.802
	VeloVAE	**0.151**	0.622	**0.855**
Dentate Gyrus 2	scVelo	1.385	-0.158	N/A
	Basic Model	0.968	0.304	0.306
	VeloVAE	**0.159**	**0.529**	**0.707**
Mouse Brain 2	scVelo	18.19	-0.777	N/A
	Basic Model	2.295	0.621	0.629
	VeloVAE	**0.152**	**0.870**	**0.897**

A.3 Supplementary Figures

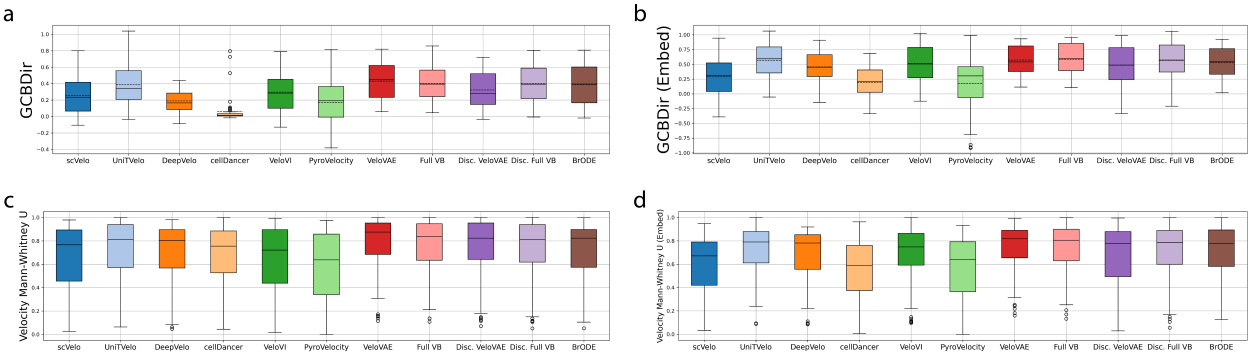

Figure A.1: **Comparison of Other Performance Metrics.** We compared VeloVAE with all other methods in terms of Generalized CBDir on both gene expression space **(a)** and 2D low-dimensional embedding **(b)**, velocity accuracy on both gene expression space **(c)** and 2D low-dimensional embedding **(d)** and time accuracy. All metrics are computed based on prior knowledge of cell development in each tissue. Note that we only compared the genes which scVelo fitted, i.e. velocity genes, as there are genes scVelo didn't fit or fitted with zero likelihood.

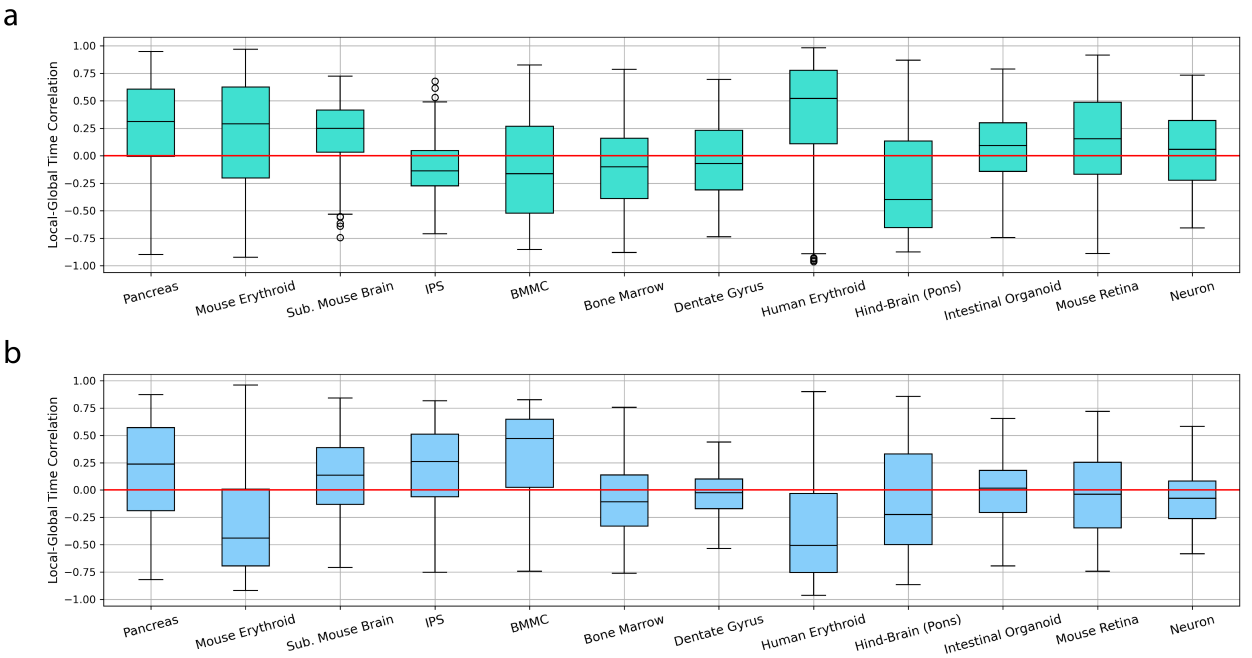

Figure A.2: **Histogram of scVelo and VeloVI Time Correlation.** We computed the the Spearman correlation between scVelo **(a)** and VeloVI **(b)** locally fitted (gene-specific) time and their global latent time respective for twelve benchmarking datasets. This gives a correlation value for each fitted gene. The figures show box plots of these correlations.

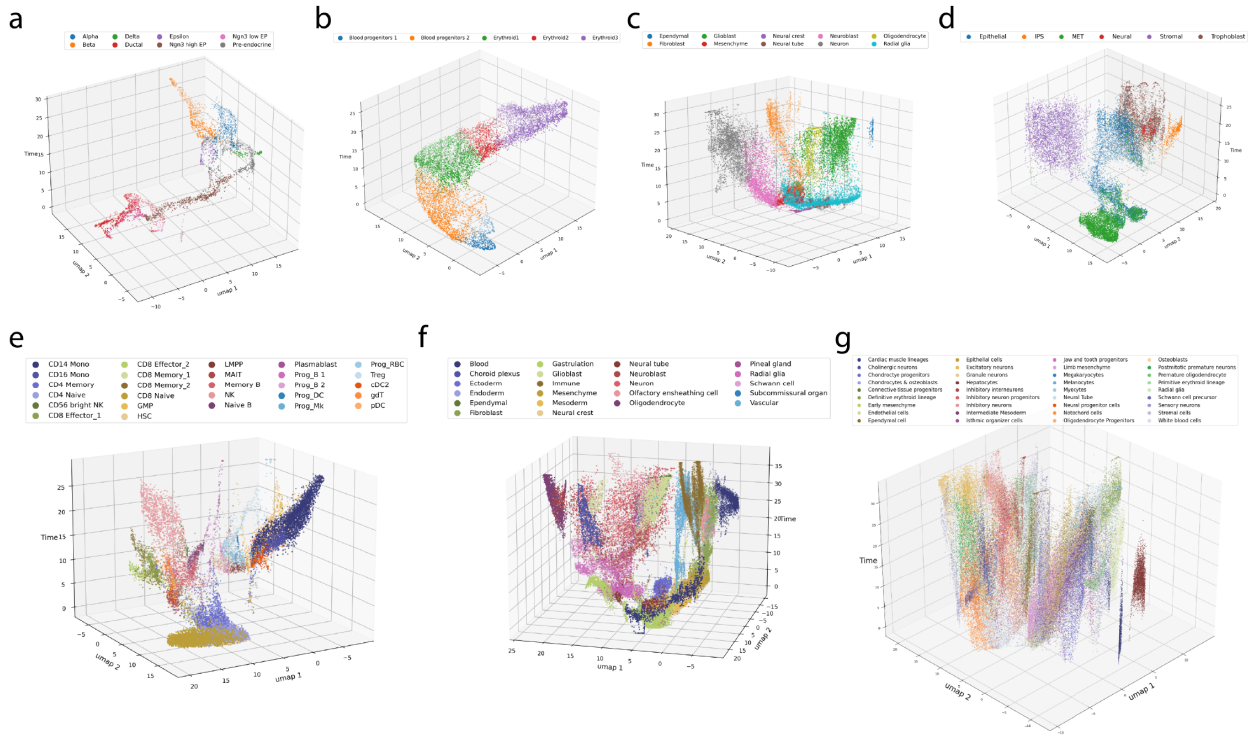

Figure A.3: **Cell State-time Plot.** 2D UMAP embeddings of the latent cell state **c** plotted versus inferred latent time. Each plot has the 2D UMAP embedding as the x-y plane and cell time as the z axis. The plots can be interpreted as time evolution of the cell state space. The seven panels correspond to pancreas **(a)**, erythroid **(b)**, subsampled mouse brain **(c)**, iPSC **(d)**, human bone marrow **(e)**, full mouse brain **(f)** and mouse embryo **(g)**.

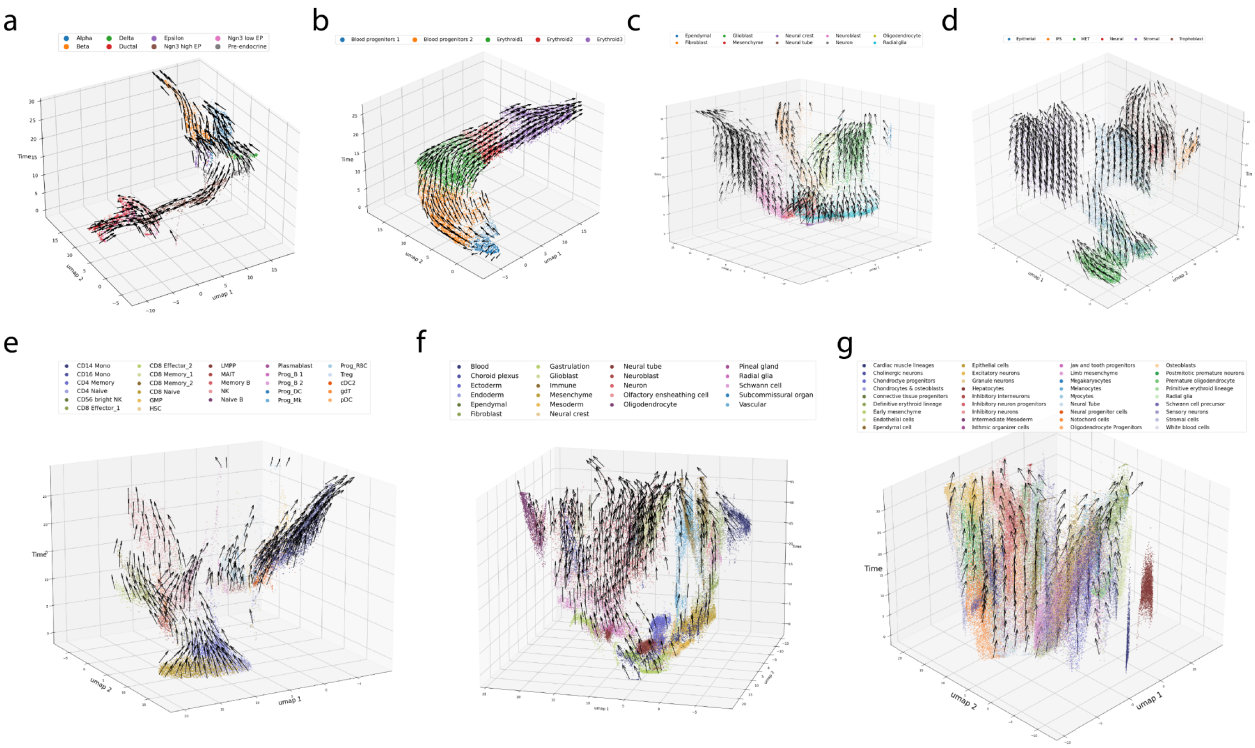

Figure A.4: **3D Velocity Plot.** 3D plots computed as described in Fig. A.3, with velocity vectors added. The x-y plane is either UMAP (a-e) or t-SNE (f-g) coordinates. The 3D embedding is computed by averaging displacement vectors towards k neareast neighbors in the immediate future, i.e. arrows point upwards along the z axis. The seven panels correspond to pancreas **(a)**, erythroid **(b)**, subsampled mouse brain**(c)**, iPSC **(d)**, human bone marrow **(e)**, full mouse brain **(f)** and mouse embryo **(g)**.

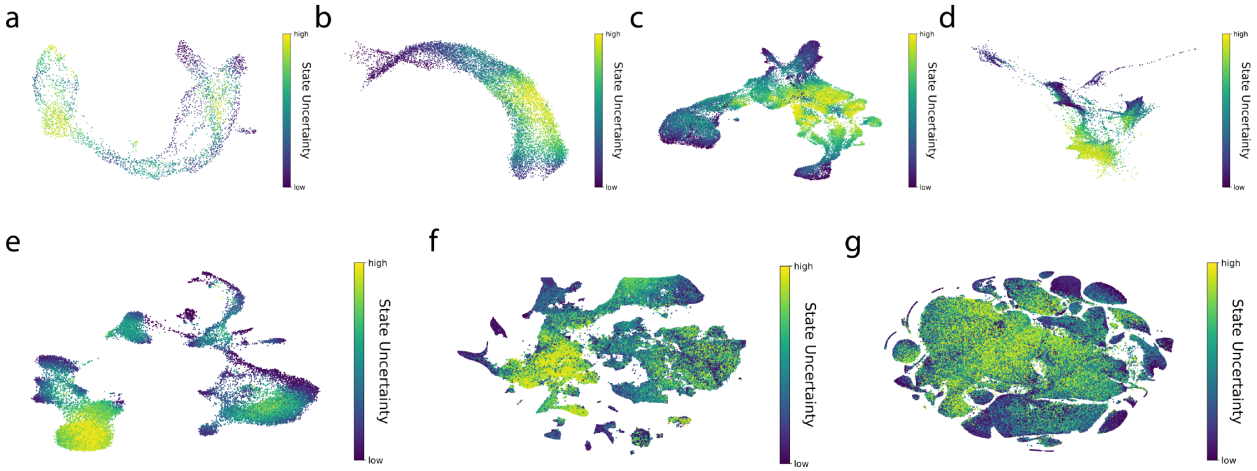

Figure A.5: **Inferred Cell State Uncertainty.** Each panel shows a UMAP or t-SNE plot colored by the multi-variate coefficient of variation of the cell state. Cell state is a continuous identification of cell type, so conceptually it should have high uncertainty in progenitor cell types, as cell fate is undetermined. Our results verified this intuition as well as the capability of learning meaningful representations using VeloVAE. The 7 panels correspond to pancreas **(a)**, erythroid **(b)**, subsampled mouse brain **(c)**, iPSC **(d)**, human bone marrow **(e)**, full mouse brain **(f)** and mouse embryo **(g)**.

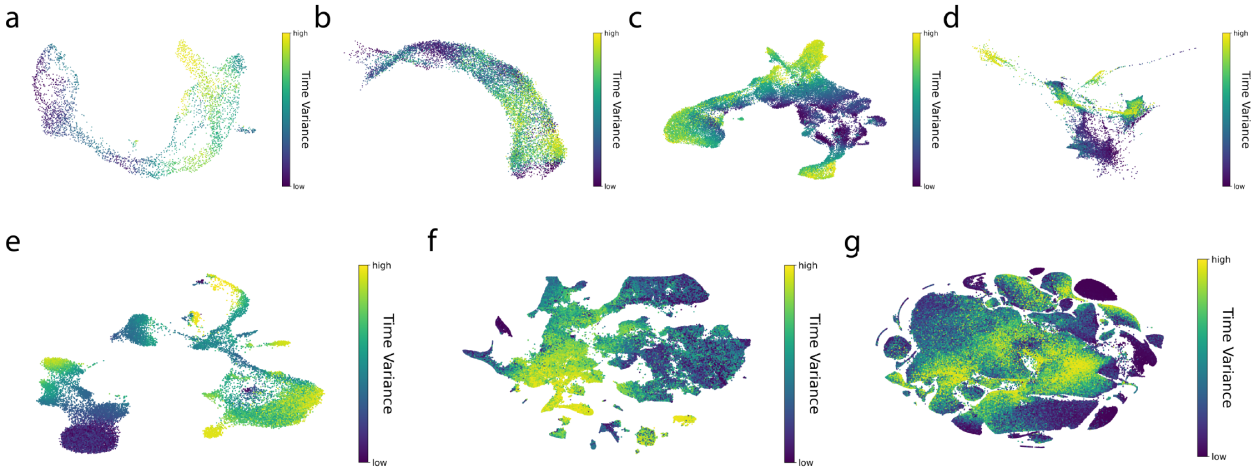

Figure A.6: **Time Variance in Test Datasets.** Each panel shows a UMAP or t-SNE plot colored by the coefficient of variation of the inferred cell time. The 7 panels correspond to pancreas **(a)**, erythroid **(b)**, subsampled mouse brain**(c)**, iPSC **(d)**, human bone marrow **(e)**, full mouse brain **(f)** and mouse embryo **(g)**.

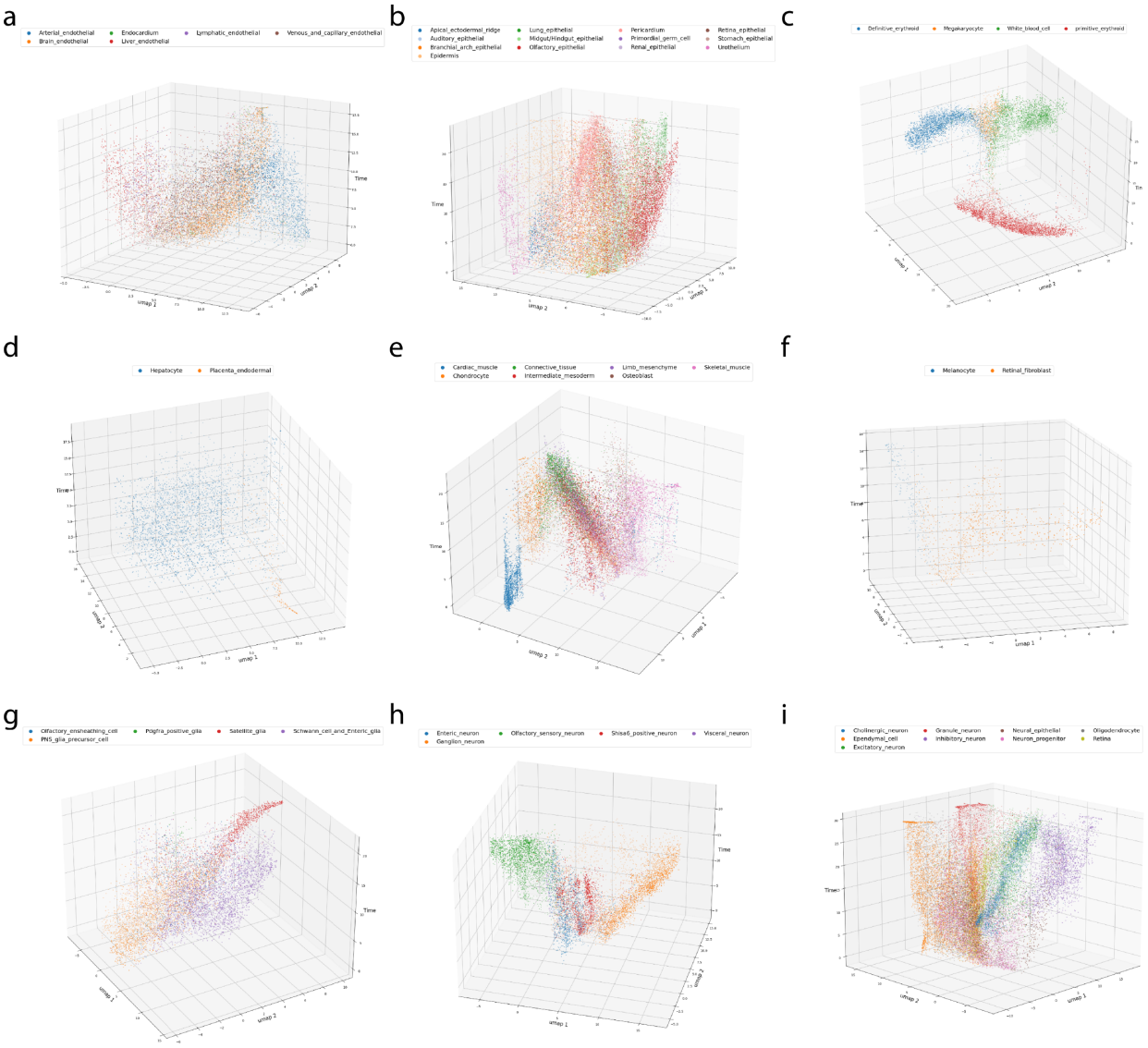

Figure A.7: **Cell State-time Plot.** We computed 2D UMAP embeddings of the latent cell state **c** and plot it versus inferred latent time. Following the setting of the original work, we used cosine distance and 15 neighbors. The 10 panels correspond to endothelial **(a)**, epithelial **(b)**, haematopoiesis **(c)**, hepatic **(d)**, mesenchymal **(e)**, melanocyte **(f)**, PNS glia**(g)**, PNS neuron **(h)** and neural tube and notochord **(i)** trajectories in the mouse embryo dataset.

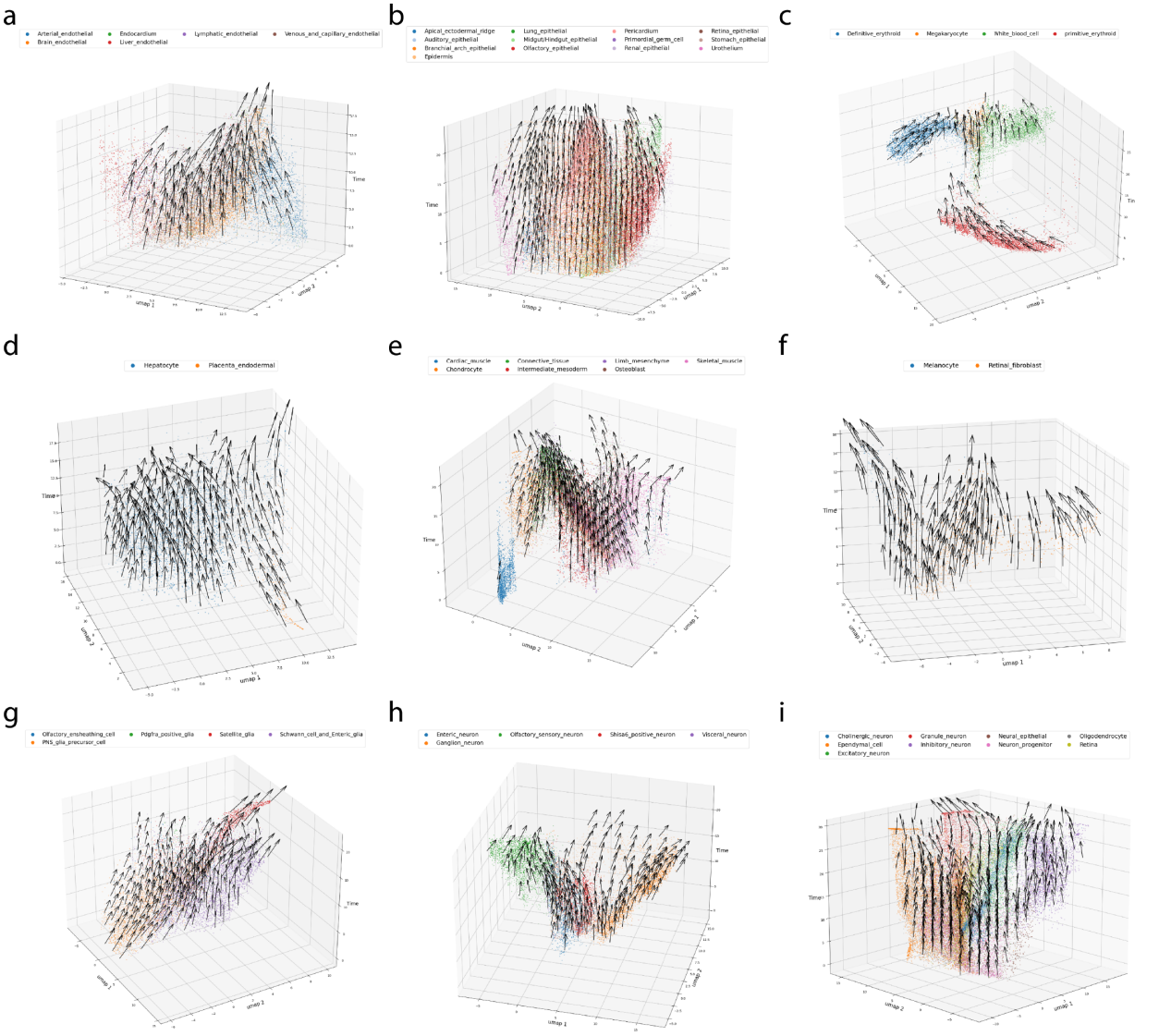

Figure A.8: **3D Velocity Plot.** 3D velocity plot computed as described in Fig. S4. The 10 panels correspond to endothelial **(a)**, epithelial **(b)**, haematopoiesis **(c)**, hepatic **(d)**, mesenchymal **(e)**, melanocyte **(f)**, PNS glia **(g)**, PNS neuron **(h)** and neural tube and notochord **(i)** trajectories in the mouse embryo dataset.

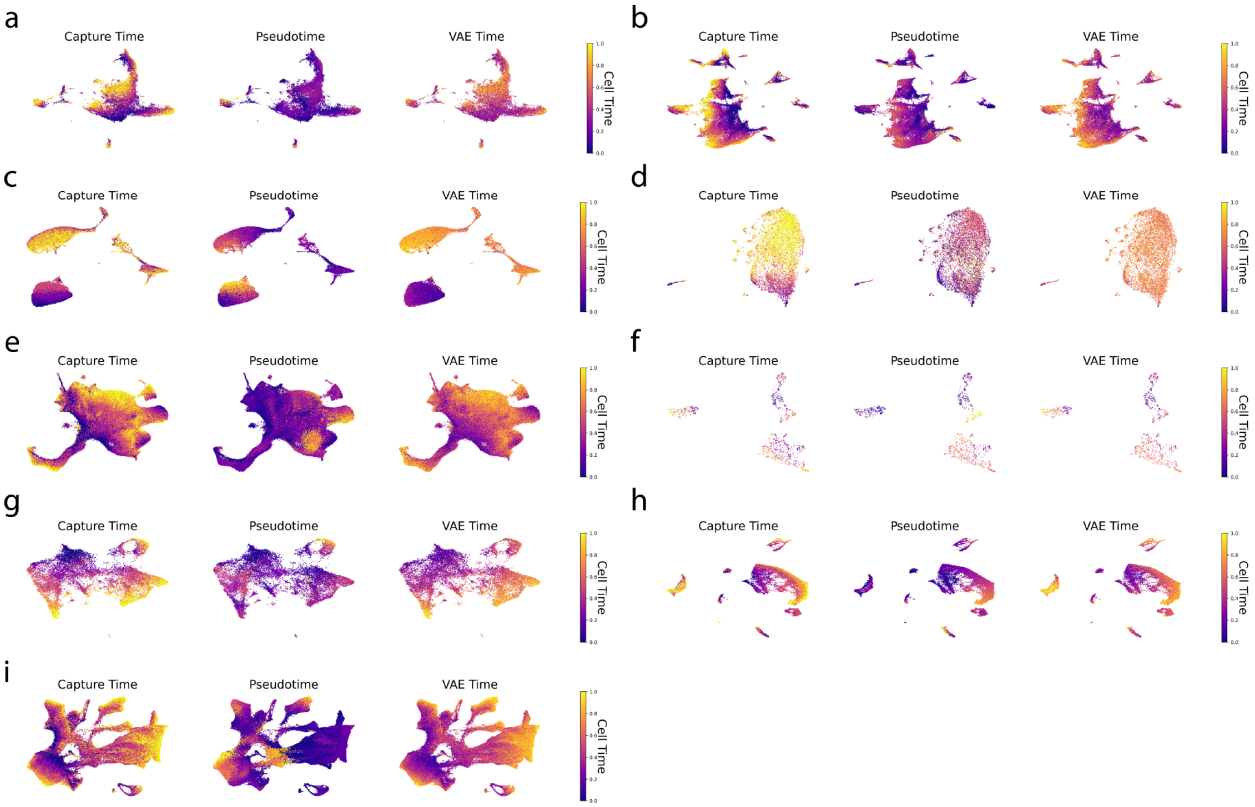

Figure A.9: **Latent Time of Major Trajectories from the Mouse Embryo.** We compare the capture time, monocle 3 pseudotime and VeloVAE latent time by showing the UMAP plots colored by latent time of ten major trajectories in the embryo dataset: endothelial**(a)**, epithelial **(b)**, haematopoiesis **(c)**, hepatic **(d)**, mesenchymal **(e)**, melanocyte **(f)**, PNS glia **(g)**, PNS neuron **(h)** and neural tube and notochord **(i)** trajectories.

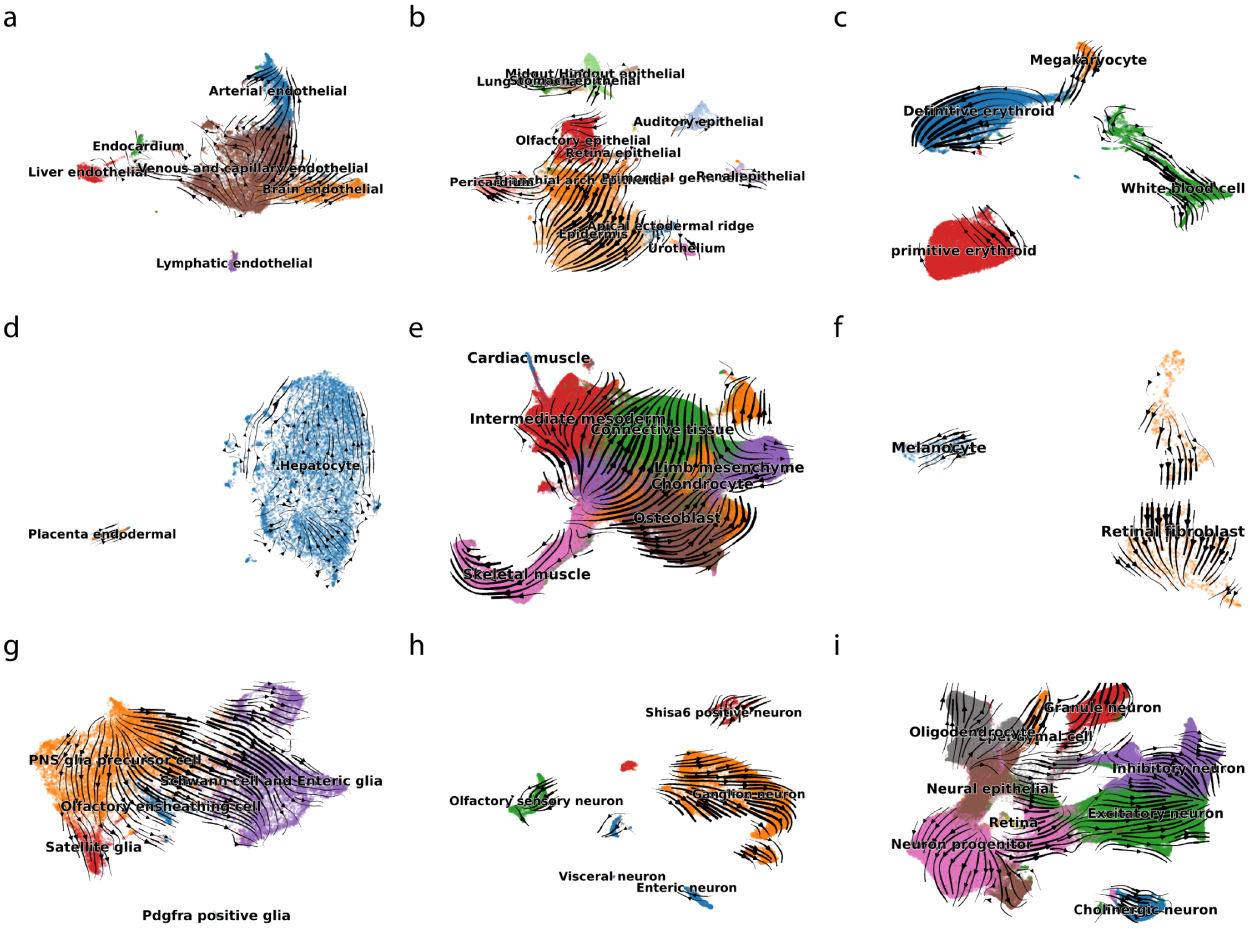

Figure A.10: **Velocity Stream Plots for Major Trajectories from the Mouse Embryo.** We plotted the velocity embedding on UMAP coordinates of ten major trajectories in the embryo dataset: endothelial **(a)**, epithelial **(b)**, haematopoiesis **(c)**, hepatic **(d)**, mesenchymal **(e)**, melanocyte **(f)**, PNS glia **(g)**, PNS neuron **(h)** and neural tube and notochord **(i)** trajectories.

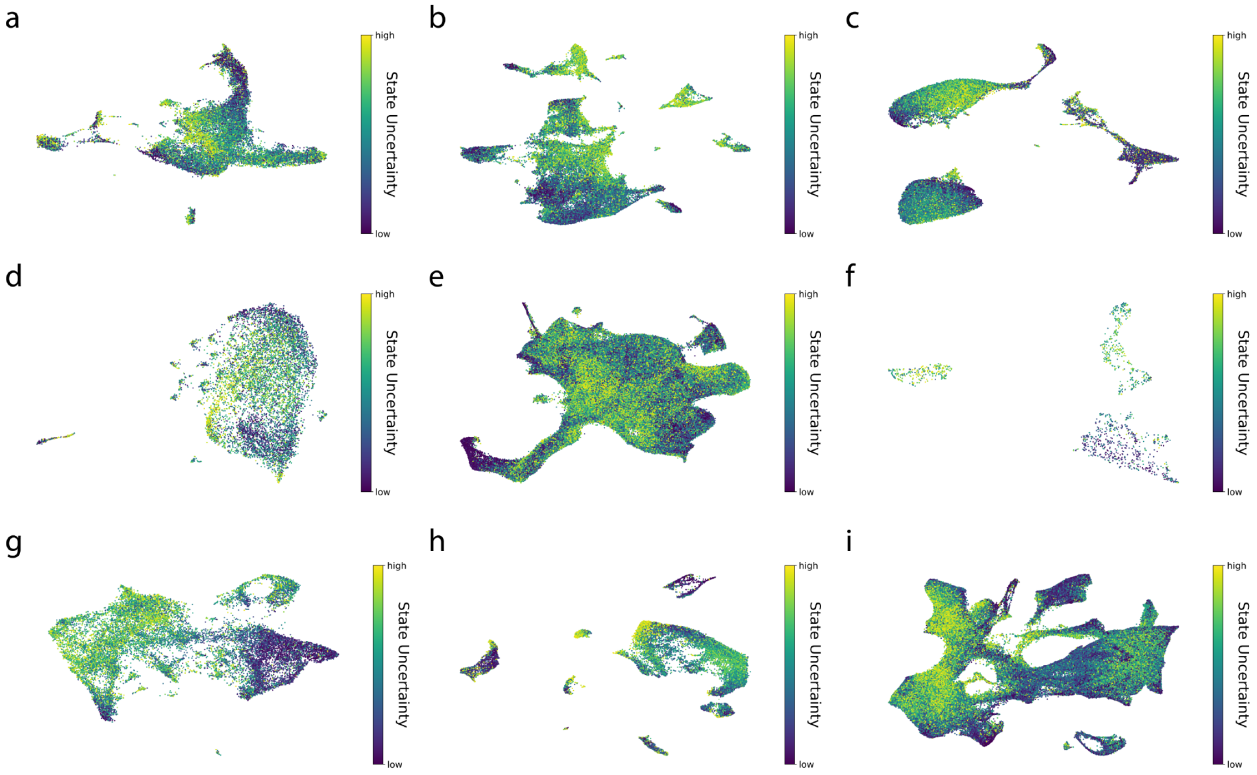

Figure A.11: **Cell State Uncertainty in Major Mouse Embryo Trajectories.** Each panel shows a UMAP plot colored by the multi-variate coefficient of variation of the cell state. The 10 panels correspond to endothelial **(a)**, epithelial **(b)**, haematopoiesis **(c)**, hepatic **(d)**, mesenchymal **(e)**, melanocyte **(f)**, PNS glia **(g)**, PNS neuron **(h)** and neural tube and notochord **(i)** trajectories in the mouse embryo dataset.

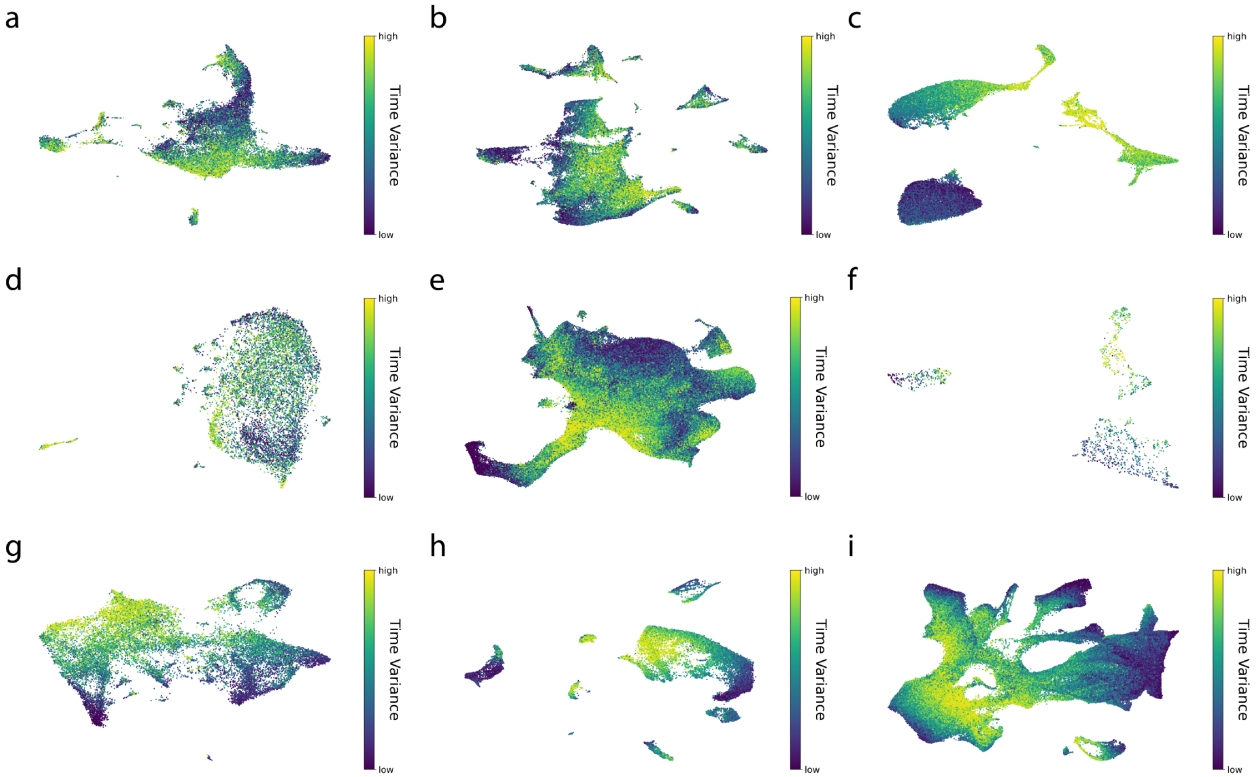

Figure A.12: **Cell Time Variance in Major Mouse Embryo Trajectories.** Each panel shows a UMAP plot colored by the coefficient of variation of the cell time. The 10 panels correspond to endothelial **(a)**, epithelial **(b)**, haematopoiesis **(c)**, hepatic **(d)**, mesenchymal **(e)**, melanocyte **(f)**, PNS glia **(g)**, PNS neuron **(h)** and neural tube and notochord **(i)** trajectories in the mouse embryo dataset.

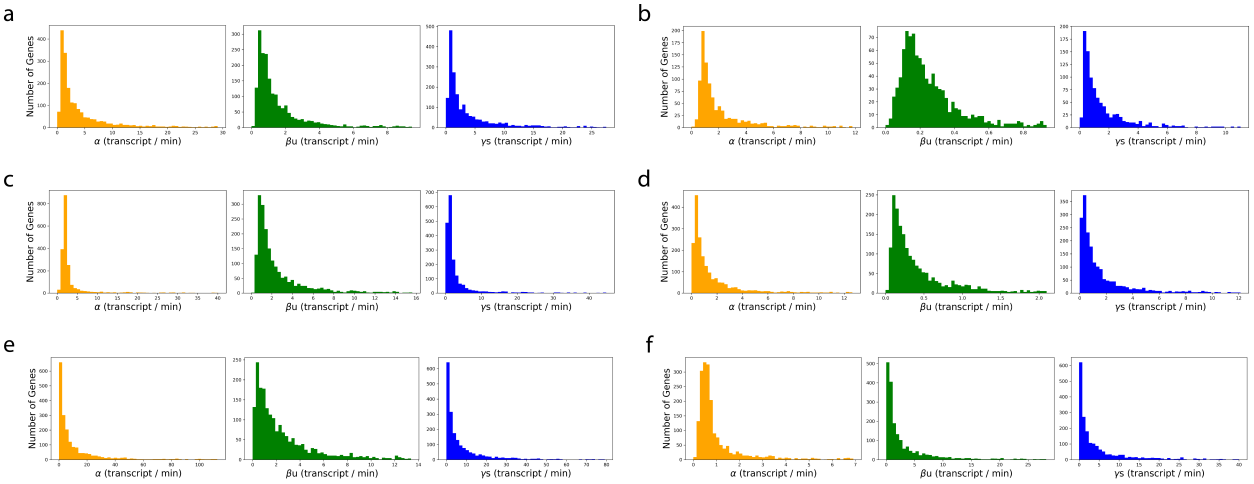

Figure A.13: **Rate Parameter Histograms.** Box and histogram plots showing the overall range of transcription, splicing and degradation rates estimated by different methods. For VeloVAE, transcription rates are reported as $\rho = 1$. For splicing and degradation, we report βu and γs to match the unit of α (mRNA / minute). Here, u and s are chosen to be half of the 95th-percentile count number. The first four panelsT correspond to pancreas **(a)**, erythroid **(a)**, iPSC **(b)**, subsampled mouse brain **(c)**, full mouse brain **(d)**, and mouse embryo **(i)** datasets.

A.4 Hyper-Parameter Settings

We trained our model with minimal change to the default hyperparameters. The only changes we made include **(1)** setting early_stop to 9 and train_ton to False for the erythroid dataset **(2)** increasing the batch size to 2048 and n_neighbors to 30 for the mouse embryo dataset, and (3) setting reverse_gene_mode to True for some datasets due to bad initialization from the dynamical model (**Table A.4**).

Parameter Name	Value	Notes
h_1	500	Width of the first hidden encoder layer/ second decoder layer
h_2	250	Width of the second hidden encoder layer/ first decoder layer
n_epochs	1000	Number of (maximum) training epochs
n_epochs_post	1000	Number of (maximum) refining epochs
n_refine	20	Maximum number of rounds in stage 2
batch_size	128	Number of samples in a mini-batch
learning_rate	$10^{-8.3p-2.25}$	Learning rate of neural network parameters in stage 1 (p is data density)
learning_rate	same as learning_rate	Learning rate of neural network parameters in stage 2
learning_rate_ode	8×learning_rate	Learning rate of ODE parameters
λ	1×10^{-3}	L2 regularization coefficient of the encoder network
λ_ρ	1×10^{-3}	L2 regularization coefficient of the decoder ρ-network
kl_t	1.0	time KL divergence coefficient
kl_z	1.0	cell state KL divergence coefficient
kl_w	0.01	BasisVAE weight KL divergence coefficient
test_iter	2	number of epochs between two consecutive tests on the validation data
n_warmup	5	number of initial epochs when only the neural network parameters are updated
early_stop	5	number of consecutive epochs in the early stopping criteria
early_stop_thred	# gene $\times 10^{-3}$	ELBO change threshold in the early stopping criteria
train_test_split	0.7	proportion of training samples out of the entire dataset
k_alt	0	number of iterations before each update of the ODE parameters (0 means simultaneous update)
train_ton	True	whether to train the switch-on time of each gene
time_overlap	0.5	overlap proportion between two informative time prior distributions
n_neighbors	10	number of neighbors in time-window KNN initial condition estimation
dt	(0.03,0.06)	coefficient of the edges of the time window in KNN initial condition estimation
reverse_gene_mode	False	whether to reverse cell time for model initialization

Table A.2: **Default Hyperparameters of VeloVAE.**

Parameter Name	Value	Notes
n_epochs	500	Number of (maximum) training epochs
batch_size	128	Number of samples in a mini-batch
learning_rate	10^{-4*p-3}	Learning rate of ODE parameters (p is data density)
n_update_noise	25	number of epochs before each update of the noise variance
test_iter	2	number of epochs between two consecutive tests on the validation data
early_stop	5	number of consecutive epochs in the early stopping criteria
early_stop_thred	# gene $\times 10^{-3}$	ELBO change threshold in the early stopping criteria
train_test_split	0.7	proportion of training samples out of the entire dataset
train_scaling	False	whether to train the scaling parameter of unspliced counts

Table A.3: **Default Hyperparameters of Branching ODE.**

Dataset	Use Capture Time as Prior	Initialization Method	Reverse Gene Mode
Pancreas[179]	No	Steady-State Model	No
Mouse Erythroid[247]	Yes	Dynamical Model	No
Mouse Brain[180]	Yes	Dynamical Model	No
iPS[88]	Yes	Dynamical Model	No
Human Hematopoiesis[86]	No	Steady-State Model	Yes
Dentate Gyrus[248]	No	Steady-State Model	No
Human Erythroid[249]	No	Steady-State Model	Yes
Mouse Hindbrain(pons)[89]	No	Steady-State Model	Yes
Mouse Retina[250]	No	Steady-State Model	No
Intestinal Organoid[251]	No	Steady-State Model	No
Neuron Genesis[252]	Yes	Dynamical Model	No

Table A.4: **Variable Hyperparameters across Datasets.** We utilized the capture time in certain datasets and reversed gene mode assignment to avoid bad initialization.

APPENDIX B

Supplementary Materials for Chapter 3

B.1 Preprocessing Details

Here, we first list the spatial graph used for training TopoVelo on each dataset in **Table B.1**. Denote k as the number of neighbors in a KNN graph and r as the radius parameter for a ball graph or a radius constraint in a KNN graph. Next, we report the parameters of the preprocessing steps in **Table B.2**. We use the following abbreviations for conciseness:

1. GF - preliminary gene filtering by total count number
2. NM - count normalization
3. VG - select variable genes
4. SVG - select spatially variable genes
5. DG - select differentially expressed genes using Wilcoxon test
6. KS - KNN Smoothing

Dataset Name	Graph Type	Parameter
E15 Mouse Brain Cortex	Delaunay	N/A
E13.5 Mouse GI Tract	KNN	k=16
E13.5 Lung	KNN	k=16
Postnatal 3-month Mouse Thymus	Ball Graph	r=120μm
E14.5 Mouse Brain	KNN with Radius Constraint	k=16, r=36.5μm (50 units)
E9 3D Neural Tube	Ball Graph	r=32.35μm (50 units)
Embryoid	Ball Graph	r=32.35μm (50 units)

Table B.1: **Spatial Graph Construction**

Dataset Name	Technology	No. Cells	No. Genes	Preprocessing Steps
E15 Mouse Brain Cortex	Slide-seq	2525	200	GF, NM, VG, KS
E13.5 Mouse GI Tract	Stereo-seq	6366	2000	GF, NM, VG, KS
E13.5 Mouse Lung	Stereo-seq	2182	2000	GF, NM, VG, KS
Postnatal 3-month Mouse Thymus	10x Visium	2165	100	GF, NM, VG, KS
E14.5 Mouse Brain	Slide-tags	4623	500	SVG, GF, NM, KS
E9 Mouse 3D Neural Tube	Slide-seq	10432	247	SVG, GF, NM, KS
Embryoid	Slide-seq (Curio)	5559	66	GF, NM, DG, KS

Table B.2: **Data Preprocessing**

B.2 Simulation of Unspliced and Spliced mRNA Data Using Coupled Spatiotemporal Dynamics

In the simulation study, we extended the method from scVelo to include spatial information and coupled it with temporal dynamics. We defined two different types of spatial dynamics in this study, 1D bidirectional and 2D radial growth. The former mimics the process of a single layer of progenitor cells growing to both sides, while the latter is a case of cells expanding outwards to form concentric circles. These two cases are simple, but can be treated as building blocks for many real biological processes. For example, the VZ layer in the mouse brain cortex grows mostly in a constant direction to form subsequent layers. As for radial growth, epithelial cells in the inner part of the gut migrate outwards to become mesenchymal cells. Although these two simulated spatial dynamics differ, they share the same steps in our simulation. Therefore, we will describe our algorithm in a general sense, but refer to the details in **Table B.3**. In the following description, we always use $\mathcal{N}(\cdot, \cdot)$ to denote a uni- or multi-variate Gaussian distribution with mean and covariance matrix inside the parenthesis and $\mathcal{U}(a, b)$ to denote a uniform distribution in the interval $[a, b]$.

The simulation algorithm is composed of three parts, simulating the spatial dynamics, generating spatially dependent transcription rates, and simulating the temporal dynamics. In the first step, we can specify the initial number of cells at $t = 0$ and start by randomly sampling their initial positions. We assume cell development has a tree structure, i.e., a progenitor cell gives rise to one or multiple descendant cells. Each initial cell is the root of a tree, and we recursively build the tree. Given a progenitor cell in a tree with time t, it has

	1D Bidirectional	Radial	Notes
Initial Location	$x(0) \sim \mathcal{N}(0, \sigma_0^2)$ $y(0) \sim \mathcal{U}(-\frac{d}{2}, \frac{d}{2})$	$r(0) \sim \mathcal{U}(0, r_0)$ $\theta(0) \sim \mathcal{U}(0, 2\pi)$	
Spatial Dynamics	$v_x = v_0 + a_x t$ $v_y \sim B \cdot \mathcal{U}(v_{y,min}, v_{y,max})$	$v_r = v_0 + a_r t$ $v_\theta \sim B \cdot \mathcal{U}(v_{\theta,min}, v_{\theta,max})$	$B \sim \text{Beta}(\alpha, \beta)$
Transcription Rate (ρ)	$\frac{1}{2} e^{\left(-\frac{(x-x^*)^2}{\sigma_x^2} - \frac{(y-y^*)^2}{\sigma_y^2}\right)}$ $+\frac{1}{2} e^{\left(-\frac{(x+x^*)^2}{\sigma_x^2} - \frac{(y+y^*)^2}{\sigma_y^2}\right)}$	$e^{\left(-\frac{(r-r^*)^2}{\sigma_x^2} - \frac{(1-\cos(\theta-\theta^*))^2}{\sigma_\theta^2}\right)}$	$(x^*, y^*) \sim$ $\mathcal{U}([0, w] \times [-\frac{d}{2}, \frac{d}{2}])$ $(r^*, \theta^*) \sim$ $\mathcal{U}([0, R] \times [0, 2\pi])$

Table B.3: **Parametric Model for Spatial Dynamics.** The table lists different forms of spatial velocity and spatially-dependent transcription rates used for simulating 1D bidirectional and 2D radial cases.

k children with time $t + \Delta t$, where k is geometrically distributed and Δt is exponentially distributed. This step stops when the number of cells in all the trees exceeds a threshold set by the user. Once cell time is sampled using the tree, we can predict their locations using the spatial dynamics model in **Table B.3**. After generating the spatial locations, the algorithm samples a spatially dependent transcription rate to resemble the real spatial coordination of cells regarding gene expression. We assume the transcription rate has a peak at a certain location and use a superposition of two heat kernels in the 1D bidirectional case. Similarly, we assume the transcription rate has a peak at a certain angle when the tissue is expanding radially and apply a heat kernel on the polar coordinates. Finally, we use the analytical form of the ODE solution (**Eq. B.1, B.2**) to generate unspliced and spliced counts. To model technical noise, we added a Gaussian noise to the noiseless data and randomly set a proportion of entries to be zero as a result of dropout. We set the sparsity of our data to be 75%.

$$u(t) = u_0 \exp(-\beta\tau) + \frac{\rho\alpha}{\beta}(1 - \exp(-\beta\tau)) \tag{B.1}$$

$$s(t) = s_0 \exp(-\gamma\tau) + \frac{\rho\alpha}{\gamma}(1 - \exp(-\gamma\tau)) + \frac{\rho\alpha - \beta u_0}{\gamma - \beta}(\exp(-\gamma\tau) - \exp(-\beta\tau)) \tag{B.2}$$

B.3 Supplementary Figures

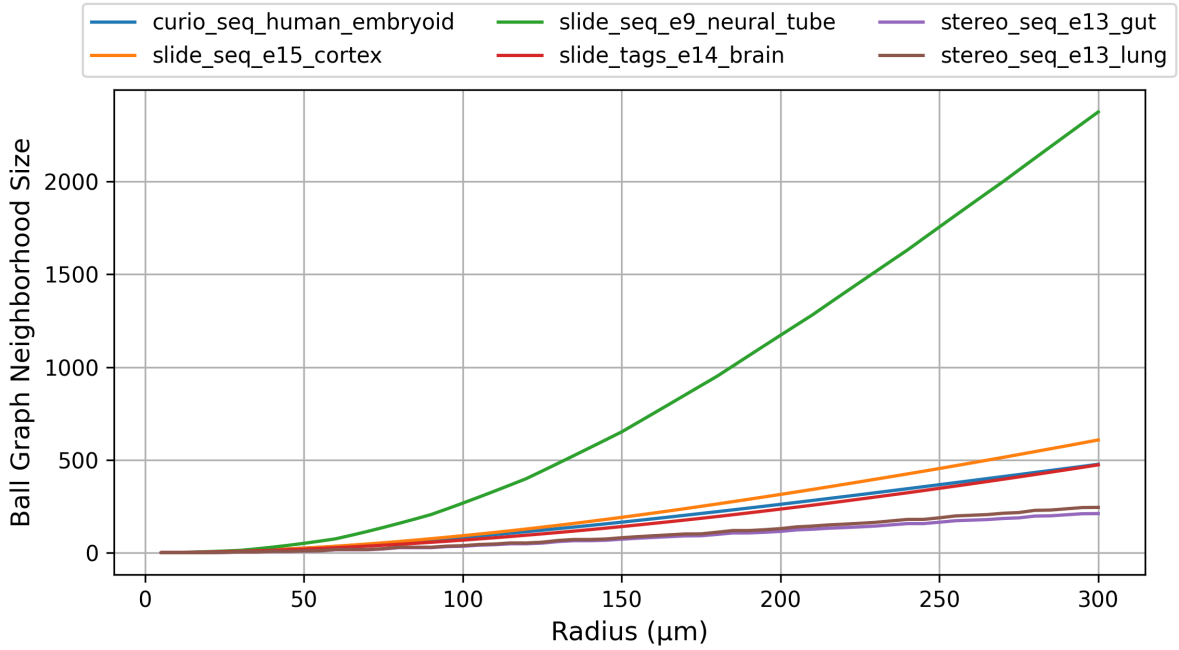

Figure B.1: **Spatial Neighborhood Size Under Different Ball Graph Parameters.** The line plot shows the average number of spatial neighbors vs. the radius of a ball graph constructed on the spatial coordinates. Each line represents a dataset.

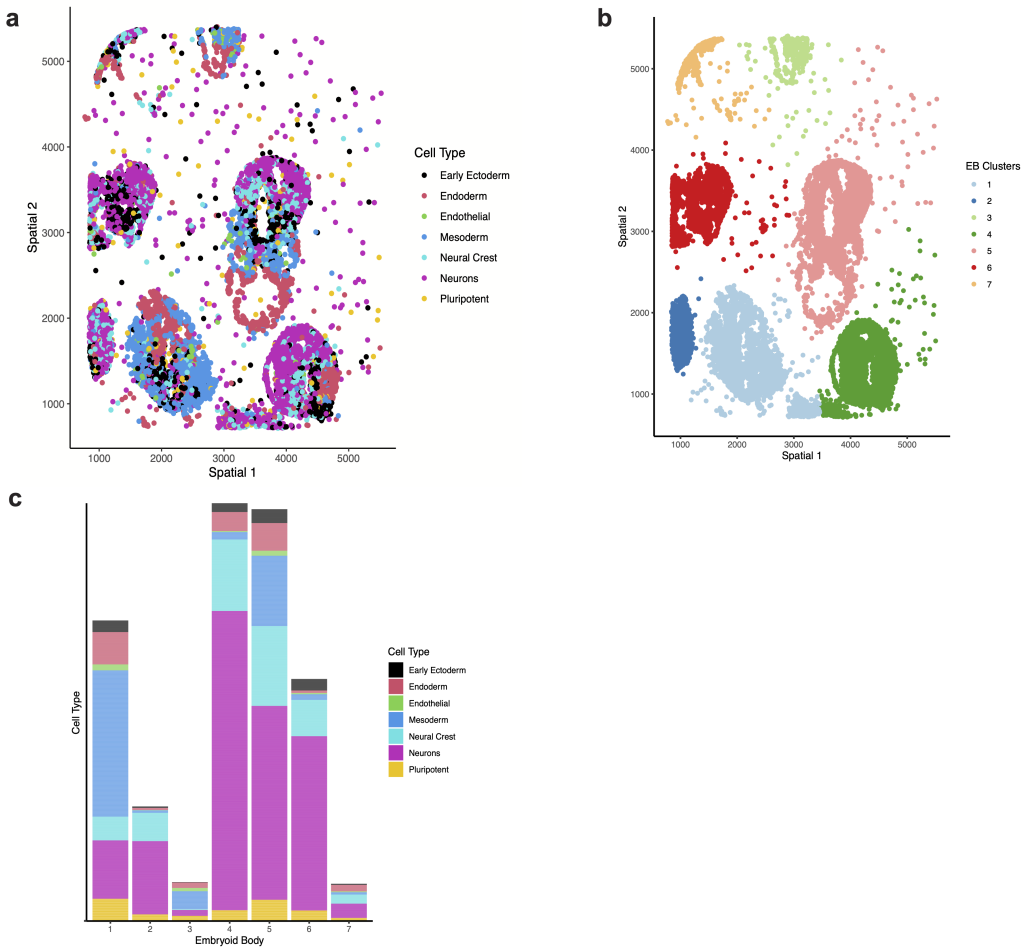

Figure B.2: **RCTD Clustering and Cell Type Compositions of Spatial Transcriptomic Data of Human Embryoid Bodies (EBs). (a)** Cell Type Assignments of Each Location Inferred by RCTD. **(b)** EB Identity Assignments of Each Location. **(c)** Cell Type Composition within Each Whole EB.

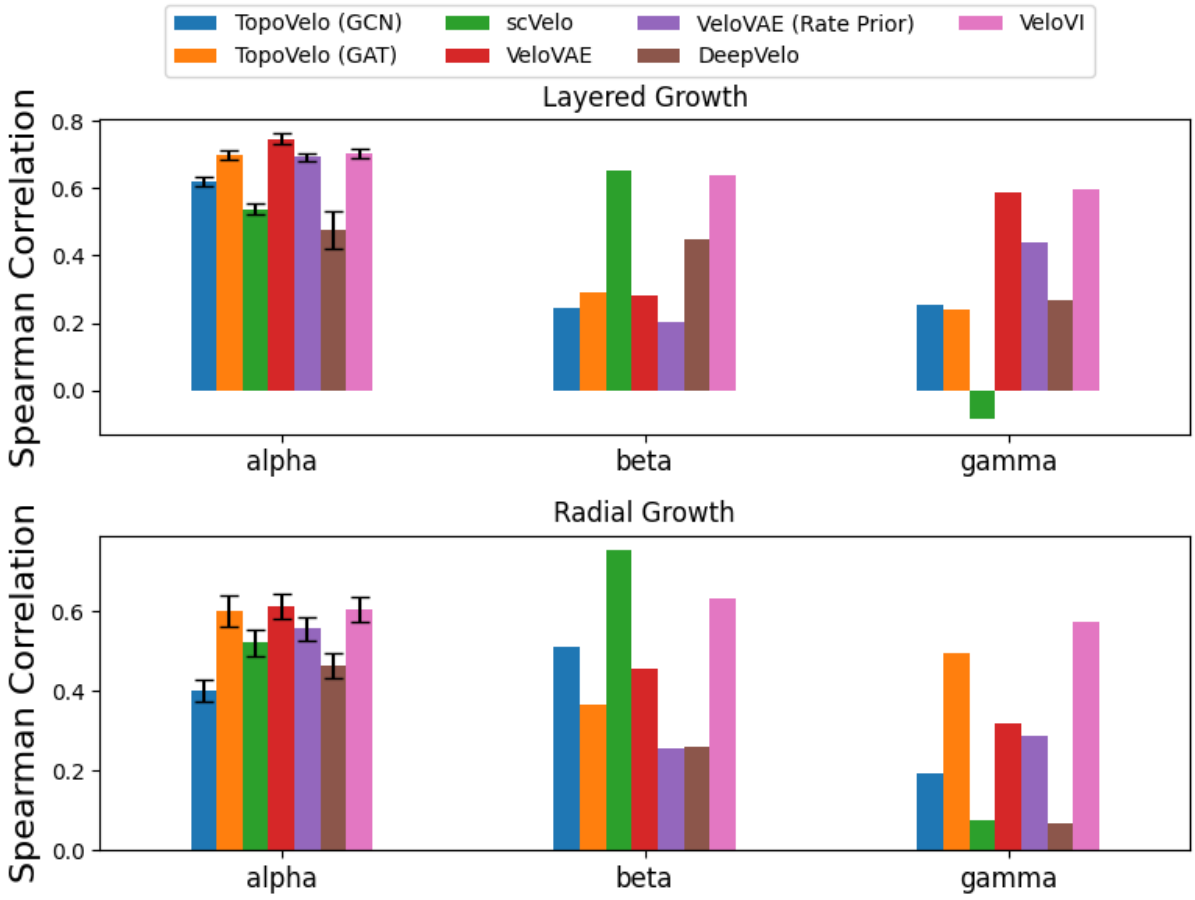

Figure B.3: **Evaluation of Rate Parameter Estimation on Simulated Datasets.** Each color bar represents a method. For the cell-specific gene transcription rate (alpha), each bar represents the mean metric and the error bar represents the standard deviation.

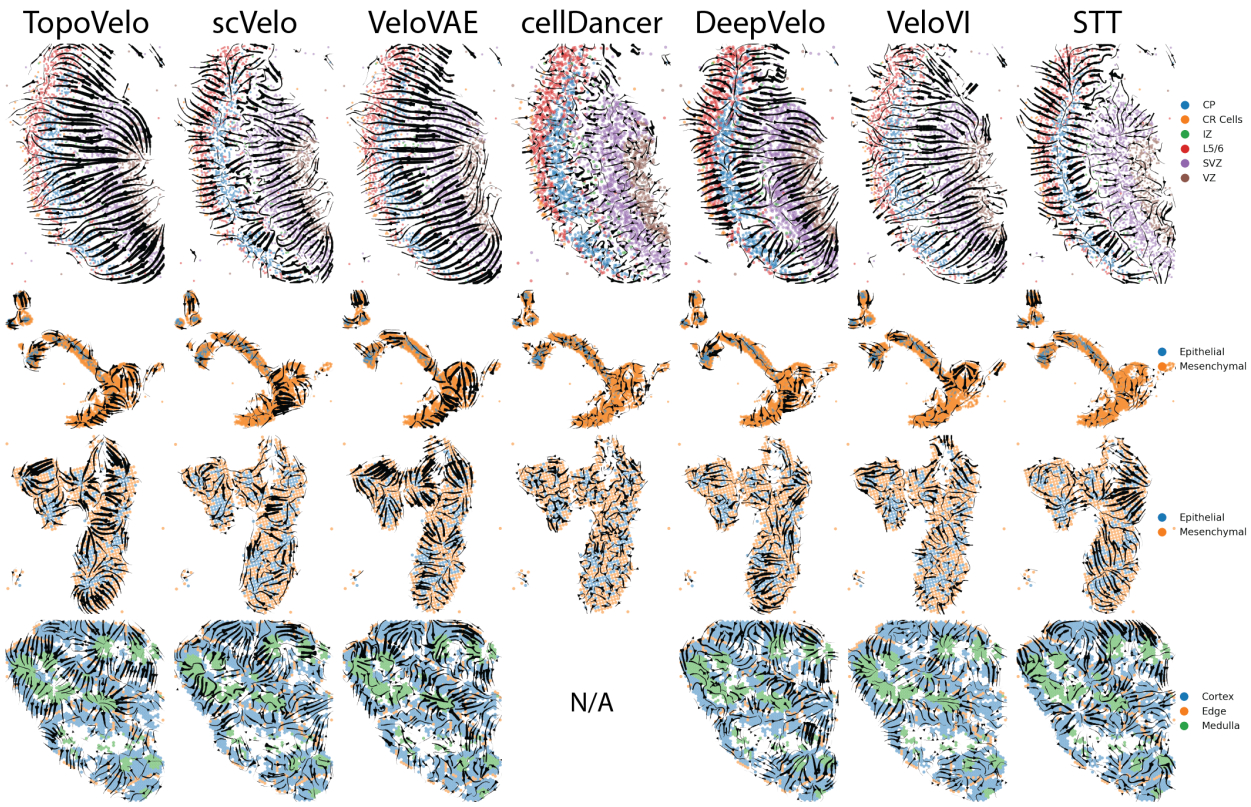

Figure B.4: **Comparison of Cell Velocity.** Each velocity stream plot visualizes the cell velocity inferred from a dataset (row) by a method (column). The stream plots, from top to bottom, are generated from Slide-seq E15 mouse brain cortex, Slide-seq E13.5 mouse gut and mouse lung, and Visium postnatal 3-month mouse thymus. Each column is a separate RNA velocity method shown at the top. "N/A" means a method failed to run on a certain datast.

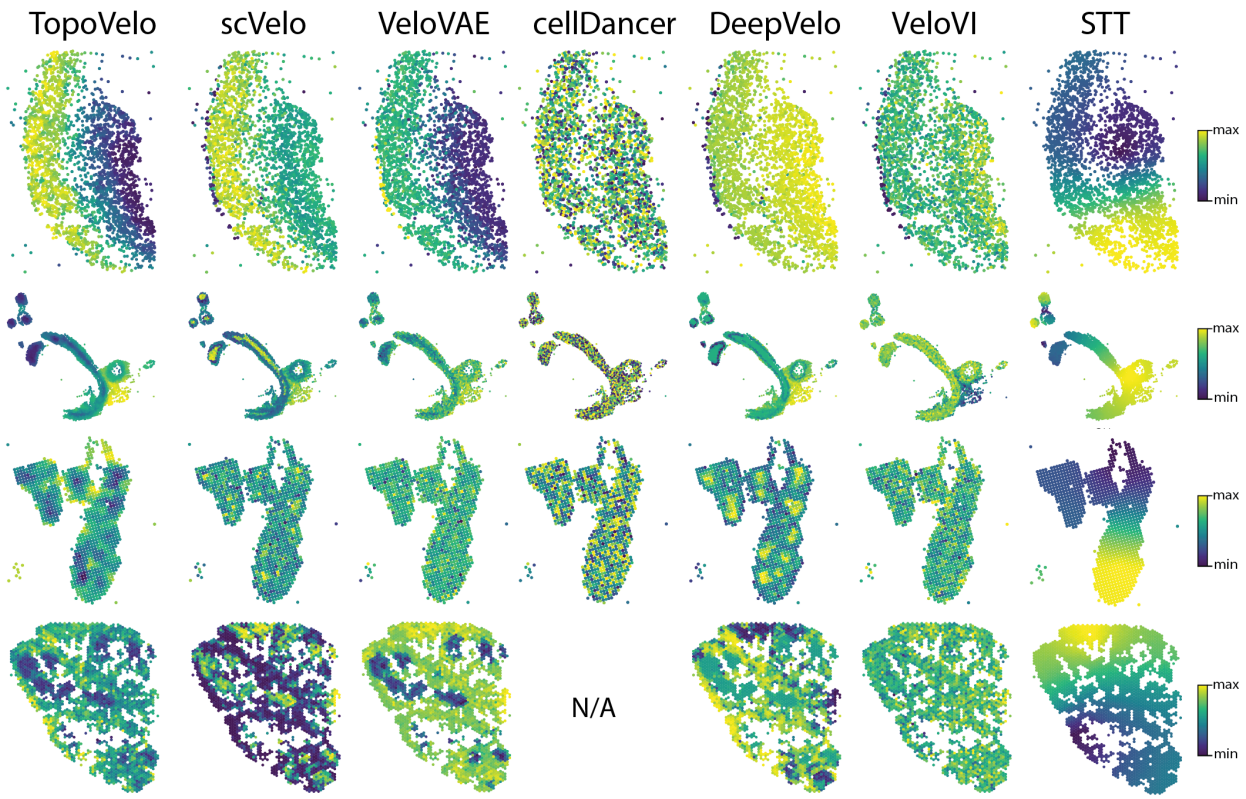

Figure B.5: **Comparison of Inferred Cell Time.** Each heatmap visualizes the cell time inferred from a dataset (row) by a method (column). The stream plots, from top to bottom, are generated from Slide-seq E15 mouse brain cortex, Slide-seq E13.5 mouse gut and mouse lung, and Visium postnatal 3-month mouse thymus. Each column is a separate RNA velocity method shown at the top. "N/A" means a method failed to run on a certain dataset.

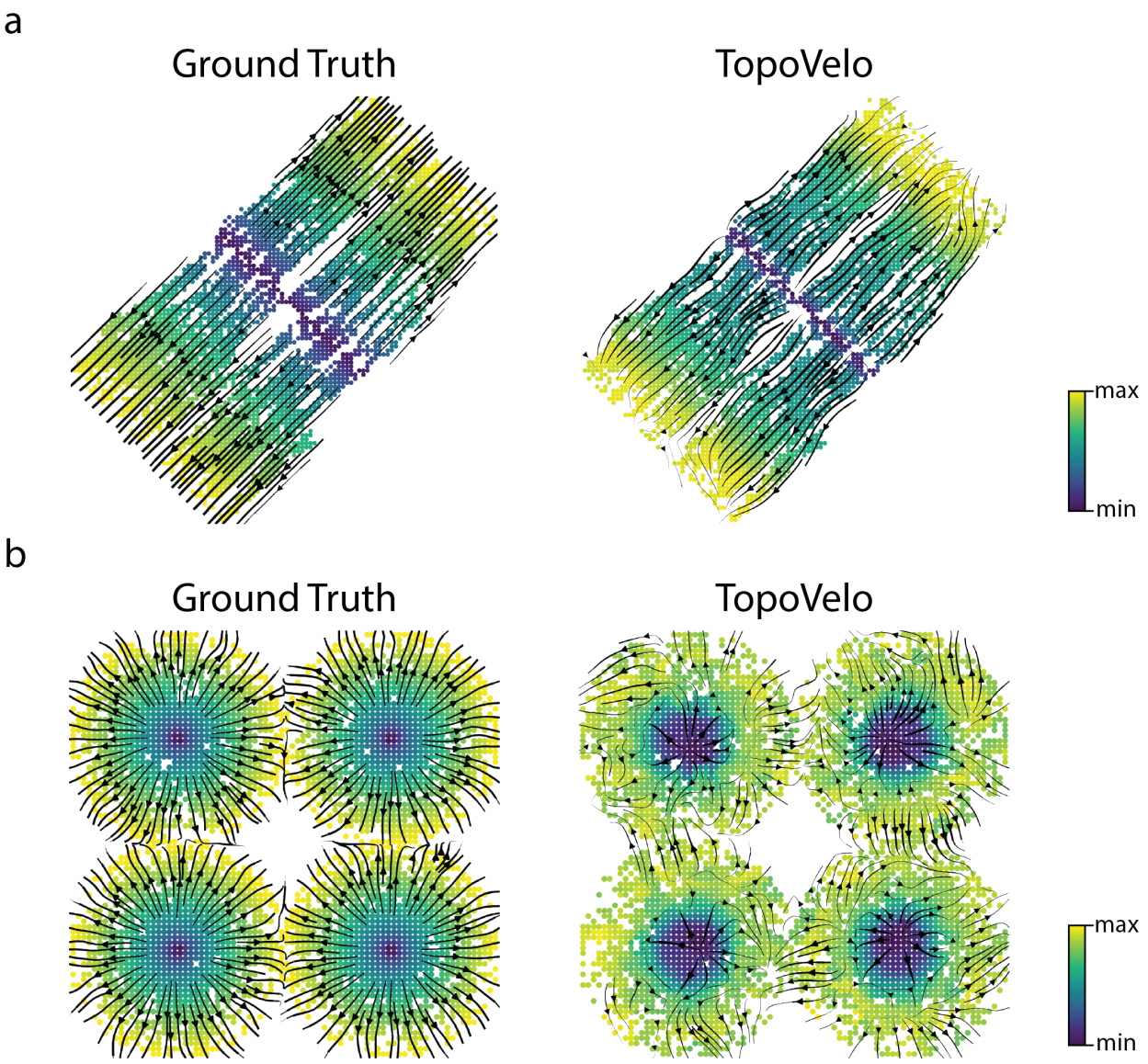

Figure B.6: **TopoVelo Robustly Models Spatial Dynamics in Simulated Datasets with Low Spatial Resolution.** Simulated datasets with layered growth patterns **(a)** and radial growth patterns **(b)**. The true velocity stream is plotted over spatial coordinates colored by the true time values. Note that these stream plots indicate cell velocity in physical coordinates, not RNA velocity in abstract UMAP coordinates.

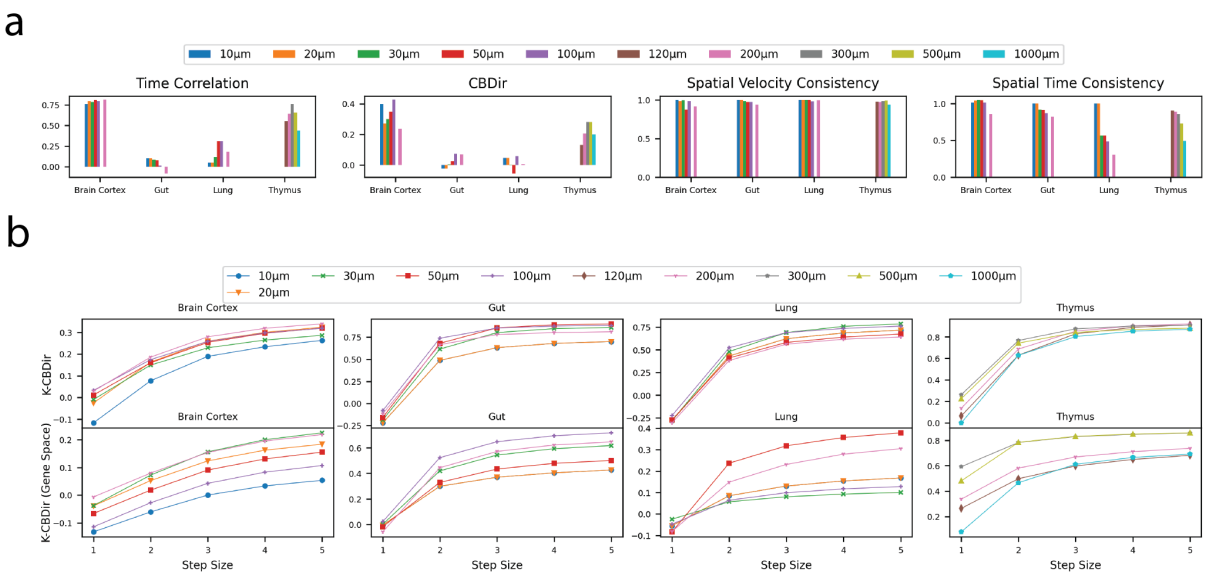

Figure B.7: **Sensitivity of TopoVelo Performance to Epsilon-Ball Graph Radius.** **(a)** Bar plots comparing time correlation, cross-boundary direction correctness, spatial velocity consistency and spatial time consistency from TopoVelo with trained with different radii. **(b)** Line plots showing the k-step CBDir (K-CBDir) metric. Each line with distinct marker and color represents one radius choice. x-axis is the step number and y-axis is the K-CBDir value.

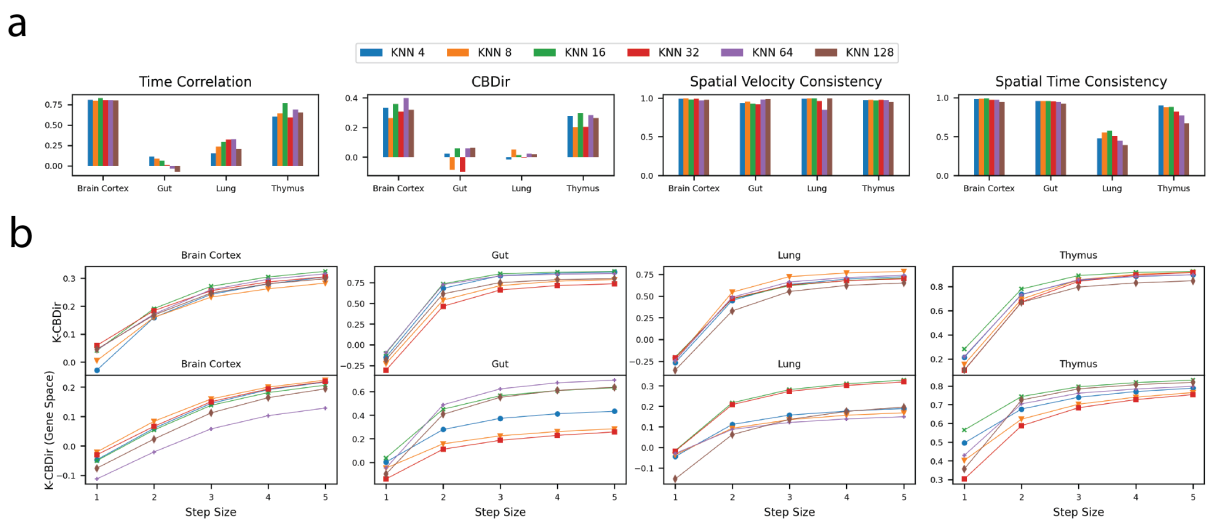

Figure B.8: **Sensitivity of TopoVelo Performance to KNN Neighborhood Size. (a)** Bar plots comparing time correlation, cross-boundary direction correctness, spatial velocity consistency and spatial time consistency from TopoVelo with trained with different KNN neighborhood size. **(b)** Line plots showing the k-step CBDir (K-CBDir) metric. Each line with distinct marker and color represents one KNN graph choice. x-axis is the step number and y-axis is the K-CBDir value.

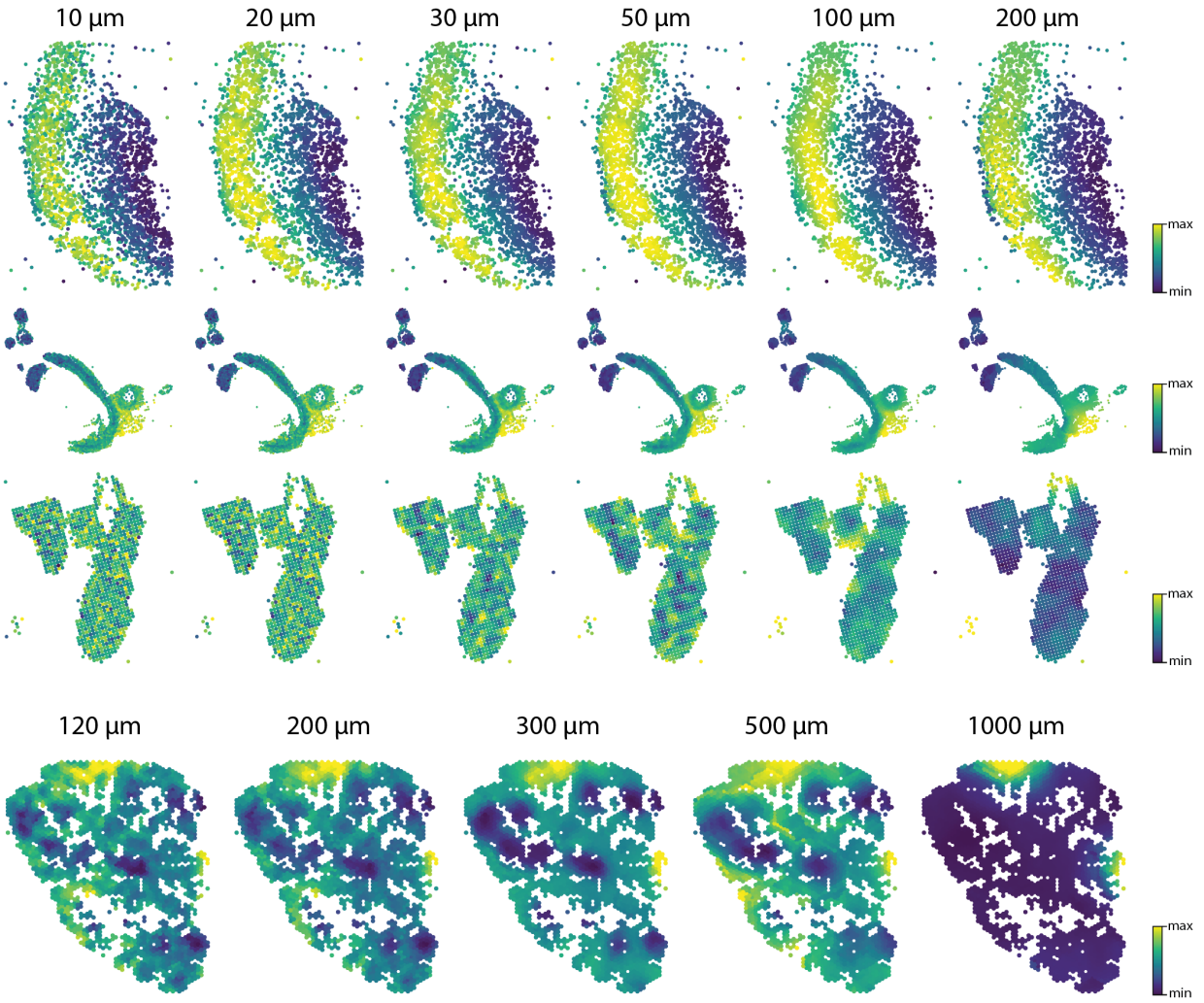

Figure B.9: **Sensitivity of Inferred Cell Time to Epsilon-ball Graph Radius.** Each panel is a heatmap showing the inferred cell time using an epsilon-ball graph with the given radius shown at the top. For the Visium thymus dataset, a different set of radius values are used due to a lower spatial resolution.

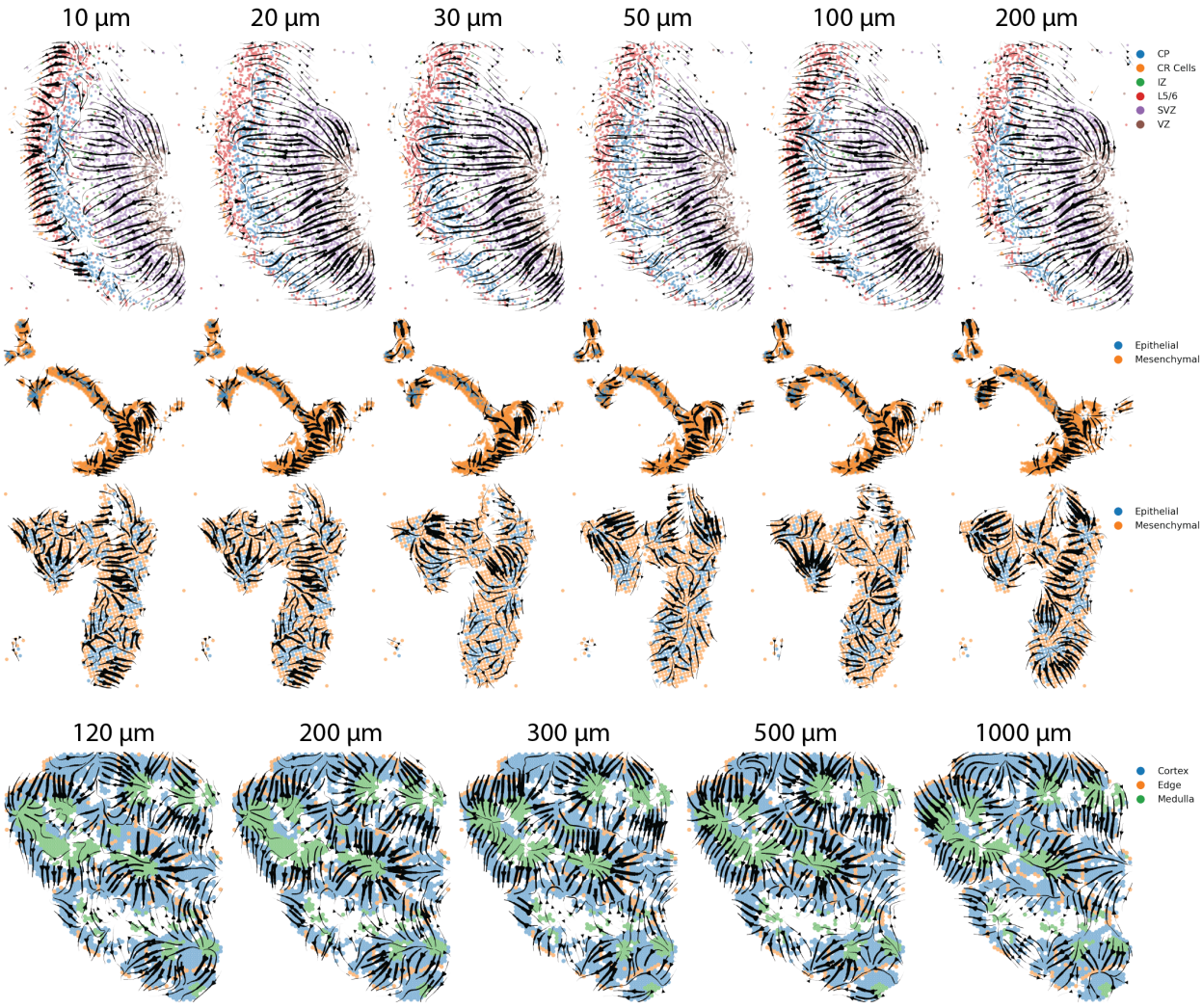

Figure B.10: **Sensitivity of Cell Velocity to Epsilon-ball Graph Radius.** Each panel is a velocity stream plot showing the cell velocity on the physical coordinates. The radius is shown at the top. For the Visium thymus dataset, a different set of radius values are used due to a lower spatial resolution.

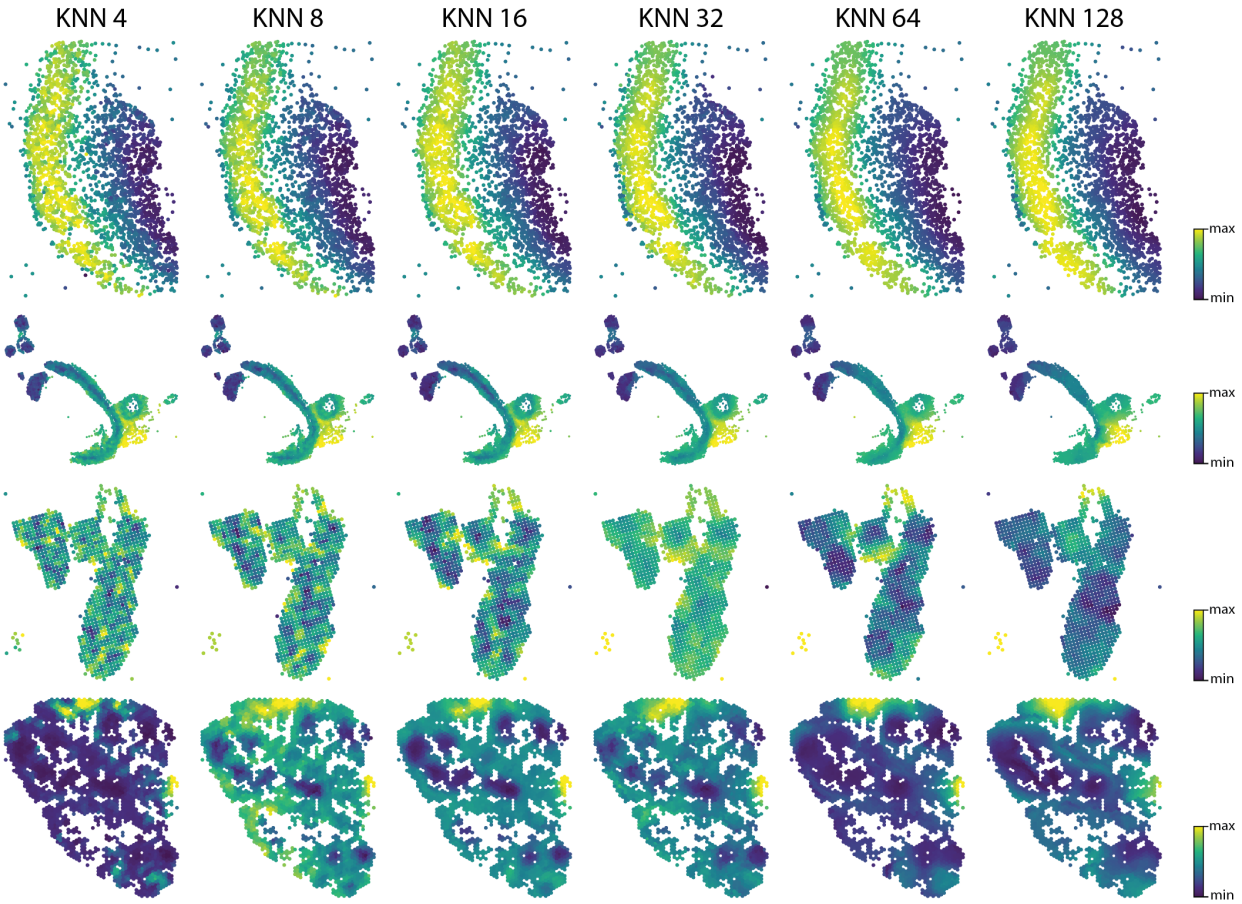

Figure B.11: **Sensitivity of Inferred Cell Time to KNN Neighborhood Size.** Each panel is a heatmap showing the inferred cell time using a spatial KNN graph with the number of neighbors shown at the top.

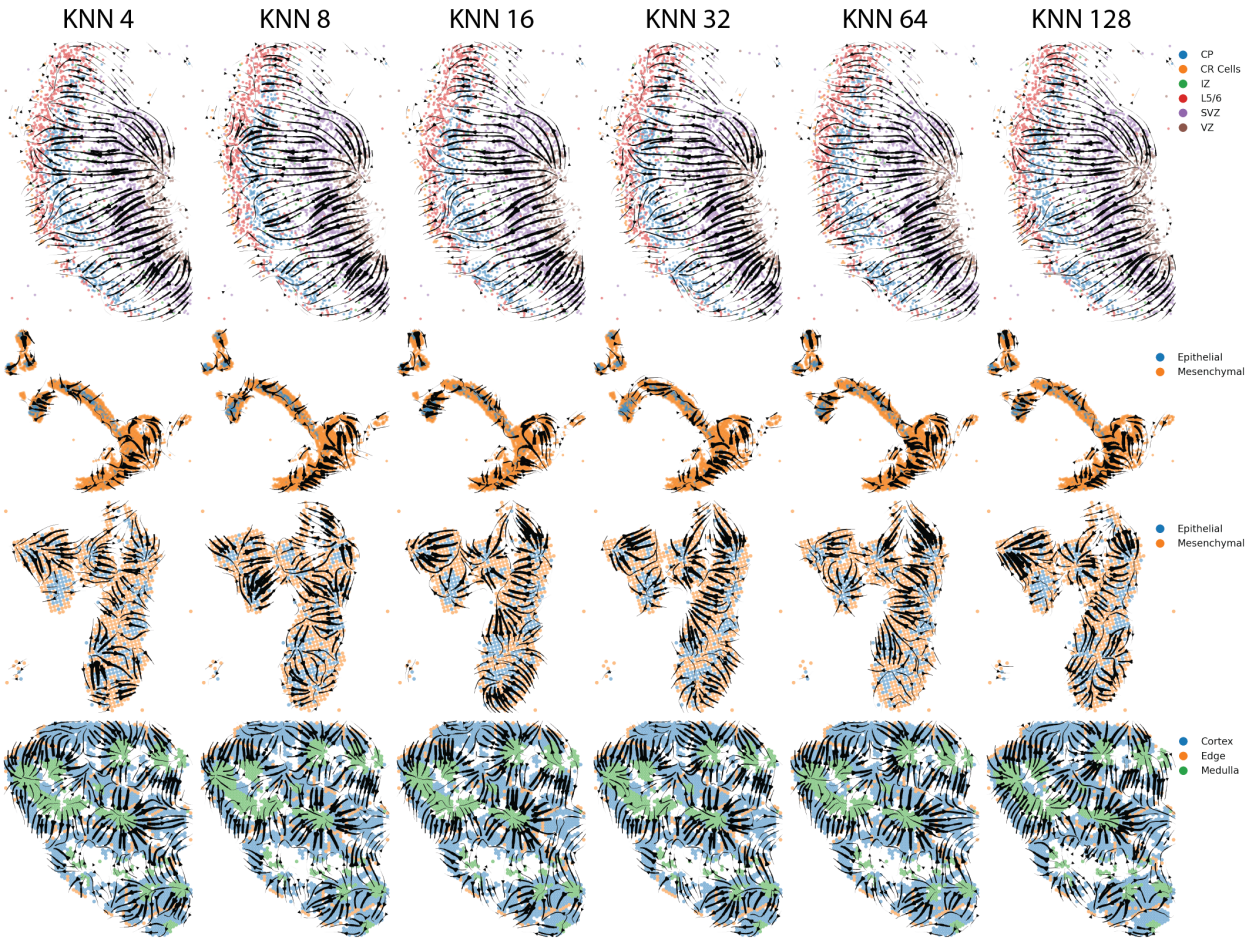

Figure B.12: **Sensitivity of Cell Velocity to KNN Neighborhood Size.** Each panel is a velocity stream plot showing the cell velocity on the physical coordinates. The number of neighbors of the spatial KNN graph is shown at the top.

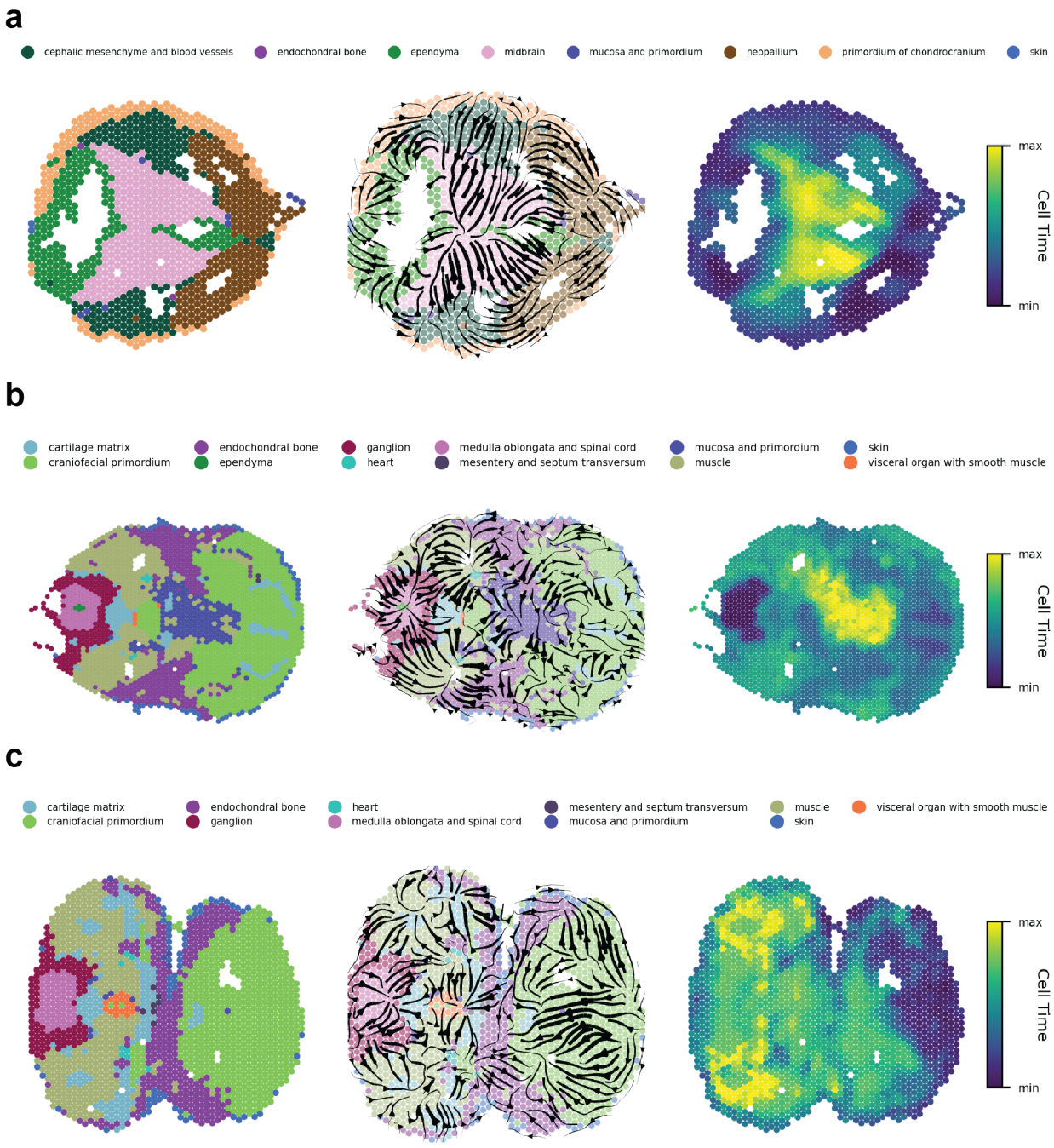

Figure B.13: **Results on Visium Data (Part 1).** Results from the first **(a)**, fourth**(b)** and fifth**(c)** slices from a 10X Visium mouse embryo dataset. Each panel shows the spatial coordinates colored by cell type annotations, cell velocity stream plot and a heatmap plot showing the developmental time inferred by TopoVelo.

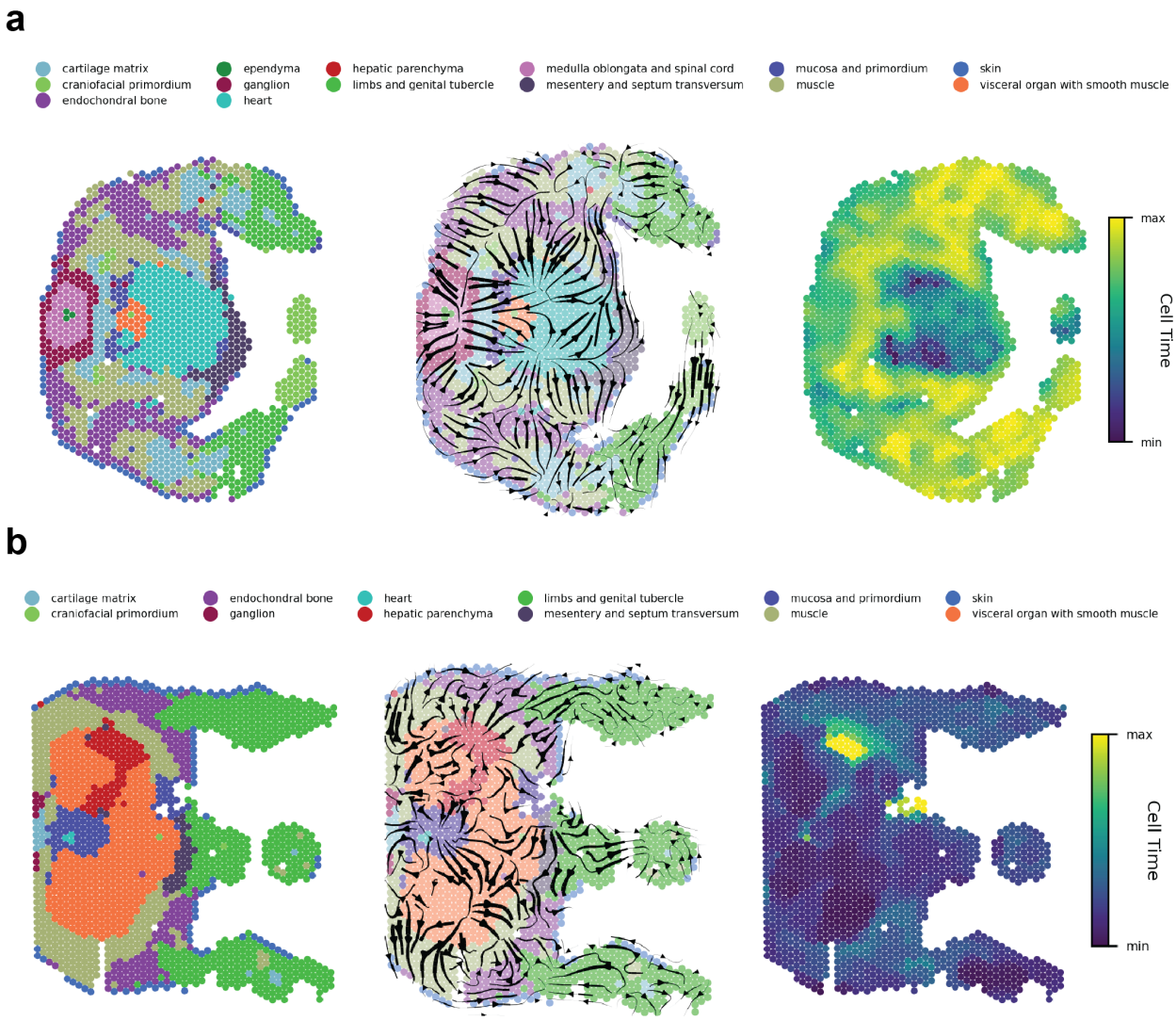

Figure B.14: **Results on Visium Data (Part 2).** Results from the sixth **(a)**, and nineth**(b)** slices from a 10X Visium mouse embryo dataset. Each panel shows the spatial coordinates colored by cell type annotations, cell velocity stream plot and a heatmap plot showing the developmental time inferred by TopoVelo.

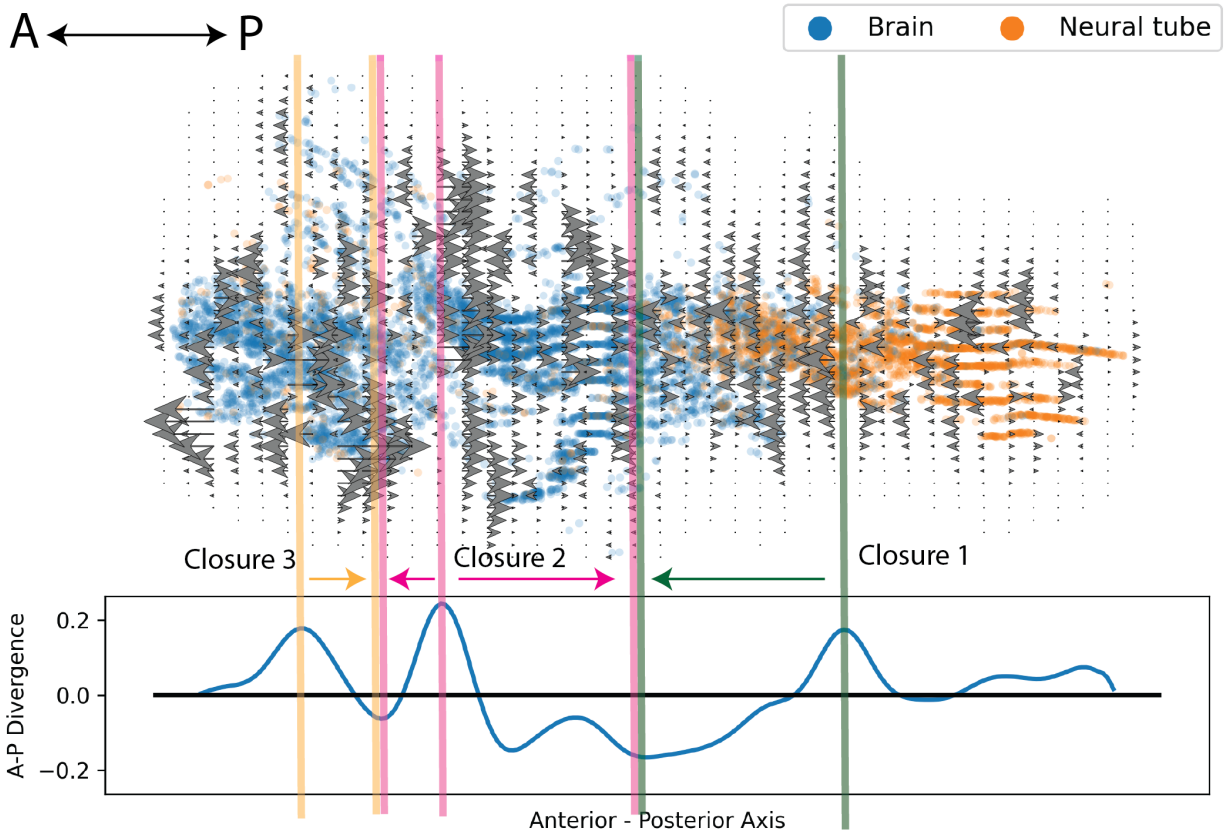

Figure B.15: **Quantifying Mouse Neural Tube Closure with Cell Velocity. Top**: 2D quiver plot showing cell velocity on the transformed anterior-posterior(A-P) axis of a E9 mouse neural tube. **Bottom**: 1D divergence of cell velocity on the A-P axis.

APPENDIX C

Supplementary Materials for Chapter 4

C.1 Space Complexity of SPN-FG

We provide a theoretical bound on the space complexity of an SPN-FG model.

Notation As introduced in **Section 4.4.1**, an SPN-FG model contains partition variables $\boldsymbol{Y} = \{Y_i : i \in [n]\}$ and connection matrix $\boldsymbol{B} \in \{0,1\}^{n \times m}$ parameterized by sum-product networks (SPN). We use the following notation throughout the derivation.

1. n: number of graph nodes.
2. m: number of factors.
3. l: SPN layer index
4. p_l: number of partitions in the l-th layer of an SPN
5. u_l: number of sum or product nodes in each partition in the l-th layer of an SPN.
6. w: maximum number of bit sequences from a product node.
7. s: Total number of trainable parameters of a single SPN.
8. S: Total number of trainable parameters of an SPN-FG model.

We define model complexity as the total number of trainable parameters in an SPN-FG. In our implementation, the joint PMF of either a row or a column of $\boldsymbol{B}$ can be parameterized by a separate SPN. We consider the case of building an SPN for each row of $\boldsymbol{B}$, i.e., each SPN models the joint distribution of connections between one node and all factors. This results in the following general formula for trainable parameters:

$$S = n(m+1) + ns \tag{C.1}$$

The first part, $n(m+1)$, represents n categorical distributions with $m+1$ categories for modeling $\boldsymbol{Y}$. The second part, ns, represents n SPNs, each having s parameters and modeling a single row of $\boldsymbol{B}$. Later, we will see that the space complexity stays the same when we choose to parameterize each column of $\boldsymbol{B}$ with an SPN. Notice that s is a function of m, n and w. Next, we derive bounds of s.

Special Case Here, we consider a special case of both m and w being a power of 2. Suppose $m = 2^d$ and $w = 2^k$. This is also the assumption in the original algorithm by Shih et al.[230].

We build an SPN by sequentially adding either a product or a sum layer to the network. The algorithm by Shih et al. keeps adding product layers until the Cartesian product of two partitions has a size exceeding the bound w. Here, we further assume $w < 2^m$ because if it's not the case, the width bound, w, has no effect and the SPN will be equivalent to a categorical distribution over all 2^m binary vectors. Once the width of an SPN exceeds w, we add sum and product layers alternatingly. Each sum node constraints the partition size u_l to be w, while each product node always combines two sets of w sequences into w^2 combinations. That is, we have

$$p_l = \begin{cases} \frac{1}{2}p_{l-1} & l \leq L_0 \\ p_{l-1} & l > L_0, l - L_0 \text{ odd (sum)} \\ \frac{1}{2}p_{l-1} & l > L_0, l - L_0 \text{ even (product)} \end{cases} \tag{C.2}$$

$$u_l = \begin{cases} u_{l-1}^2 & l \leq L_0 \\ w & l > L_0, l - L_0 \text{ odd (sum)} \\ w^2 & l > L_0, l - L_0 \text{ even (product)} \end{cases} \tag{C.3}$$

Here $L_0 + 1$ is the lowest index of the layer whose partition size exceeds the budget w, i.e. $L_0 := \max_l u_l \leq w$. Using **Eq. C.2-C.3**, we have $u_l = 2^{2^l}$ when $l < L_0$. This implies

$$L_0 := \max\{l : 2^{2^l} \leq 2^k\} \implies L_0 = \lfloor \log_2 k \rfloor \tag{C.4}$$

The trainable parameters of our SPN are the mixture weights of sum nodes and in each sum layer, the number of sum nodes equals the number of partitions times number of nodes

for each partition. Therefore, the total number of trainable parameters of each SPN equals:

$$s = p_{L_0+1} \cdot u_{L_0+1} + \sum_{l'=1}^{d-L_0-1} p_{L_0+2l'+1} \cdot u_{L_0+2l'+1} \tag{C.5}$$

$$= 2^{2^{L_0+1}} \frac{m}{2^{L_0+1}} + \sum_{l'=1}^{d-L_0-1} \frac{m}{2^{L_0+1+l'}} w^2 \tag{C.6}$$

$$= 2^{2^{L_0+1}-L_0-1} m + mw^2 \left(\frac{1}{2^{L_0+1}} - \frac{1}{2^d} \right) \tag{C.7}$$

From **Eq. C.4**, we have

$$\begin{aligned} & \log_2 k - 1 < L_0 \leq \log_2 k \\ \iff & \frac{k}{2} < 2^{L_0} \leq k \\ \iff & \sqrt{w} = 2^{\frac{k}{2}} < 2^{2^{L_0}} \leq 2^k = w \end{aligned} \tag{C.8}$$

By plugging the upper and lower bound in the above inequality into **Eq. C.7**, we have

$$\frac{mw}{2 \log w} + mw^2 \left(\frac{1}{2 \log w} - \frac{1}{m} \right) < s < mw^2 \left(\frac{2}{\log w} - \frac{1}{m} \right) \tag{C.9}$$

$$\implies s = \mathbf{\Theta} \left(\frac{mw^2}{\log w} \right) \tag{C.10}$$

$$\implies S = n(m+1) + ns = \mathbf{\Theta} \left(\frac{mnw^2}{\log w} \right) \tag{C.11}$$

When each SPN models a column of $\boldsymbol{B}$ instead of a row, we have $s = \mathbf{\Theta}(\frac{nw^2}{\log w})$ and hence,

$$S = n(m+1) + ms = \mathbf{\Theta} \left(\frac{mnw^2}{\log w} \right)$$

Finally, we conclude that

$$S = \mathbf{\Theta} \left(\frac{mnw^2}{\log w} \right)$$

Now we consider the alternative way of modeling each column of $\boldsymbol{B}$ with an SPN. Then, the total number of parameters becomes

$$S = n(m+1) + ms$$

Following exactly the same derivation with m replaced with n, we have each SPN $s = \mathbf{\Theta}(\frac{nw^2}{\log w})$

and the overall m parallel SPNs have a space complexity of $S = \boldsymbol{\Theta}\left(\frac{mnw^2}{\log w}\right)$. Hence, we end up with the same space complexity.

General Case Because the number of nodes or factors cannot always be a power of two, we would like to extend the original algorithm by considering any m and n. Again, we first consider building n SPNs, each modeling the joint distribution of m entries in a row of $\boldsymbol{B}$. Our algorithm first decomposes m into its binary representation:

$$m = \sum_{i=0}^{r} b_i 2^i,\ b_i \in \{0, 1\}$$

We build an SPN for each 2^i entries, and finally use a product unit to concatenate them together - this comes at a price of not modeling the full joint distribution of all bits, but results in a convenient implementation and nice properties such as decomposability and smoothness. Denote $s(m)$ as the space complexity of an SPN with m input bits. In the special case above, we assume the width bound $w < 2^m$. When $w \geq 2^m$, the SPN is equivalent to a categorical distribution with a support on all possible binary vectors. Thus, we have

$$s(2^d) = \begin{cases} 2^{2^{L_0+1}-L_0-1}m + mw^2\left(\frac{1}{2^{L_0+1}} - \frac{1}{2^d}\right) & w < 2^m \\ 2^m & w \geq 2^m \end{cases}$$

We assume $2^d \leq m < 2^{d+1}$ and $w = 2^k$. Then, we have

$$s(m) = \sum_{r=0}^{d} b_r s(2^r) + \mathbf{1}_{\{2^r > k \,\wedge\, r < d\}} \cdot w \tag{C.12}$$

Here, $\mathbf{1}_{\{\cdot\}}$ is the indicator function. The right hand side of **Eq. C.12** has two parts. The first part is just sum of all sub-networks. The second part accounts for the fact that $\forall r < d$, the SPN with 2^r nodes already reaches the end with w^2 combinations and instead of using a single sum node to select one out of w^2 nodes, w sum nodes will select one out of every w inputs, and only w outputs come out of the sum layer. Then, a final layer will select one out of the w outputs to output a single value. On the contrary, the largest SPN with 2^d nodes will also reach the last product layer with w^2 nodes, but then, by the algorithm, the w^2 outputs will go through a final layer with a single sum node to select one out of w^2 inputs, so it has w fewer parameters than the other sub-networks with 2^r nodes ($r < d$).

Using the previous result, we can get an upper and lower bound of $s(m)$.

$$\begin{aligned} s(m) &= \sum_{r=0}^{d} b_r s(2^r) + \mathbf{1}_{\{2^r > k \,\wedge\, r < d\}} \cdot w \\ &\geq s(2^d) \\ &\geq \frac{2^d w}{2\log_2 w} + 2^d w^2 \left(\frac{1}{2\log_2 w} - \frac{1}{2^d}\right) \\ &\geq \frac{(\frac{m}{2}+1)w}{2\log_2 w} + (\frac{m}{2}+1)w^2 \frac{2}{\log_2 w} - w^2 \end{aligned} \tag{C.13}$$

The upper bound depends on $L_0 = \lfloor \log_2 k \rfloor$ because small networks do not need sum layers to pre-select outputs from product nodes.

$$\begin{aligned} s(m) &= \sum_{r=0}^{d} b_r s(2^r) + \mathbf{1}_{\{2^r > k \,\wedge\, r < d\}} \cdot w \\ &\leq \sum_{r=0}^{L_0} 2^{2^r} + \sum_{r=L_0+1}^{d} s\left(2^r\right) + (d - L_0 - 1) \cdot w \\ &\leq \sum_{r=0}^{L_0} 2^{2^r} + \sum_{r=L_0+1}^{d} 2^r w^2 \left(\frac{2}{\log_2 w} - \frac{1}{2^r}\right) + (d - L_0 - 1) \cdot w \\ &\leq 2^{2^{L_0}} \cdot L_0 + \left(2^{d+2} - 2^{L_0+2}\right) \frac{w^2}{\log_2 w} - (d - L_0)\, w^2 + (d - L_0 - 1)w \\ &< w\left[\log_2 \log_2(w)\right] + \frac{(4m - 2\log_2 w)w^2}{\log_2 w} - \left(\log_2(m) - 1 - \log_2 \log_2(w)\right) w^2 \\ &\quad + (\log_2(m) - \log_2 \log_2(w))w \\ &= w \log_2(m) + \frac{(4m - 2\log_2 w)w^2}{\log_2 w} - \left(\log_2(m) - 1 - \log_2 \log_2(w)\right) w^2 \end{aligned} \tag{C.14}$$

Therefore, we have $s(m) = \mathbf{\Theta}(\frac{mw^2}{\log w})$ and consequently

$$S = \mathbf{\Theta}\left(\frac{mnw^2}{\log w}\right)$$

C.2 Identifiability of Causal DAGs by ABCDEFG

In this section, we introduce key concepts from the existing literature [145, 149, 151] and prove the identifiability of our method. Previously, Yang et al. introduced the concept of $\mathcal{I}$-Markov equivalence as an extension of Markov equivalence. Brouillard et al. proved the

identifiability of $\mathcal{I}$-Markov equivalent graphs under score maximization. Later, Lopez et al. provided a sufficient condition for a causal DAG to be unique given its corresponding f-DAG. Here, we extend the theory of causal discovery for DAGs and f-DAGs by showing (1) a derivation of the variational Bayes approach to causal discovery, (2) the identifiability of $\mathcal{I}$-Markov equivalent causal graphs under ELBO maximization, and (3) a sufficient and necessary condition for the equivalence between $\mathcal{I}$-Markov equivalence of f-DAGs and $\mathcal{I}$-Markov equivalence of their half-square graphs.

C.2.1 Theoretical Foundation for Bayesian Causal Discovery of Factor DAGs

We first introduce concepts about causal discovery and factor DAG as from DCDI[149] and DCD-FG[151].

Definition C.1 (Lopez et al.[151])**.** *Given a set of nodes, V, and factors, F, a factor directed acyclic graph (f-DAG), denoted as (V, F, E), is a directed acyclic graph $(V \cup F, E)$ where edges $E \subseteq \{(i, j) : i \in V, j \in F \text{ or } i \in F, j \in V\}$.*

An f-DAG is a DAG with two different types of vertices, nodes and factors. All edges connect two vertices of different types. Alternatively, if we represent an f-DAG using an adjacency matrix $\mathbf{A}$, we can use $\mathbf{U}$ and $\mathbf{V}$ to represent node-to-factor and factor-to-node adjacency matricies. Then we have $\mathbf{A} = \mathbf{U} \circ \mathbf{V}$ where $\circ$ denotes the matrix Boolean product. Furthermore, we can condense an f-DAG to a node-only or factor-only graph as defined below.

Definition C.2 (Lopez et al.[151])**.** *Given an f-DAG, $D = (V, F, E)$, its half-square node graph is defined as $D^2[V] = (V, \{(i, j) : \exists f \in F, (i, f), (f, g) \in E\})$, and half-square factor graph is defined as $D^2[F] = (F, \{(f, g) : \exists i \in V, (f, i), (i, g) \in D\})$.*

A half-square graph essentially keeps all dependency relations between nodes in the original factor graph. The factors can be interpreted as intermediate nodes on the paths between causally-related observations. We also note that the mapping from the set of f-DAGs to half-square graphs is an surjection.

Given a set of causally related random variables $\boldsymbol{X} = \{X_1, \ldots, X_n\}$ with a causal graph G. A fundamental assumption of a causal DAG underlying $\boldsymbol{X}$ is the Markov property, which leads to a factorization of the joint distribution. Here, we denote $\boldsymbol{\pi_i}$ as the set of all parents of i in G.

Definition C.3 (Brouillard et al.[149]). *Let $G = (V, E)$ be a causal DAG with n nodes and $\mathcal{I}^* = \{I_k : k \in [l]\}$ be a set of interventions. We define $\mathcal{M}_{\mathcal{I}^*}(G)$ as the set of joint distributions factorized according to the Markov property, i.e. $\boldsymbol{\mathcal{M}}_{\mathcal{I}^*}(G) := \{\{p^{(k)} : k \in [n^{\mathcal{I}^*}]\} : p^{(k)}(\boldsymbol{X}) = \prod_{i=1}^{n} p^{(k)}(X_i|\boldsymbol{X}_{\boldsymbol{\pi_i}})\}$*

By convention, $I_1 = \emptyset$ represents a pure observational setting.

Based on the definition above, Yang et al.[145] defined a type of equivalence relation called $\mathcal{I}$-Markov equivalence relation to describe DAG equivalence under interventions.

Definition C.4 ($\mathcal{I}$-Markov Equivalence[145]). *Two DAGs G_1 and G_2 are $\mathcal{I}$-Markov equivalence if and only if $\boldsymbol{\mathcal{M}}_{\mathcal{I}}(G_1) = \boldsymbol{\mathcal{M}}_{\mathcal{I}}(G_2)$. We denote by $\mathcal{I}$-MEC(G) as the set of all DAGs which are $\mathcal{I}$-Markov equivalent to G.*

In the rest of section B, we use the notation $\simeq_{\mathcal{I}}$ to denote $\mathcal{I}$-Markov equivalence relation.

Since we consider the set of f-DAGs, the causal relations between i and j are passed through latent factors. Denote $\boldsymbol{\pi}_{\boldsymbol{i}}^{\boldsymbol{D}}$ as the set of parents of a vertex i(node or factor) in the graph D. Next, we use a continuous random variable $\boldsymbol{Z} = \{Z_1, \ldots, Z_m\}$ to represent the factors. Then, we have a class of joint distributions of $\boldsymbol{X}$ and $\boldsymbol{Z}$ produced by an f-DAG.

Definition C.5 (Family of Distributions associated with an f-DAG). *Let $D = (V, F, E)$ be an f-DAG with n nodes and m factors and $\mathcal{I}^* = \{I_1, \ldots, I_k\}$ be a set of interventions. Then, $\boldsymbol{\mathcal{M}}_{\mathcal{I}^*}(D)$ is defined as the set of probabilistic models with the following form:*

$$\boldsymbol{\mathcal{M}}_{\mathcal{I}^*}(D) = \left\{ \{p^{(k)}(\boldsymbol{X}, \boldsymbol{Z}) : k \in [n^{\mathcal{I}^*}]\} : p^{(k)}(\boldsymbol{X}, \boldsymbol{Z}) = \prod_{i=1}^{n} p^{(k)}(X_i|\boldsymbol{Z}_{\boldsymbol{\pi}_{\boldsymbol{i}}^{\boldsymbol{D}}}) \prod_{j=1}^{m} p^{(k)}(Z_j|\boldsymbol{X}_{\boldsymbol{\pi}_{\boldsymbol{j}}^{\boldsymbol{D}}}) \right\} \tag{C.15}$$

where $p^{(k)}(X_i|\boldsymbol{Z}_{\boldsymbol{\pi}_{\boldsymbol{i}}^{\boldsymbol{D}}}) \neq p^{(1)}(X_i|\boldsymbol{Z}_{\boldsymbol{\pi}_{\boldsymbol{i}}^{\boldsymbol{D}}})$ if and only if $i \in I_k$ and $p^{(k)}(Z_j|\boldsymbol{X}_{\boldsymbol{\pi}_{\boldsymbol{j}}^{\boldsymbol{D}}}) \neq p^{(1)}(Z_j|\boldsymbol{X}_{\boldsymbol{\pi}_{\boldsymbol{j}}^{\boldsymbol{D}}})$ if and only if $j \in I_k$.

The above definition assumes knowledge of the intervention targets. When interventions are unknown, we are able to extend f-DAGs in a similar way to the $\mathcal{I}$-DAG introduced by Yang et al.[145]. We first mention the concept of $\mathcal{I}$-DAG and then extend it to f-DAGs.

Definition C.6 ($\mathcal{I}$-DAG[145]). *Let $G = (V, E)$ be a DAG and $\mathcal{I} = \{I_1, \ldots, I_{n^{\mathcal{I}}}\}$ be a set of interventions with $I_k \subseteq V, \forall k$. An interventional DAG ($\mathcal{I}$-DAG) is defined as an augmented graph*

$$G^{\mathcal{I}} = (V \cup \Xi, E \cup E^{\mathcal{I}})$$

where $\Xi := \{\xi_k : k \in [n^{\mathcal{I}}]\}$ is a set of intervention nodes representing $I_1, \ldots, I_k$ and $E^{\mathcal{I}} \subseteq \{(\xi_k, i) : i \in I_k, k \in [n^{\mathcal{I}}]\}$ is a set of edges from interventions to targets.

Definition C.7 (Extended f-DAG)**.** *Let $D = (V, F, E)$ be an f-DAG and $\mathcal{I} = \{I_1, \ldots, I_{n^{\mathcal{I}}}\}$ be a set of interventions. Let $\Xi = \{\xi_k, k \in [n^{\mathcal{I}}]\}$ be $n^{\mathcal{I}}$ nodes corresponding to the l interventions. An extended f-DAG is defined as an f-DAG $D^{\mathcal{I}} = (V \cup \Xi, F, E \cup E^I)$ where $E^{\mathcal{I}} \subseteq \{(\xi_k, f) : f \in F\}$, i.e. set of edges from intervention nodes to factors.*

An extended f-DAG is obtained by adding intervention nodes to an f-DAG. Here, we also have low-rank assumption that interventions causally affects downstream nodes via a small number of factors. Put in a matrix form, the adjacency matrix of an extended f-DAG has a low-rank Boolean matrix factorization as

$$\mathbf{A}^{\boldsymbol{\mathcal{I}}} = \begin{bmatrix} \mathbf{U} \\ \mathbf{W} \end{bmatrix} \circ \begin{bmatrix} \mathbf{V} \ \mathbf{0}_{\boldsymbol{m} \times \boldsymbol{n}^{\boldsymbol{\mathcal{I}}}} \end{bmatrix}$$

where $\mathbf{W} \in \mathbb{R}^{n^{\mathcal{I}} \times m}$ is an adjacency matrix representing edges from intervention nodes to factors.

Given the definition of $\boldsymbol{\mathcal{M}}_{\mathcal{I}^*}(D)$ and $\mathcal{I}$-Markov equivalence, we can further define $\mathcal{I}$-Markov equivalence relation between f-DAGs.

Definition C.8 ($\mathcal{I}$-Markov Equivalence Class of f-DAGs)**.** *Given a set of interventions, $\mathcal{I}$, two f-DAGs D_1 and D_2 are $\mathcal{I}$-Markov equivalent if $\boldsymbol{\mathcal{M}}_{\mathcal{I}}(D_1) = \boldsymbol{\mathcal{M}}_{\mathcal{I}}(D_2)$.*

The concept of $\boldsymbol{\mathcal{M}}_{\mathcal{I}}(D)$ and $\mathcal{I}$-Markov equivalence for f-DAGs are just the same as those for DAGs except for classifying vertices into nodes and factors.

The following theorem regarding the concept of $\mathcal{I}$-DAG connects statistical independence to graph structures.

Theorem C.9 (Yang et al.[145])**.** *Two DAGs G_1 and G_2 belong to the same $\mathcal{I}$-Markov Equivalence Class ($\mathcal{I}$-MEC) if and only if their $\mathcal{I}$-DAGs have the same skeleton and v-structures.*

Since f-DAGs are one type of DAG, we easily obtain the following corollary.

Corollary C.10. *Two f-DAGs D_1 and D_2 belong to the same $\mathcal{I}$-MEC if and only if their extended f-DAGs have the same skeleton and v-structures.*

Proof. Suppose D_1 and D_2 have n nodes and m factors. Let G_1 and G_2 be two DAGs obtained by removing the labeling of node or factor in D_1 and D_2. We still keep the bijection between vertices and random variables $\boldsymbol{X} = \{X_i : i \in [n]\}$ and $\boldsymbol{Z} = \{Z_j : j \in [m]\}$. Then, G_1 and G_2 are $\mathcal{I}$-Markov equivalent by Theorem C.9. By the definition of $\mathcal{I}$-Markov equivalence,

we have

$$\mathcal{M}_{\mathcal{I}}(G_1) = \mathcal{M}_{\mathcal{I}}(G_2) \implies p_1^{(k)}(\boldsymbol{X}, \boldsymbol{Z}) = p_2^{(k)}(\boldsymbol{X}, \boldsymbol{Z}) \forall k$$
$$\implies \mathcal{M}_{\mathcal{I}}(D_1) = \mathcal{M}_{\mathcal{I}}(D_2) \implies D_1 \in \mathcal{I}\text{-MEC } (D_2)$$

The first implication comes from the Markov property (d-separation in graphs implies conditional independence). The second implication comes from definition of extended f-DAGs. The last implication comes from the definition of $\mathcal{I}$-Markov equivalence class of f-DAGs. □

In reality, we can use a single encoder function to get $Z_j \sim p(f_{enc}(\mathbf{U_j} \odot \boldsymbol{X}; \boldsymbol{\Theta}))$ and $X_i \sim p(f_{dec}(\mathbf{V_i} \odot \boldsymbol{X}; \boldsymbol{\Phi}))$ to represent the conditional distribution $p^{(k)}(X_i|\boldsymbol{Z}_{\boldsymbol{\pi}_i^D})$ and $p^{(k)}(Z_j|\boldsymbol{X}_{\boldsymbol{\pi}_j^D})$. Thus, we define a second set of joint distributions representing our model capacity.

Definition C.11 (Family of Parametric Distributions associated with an f-DAG)**.** *Let $D = (V, F, E)$ be an f-DAG with n nodes and m factors. Consider two parametric functions $f_{enc} : \mathbb{R}^n \to \mathbb{R}^m$, parameterized by $\boldsymbol{\Theta} \in \Omega(\boldsymbol{\Theta})$ and $f_{dec} : \mathbb{R}^m \to \mathbb{R}^n$, parameterized by $\boldsymbol{\Phi} \in \Omega(\boldsymbol{\Phi})$. In addition, let $\mathbf{U}$ and $\mathbf{V}$ be node-to-factor and factor-to-node matrices of an f-DAG D. Then, $\boldsymbol{\mathcal{F}}_{\mathcal{I}^*}(D)$ is defined as the set of probabilistic models with the following form:*

$$\boldsymbol{\mathcal{F}}_{\mathcal{I}^*}(D) = \left\{ \{f^{(k)}(\boldsymbol{X}, \boldsymbol{Z}) : k \in [n^{\mathcal{I}^*}]\} : f^{(k)}(\boldsymbol{X}, \boldsymbol{Z}) = \prod_{i=1}^{n} f^{(k)}(X_i|\boldsymbol{Z}_{\boldsymbol{\pi}_i^D}) \prod_{j=1}^{m} f^{(k)}(Z_j|\boldsymbol{X}_{\boldsymbol{\pi}_j^D}) \right\} \tag{C.16}$$

where $f^{(k)}(Z_j|\boldsymbol{X}_{\boldsymbol{\pi}_j^D}) = p(f_{enc}(\mathbf{U_j} \odot \boldsymbol{X}))$, $f^{(k)}(X_i|\boldsymbol{Z}_{\boldsymbol{\pi}_i^D}) = p(f_{enc}(\mathbf{V_i} \odot \boldsymbol{Z}))$, $f^{(k)}(X_i|\boldsymbol{Z}_{\boldsymbol{\pi}_i^D}) \neq f^{(1)}(X_i|\boldsymbol{Z}_{\boldsymbol{\pi}_i^D})$ if and only if $i \in I_k$ and $f^{(k)}(Z_j|\boldsymbol{X}_{\boldsymbol{\pi}_j^D}) \neq f^{(1)}(Z_j|\boldsymbol{X}_{\boldsymbol{\pi}_j^D})$ if and only if $j \in I_k$.

C.2.2 Derivation of Bayesian Framework for Differentiable Causal Discovery

Most existing works on differentiable causal discovery sets a parametric form of causal DAGs. That is usually achieved by treating each edge as a categorical random variable and draw samples using the Gumbel softmax trick[228], allowing automatic differentiation in backpropagation. However, none of the works connect such a probabilistic model with the objective of score maximization. Here, we present a Bayesian framework for differentiable causal discovery and show that it reduces to score maximization under a uniform prior over the space of DAGs.

Consider a set of causally related random variables $\boldsymbol{X} = \{X_i : i \in [n]\}$ and a random intervention set $I^* \subseteq [n]$. First, we assume the observations are generated from a single

causal graph G^* via a generative model $p(\boldsymbol{X}|G^*, I^*)$. We assume each intervention either removes edges towards targets (hard) or keep the same graph structure (soft). Thus, the generative model becomes $p(\boldsymbol{X}|G^*, I^*)$ under different interventions. When I^* is known, we can obtain a MAP estimate of G:

$$\hat{G} = \arg\max_{G \in \boldsymbol{\mathcal{G}}} p(G|\boldsymbol{X}, I^*) \tag{C.17}$$

In order to convert this optimization problem to a differentiable one, we consider a variational distribution $q(G)$ and optimizing a KL divergence instead:

$$\hat{G} = \arg\max_{G} q^*(G) \text{ where } q^*(G) = \arg\min_{q(G)} KL(q(G)||p(G|\boldsymbol{X}, I^*)) \tag{C.18}$$

Because we have control over $q(G)$, finding its mode will be easy. Directly optimizing $KL(q(G)||p(G|\boldsymbol{X}))$ suffers from the intractability problem since $p(G|\boldsymbol{X}) = \frac{p(\boldsymbol{X}|G)p(G)}{\sum_{G'} p(\boldsymbol{X}|G')p(G')}$ and the space of DAGs is super-exponential in the number of nodes. Thus, we can derive an alternative objective in the following form:

$$\begin{aligned}
& KL(q(G)||p(G|\boldsymbol{X}, I^*)) \\
=& \mathbb{E}_{p(\boldsymbol{X}, I^*)} \left[\mathbb{E}_{q(G)} \left[\log \frac{q(G)}{p(G|\boldsymbol{X}, I^*)} \right] \right] \\
=& \mathbb{E}_{p(\boldsymbol{X}, I^*)} \left[\mathbb{E}_{q(G)} \left[\log \frac{q(G)p(\boldsymbol{X}|I^*)}{p(\boldsymbol{X}|G, I^*)p(G|I^*)} \right] \right] \\
=& \mathbb{E}_{p(\boldsymbol{X}, I^*)} \left[\log p(\boldsymbol{X}|I^*) - \mathbb{E}_{q(G)} \left[\log p(\boldsymbol{X}|G, I^*) \right] + KL(q(G)||p(G|I^*)) \right] \\
\Longrightarrow & \min_{q(G)} KL(q(G) \,||\, p(G|\boldsymbol{X}, I^*)) \\
=& \max_{q(G)} \mathbb{E}_{p(\boldsymbol{X}, I^*)} \left[\mathbb{E}_{q(G)} \left[\log p(\boldsymbol{X}|G, I^*) \right] - KL(q(G)||p(G|I^*)) \right] \\
=& \max_{q(G)} ELBO(G)
\end{aligned} \tag{C.19}$$

In reality, $p(\boldsymbol{X}, I^*)$ is replaced with am empirical distribution from any dataset. For I^*, we can conduct additional experiments by perturbing some nodes I_k in the k-th experiment. For the empirical data distribution, we assume the data samples are generated from $p(\boldsymbol{X}|G^*)$ instead of $p(\boldsymbol{X})$. The data samples are not drawn from the marginal over $\boldsymbol{X}$ because we assume a single causal graph G^* underlying the data generative process. We use parametric models $f^{(k)}(X_i|\boldsymbol{Z}_{\boldsymbol{\pi}_i^D}; \boldsymbol{\Phi})$ and $f^{(k)}(Z_j|\boldsymbol{X}_{\boldsymbol{\pi}_j^D}; \boldsymbol{\Theta})$ for distributional fitting and $q(G; \boldsymbol{\Lambda})$ for graph fitting. In addition, we need to add an L1 regularization on G to account for the sparsity

constraint. Now the optimization problem becomes:

$$\sup_{\boldsymbol{\Lambda},\boldsymbol{\Phi}} \sum_{k=1}^{n^{\mathcal{I}}} \mathbb{E}_{p^{(k)}(\boldsymbol{X}|G^*)} \left[\mathbb{E}_{q(G;\boldsymbol{\Lambda})} \left[\log f^{(k)}(\boldsymbol{X}|G;\boldsymbol{\Phi}) \right] - KL(q^{(k)}(G;\boldsymbol{\Lambda})||p^{(k)}(G)) \right] - \lambda|G| \quad \text{(C.20)}$$

The objective function is similar to the one proposed in the VAE paper[171] except that we have a latent space of DAGs instead of a low-dimensional latent embedding. In addition, we assume interventions changes neither the prior graph distribution nor our variational posterior. The objective can be extended to that of a β-VAE:

$$\begin{aligned}
&\sup_{\boldsymbol{\Phi},\boldsymbol{\Lambda}} \sum_{k=1}^{n^{\mathcal{I}}} \mathbb{E}_{p^{(k)}(\boldsymbol{X}|G^*)} \left[\mathbb{E}_{q(G;\boldsymbol{\Lambda})} \left[\log f^{(k)}(\boldsymbol{X}|G;\boldsymbol{\Phi}) \right] \right] - \beta KL(q(G;\boldsymbol{\Lambda})||p(G)) - \lambda|G| \\
=&\sup_{\boldsymbol{\Lambda}} \mathbb{E}_{q(G;\boldsymbol{\Lambda})} \left[\sup_{\boldsymbol{\Phi}} \sum_{k=1}^{n^{\mathcal{I}}} \mathbb{E}_{p^{(k)}(\boldsymbol{X}|G^*)} \left[\log f^{(k)}(\boldsymbol{X}|G;\boldsymbol{\Phi}) \right] - \lambda|G| \right] - \beta KL(q(G;\boldsymbol{\Lambda})||p(G))
\end{aligned} \quad \text{(C.21)}$$

Notice that the score function is under the expectation of $q(G;\boldsymbol{\Lambda})$. If we set $\beta = 0$ and $q(G;\boldsymbol{\Lambda}) = \delta(G)$, the Dirac delta function, the optimization problem becomes exactly the same as a score maximization problem as presented in previous score-based methods. The constraint on $q(G;\boldsymbol{\Lambda})$ ensures that $q(G;\boldsymbol{\Lambda})$ does not deviate from the prior arbitrarily. Next, we will prove the identifiability of this Bayesian framework. It turns out to be a corollary from previous results in DCDI.

Theorem C.12 (Brouillard et al.[149])**.** *Let $\boldsymbol{X} = \{X_1, \ldots, X_n\}$ be a set of causally related random variables with a causal DAG $G^* = (V, E)$ and $\mathcal{I}^* = \{I_k : k \in [n^{\mathcal{I}^*}]\}$ be a set of interventions with $I_1 = \emptyset$. Assume the following:*

1. *The set of distributions from our parametric models contains the ground truth interventional distributions: $\{p^{(k)}(\boldsymbol{X}) : k \in [n^{\mathcal{I}^*}]\} \in \boldsymbol{\mathcal{F}}_{\mathcal{I}^*}(G^*)$ where $\boldsymbol{\mathcal{F}}_{\mathcal{I}^*}(G^*) = \{\{f^{(k)}(\boldsymbol{X}|G^*;\boldsymbol{\Phi})\} : \boldsymbol{\Phi} \in \Omega(\boldsymbol{\Phi})\}$.*

2. *Denote $\perp\!\!\!\perp_{G^*}$ as the d-separation relation in G^*. $\mathcal{I}$-faithfulness contains the following two conditions.*

   (a) *For any disjoint set $A, B, C \subset V$, $\boldsymbol{X_A} \perp\!\!\!\perp \boldsymbol{X_B}|\boldsymbol{X_C} \implies A \perp\!\!\!\perp_{G^*} B|C$*

   (b) *For any disjoint sets $A, C \subset V$ and $k \in [n^{\mathcal{I}^*}]$, $p^{(k)}(\boldsymbol{X_A}|\boldsymbol{X_C}) = p^{(1)}(\boldsymbol{X_A}|\boldsymbol{X_C}) \implies A \perp\!\!\!\perp_{G^{*\mathcal{I}^*}} \xi_k|C$*

3. *$\forall G, I, \boldsymbol{\Phi}, f^{(k)}(\boldsymbol{X}|G, I;\boldsymbol{\Phi}) > 0$.*

4. $\forall k \in [n^{\mathcal{I}^*}]$, $\left|\mathbb{E}_{p^{(k)}(\boldsymbol{X})}\left[\log p^{(k)}(\boldsymbol{X})\right]\right| < +\infty$.

Define the score function as

$$S_{\mathcal{I}^*}(G) = \sup_{\boldsymbol{\Phi}} \sum_{k=1}^{n^{\mathcal{I}^*}} \mathbb{E}_{p^{(k)}(\boldsymbol{X})}\left[\log f^{(k)}(\boldsymbol{X}|G;\boldsymbol{\Phi})\right] - \lambda|G|$$

Then, with a small enough $\lambda > 0$*, we have* $S_{\mathcal{I}^*}(G^*) > S_{\mathcal{I}^*}(G)$.

The previous theorem claims optimality of the score function when the causal DAG is treated as a deterministic object. Next, we give a probabilistic view of this optimality. First, we define the Bayesian score function as follows.

Definition C.13 (Bayesian Score Function)**.** *Let* $\boldsymbol{X} = \{X_1, \ldots, X_n\}$ *be a set of causally related random variables with a causal DAG* G^* *and* $\mathcal{I}^* = \{I_k : k \in [n^{\mathcal{I}}]\}$ *be a set of interventions with* $I_1 = \emptyset$*. Let* $p(G)$ *be a prior over DAGs and* $q(G;\boldsymbol{\Lambda})$ *be a variational distribution. The Bayesian score function,* $\boldsymbol{\mathcal{L}}(q(G;\boldsymbol{\Lambda}))$ *is defined as*

$$\boldsymbol{\mathcal{L}}(q(G;\boldsymbol{\Lambda})) = \mathbb{E}_{q(G;\boldsymbol{\Lambda})}\left[S_{\mathcal{I}^*}(G)\right] - \beta KL(q(G;\boldsymbol{\Lambda})||p(G))$$

where $S_{\mathcal{I}^*}(G)$ *is the score function defined in Theorem C.12.*

Theorem C.14 (Identifiability via ELBO maximization)**.** *Let* $\boldsymbol{X} = \{X_1, \ldots, X_n\}$ *be a set of causally related random variables with a causal DAG* G^* *and* $\mathcal{I}^* = \{I_k : k \in [n^{\mathcal{I}}]\}$ *be a set of interventions with* $I_1 = \emptyset$*. Let* $\boldsymbol{\mathcal{G}}$ *be a subset of all causal DAGs and* $q^*(G)$ *be an optimal graph distribution from the optimization problem:*

$$\sup_{q(G;\boldsymbol{\Lambda}):supp(q)\subseteq\boldsymbol{\mathcal{G}}} \boldsymbol{\mathcal{L}}(q(G;\boldsymbol{\Lambda})),$$

where

$$\boldsymbol{\mathcal{L}}(q(G;\boldsymbol{\Lambda})) = \mathbb{E}_{q(G;\boldsymbol{\Lambda})}\left[S_{\mathcal{I}^*}(G)\right] - \beta KL(q(G;\boldsymbol{\Lambda})||p(G)),$$
$$S_{\mathcal{I}^*}(G) = \sup_{\boldsymbol{\Phi}} \sum_{k=1}^{n^{\mathcal{I}}} \mathbb{E}_{p^{(k)}(\boldsymbol{X})}\left[\log f^{(k)}(\boldsymbol{X}|G;\boldsymbol{\Phi})\right] - \lambda|G|.$$

If $G^* \in \boldsymbol{\mathcal{G}}$*, then, under the same assumptions as those in Theorem C.12, for small enough* $\beta > 0$ *and small enough* $\lambda > 0$*,* $\hat{G} = \arg\max_G q^*(G)$ *is* $\mathcal{I}^*$*-Markov equivalent to* G^*.

Proof. We prove this theorem by contradiction. Suppose $\exists \hat{G} = \arg\max_G q^*(G)$ such that $\hat{G} \not\simeq_{\mathcal{I}^*} G^*$.

Consider another PMF $q'(G)$ which has the same support and same mass as $q^*(G)$ except for $q'(G^*) - q^*(G^*) = \epsilon > 0$ and consequently, $q'(\hat{G}) - q^*(\hat{G}) = -\epsilon < 0$. Because $q^*(\hat{G}) > 0$, such ϵ exists. By the definition of q^*, $\boldsymbol{\mathcal{L}}(q^*) \geq \boldsymbol{\mathcal{L}}(q')$. Then, we have

$$\begin{aligned}
\boldsymbol{\mathcal{L}}(q') - \boldsymbol{\mathcal{L}}(q^*) &= \left[\mathbb{E}_{q'(G)}\left[S_{\mathcal{I}^*}(G)\right] - \beta KL(q'(G)||p(G))\right] - \left[\mathbb{E}_{q^*(G)}\left[S_{\mathcal{I}^*}(G)\right] - \beta KL(q^*(G)||p(G))\right] \\
&= \sum_{G \in \boldsymbol{\mathcal{G}}} (q'(G) - q^*(G)) S_{\mathcal{I}^*}(G) + \beta \left[KL(q^*(G)||p(G)) - KL(q'(G)||p(G))\right] \\
&= \epsilon \left(S_{\mathcal{I}^*}(G^*) - S_{\mathcal{I}^*}(\hat{G})\right) + \beta \left[KL(q^*(G)||p(G)) - KL(q'(G)||p(G))\right] \qquad \text{(C.22)}
\end{aligned}$$

By Theorem C.12, $\exists \lambda > 0$ such that $S_{\mathcal{I}^*}(G^*) > S_{\mathcal{I}^*}(G), \forall G \not\simeq_{\mathcal{I}^*} G^*$. Therefore, $S_{\mathcal{I}^*}(G^*) - S_{\mathcal{I}^*}(\hat{G}) = \Delta > 0$. If $\sum_{k=1}^{n^{\mathcal{I}^*}} \left[KL(q^*(G)||p(G)) - KL(q'(G)||p(G))\right] \geq 0$, we already have $\boldsymbol{\mathcal{L}}(q') > \boldsymbol{\mathcal{L}}(q^*)$. Otherwise, we can pick

$$0 < \beta < \frac{\epsilon \Delta}{KL\left(q'(G)||p(G)\right) - KL\left(q^*(G)||p(G)\right)}$$

and $\boldsymbol{\mathcal{L}}(q') > \boldsymbol{\mathcal{L}}(q^*)$. Both cases contradict the fact that $\boldsymbol{\mathcal{L}}(q^*) \geq \boldsymbol{\mathcal{L}}(q')$. Therefore, we conclude that G^* must be a mode of q. □

Notice that we add a constraint on the support of $q(G; \boldsymbol{\Lambda})$ to account for cases when we have prior knowledge about the DAG and only need to search over a subset. As discussed below, this applies when the true causal DAG is a half-square graph of an f-DAG. If we set $\boldsymbol{\mathcal{G}}$ to the set of all DAGs, the constraint will be removed.

ABCDEFG aims at optimizing $KL(q(D^2[V]) \mid\mid p(G|\boldsymbol{X}))$ with respect to a distribution on f-DAGs instead of DAGs. As long as the adjacency matrix of the true causal DAG can be factorized as a Boolean product of a node-to-factor and factor-to-node matrices, optimization over f-DAGs guarantees identifiability of the true causal DAG, as a half-square graph of an optimal f-DAG.

C.2.3 Extension to Unknown Intervention Targets

Bayesian Framework for Interventional Causal DAG Discovery The Bayesian framework can be further extended to unknown intervention targets. Consider a set of causally related random variables $\boldsymbol{X} = \{X_i : i \in [n]\}$ and a random intervention set $I^* \subseteq [n]$. With the same assumptions as the Bayesian framework in **Section C.2.2**, our goal is to identify the true $\mathcal{I}$-DAG, $(G^*)^{I^*}$. Following a similar argument, we can convert MAP estimation

into a continuous optimization problem:

$$\hat{G}^{\mathcal{I}} = \underset{G^{\mathcal{I}} \in \boldsymbol{\mathcal{G}}^{\boldsymbol{\mathcal{I}}}}{\arg\max}\, q^*\left(G^{\mathcal{I}}\right) \text{ where } q^*\left(G^{\mathcal{I}}\right) = \underset{q(G^{\mathcal{I}})}{\arg\min}\, KL\left(q\left(G^{\mathcal{I}}\right) \,||\, p(G^{\mathcal{I}}|\boldsymbol{X})\right)$$

Alternatively, we can optimize the ELBO as follows:

$$\begin{aligned}
& KL(q(G^{\mathcal{I}})||p(G^{\mathcal{I}}|\boldsymbol{X})) \\
=& \mathbb{E}_{p(\boldsymbol{X})}\left[\mathbb{E}_{q(G^{\mathcal{I}})}\left[\log \frac{q(G^{\mathcal{I}})}{p(G^{\mathcal{I}}|\boldsymbol{X})}\right]\right] \\
=& \mathbb{E}_{p(\boldsymbol{X})}\left[\mathbb{E}_{q(G^{\mathcal{I}})}\left[\log \frac{q(G^{\mathcal{I}})p(\boldsymbol{X})}{p(\boldsymbol{X}|G^{\mathcal{I}})p(G^{\mathcal{I}})}\right]\right] \\
=& \mathbb{E}_{p(\boldsymbol{X})}\left[\log p(\boldsymbol{X}) - \mathbb{E}_{q(G^{\mathcal{I}})}\left[\log p(\boldsymbol{X}|G^{\mathcal{I}})\right] + KL(q(G^{\mathcal{I}})||p(G^{\mathcal{I}})\right] \qquad \text{(C.23)} \\
\Longrightarrow \quad & \min_{q(G^{\mathcal{I}})} KL(q(G^{\mathcal{I}})||p(G^{\mathcal{I}}|\boldsymbol{X})) \\
=& \max_{q(G^{\mathcal{I}})} \mathbb{E}_{q(G^{\mathcal{I}})}\left[\log p(\boldsymbol{X}|G^{\mathcal{I}})\right] - KL(q(G^{\mathcal{I}})||p(G^{\mathcal{I}})) \\
=& \max_{q(G^{\mathcal{I}})} ELBO(G^{\mathcal{I}}) \qquad \text{(C.24)}
\end{aligned}$$

The optimization problem is exactly the same as the one in **Section C.2.2**, except that we consider a distribution over an $\mathcal{I}$-DAG instead of a DAG. Similar to the derivation of **Thm. C.14**, we first present a theorem from Brouillard et al.[149].

Theorem C.15 ($\mathcal{I}$-DAG Identifiability of DCDI[149])**.** *Let $\boldsymbol{X} = \{X_1, \ldots, X_n\}$ be a set of causally related random variables with a causal DAG $G^* = (V, E)$ and $\mathcal{I}^* = \{I_k : k \in [n^{\mathcal{I}}]\}$ be a set of interventions with $I_1 = \emptyset$. Define the score function as*

$$S(G, \mathcal{I}) = \sup_{\boldsymbol{\Phi}} \sum_{k=1}^{n^{\mathcal{I}}} \mathbb{E}_{p^{(k)}(\boldsymbol{X})}\left[\log f^{(k)}(\boldsymbol{X}|G, \mathcal{I}; \boldsymbol{\Phi})\right] - \lambda|G| - \lambda_R|\mathcal{I}|$$

Then, under the same assumptions from Thm. C.12 and with a small enough $\lambda > 0$ and $\lambda_R > 0$, we have $S(G^*, \mathcal{I}^*) > S(G, \mathcal{I})$ for any $G \not\simeq_{\mathcal{I}^*} G^*$ or $\mathcal{I} \neq \mathcal{I}^*$.

In the implementation, the unknown interventions are parameterized by a binary matrix $R^{\mathcal{I}} \in \{0, 1\}^{n^{\mathcal{I}} \times n}$ where $R^{\mathcal{I}}_{kj} = 1$ if and only if $j \in I_k$. Next, we prove identifiability of our Bayesian framework.

Theorem C.16 (Identifiability for untargeted interventions via ELBO maximization)**.** *Let (1) $\boldsymbol{X} = \{X_1, \ldots, X_n\}$ be a set of causally related random variables with a causal DAG G^*, (2) $\mathcal{I}^* = \{I_k : k \in [n^{\mathcal{I}}]\}$ be a set of unobserved interventions with $I_1 = \emptyset$ and (3)*

$\boldsymbol{\mathcal{G}^{\mathcal{I}}}$ be the set of all $\mathcal{I}$-DAGs with $n^{\mathcal{I}^*}$ interventions. $\forall G^{\mathcal{I}} \in \mathcal{G}^{\mathcal{I}}$, define R as the adjacency matrix of intervention-to-node graph. Let $q^*(G^{\mathcal{I}})$ be an optimal graph distribution from the optimization problem:

$$\sup_{q(G^{\mathcal{I}};\boldsymbol{\Lambda}):supp(q)\subseteq\boldsymbol{\mathcal{G}^{\mathcal{I}}}} \boldsymbol{\mathcal{L}}(q(G^{\mathcal{I}};\boldsymbol{\Lambda}))$$

where

$$\boldsymbol{\mathcal{L}}(q(G^{\mathcal{I}};\boldsymbol{\Lambda})) = \mathbb{E}_{q(G;\boldsymbol{\Lambda})}\left[S(G,\mathcal{I})\right] - \beta KL\left(q(G,\mathcal{I}) \,||\, p(G,\mathcal{I})\right)$$
$$S(G,\mathcal{I}) = \sup_{\boldsymbol{\Phi}} \sum_{k=1}^{n^{\mathcal{I}}} \mathbb{E}_{p^{(k)}(\boldsymbol{X})}\left[\log f^{(k)}(\boldsymbol{X}|G^{\mathcal{I}};\boldsymbol{\Phi})\right] - \lambda|G| - \lambda_R|\mathcal{I}|$$

If $G^* \in \boldsymbol{\mathcal{G}}$, then, under the same assumptions as those in Theorem C.12, for small enough $\beta > 0$ and small enough $\lambda > 0, \lambda_R > 0$, for any $\hat{G}^{\hat{\boldsymbol{\mathcal{I}}}} = \arg\max_{G^{\mathcal{I}}} q^*(G^{\mathcal{I}})$, $\hat{G} \simeq_{\mathcal{I}^*} G^*$ and $\hat{\mathcal{I}} = \mathcal{I}^*$.

Proof. The proof uses a similar technique as in proof of **Thm. C.14**.

We prove this theorem by contradiction. Suppose $\exists \hat{G}^{\hat{I}} = \arg\max_G q^*(G^{\mathcal{I}})$ such that $\hat{G} \not\simeq_{\mathcal{I}^*} G^*$ or $\hat{\mathcal{I}} \neq \mathcal{I}^*$. Consider another PMF $q'(G^{\mathcal{I}})$ which has the same support and same mass as $q^*(G^{\mathcal{I}})$ except for $q'((G^*)^{\mathcal{I}^*}) - q^*((G^*)^{\mathcal{I}^*}) = q^*(\hat{G}^{\hat{\boldsymbol{\mathcal{I}}}}) - q'(\hat{G}^{\hat{\boldsymbol{\mathcal{I}}}}) = \epsilon > 0$. Because $q^*(\hat{G}^{\hat{\mathcal{I}}}) > 0$, such ϵ exists. By the definition of q^*, $\boldsymbol{\mathcal{L}}(q^*) \geq \boldsymbol{\mathcal{L}}(q')$. Then, we have

$$\begin{aligned}
&\boldsymbol{\mathcal{L}}(q') - \boldsymbol{\mathcal{L}}(q^*) \\
=&\left[\mathbb{E}_{q'(G^{\mathcal{I}})}\left[S(G,\mathcal{I})\right] - \beta KL(q'(G^{\mathcal{I}})||p(G^{\mathcal{I}}))\right] - \left[\mathbb{E}_{q^*(G^{\mathcal{I}})}\left[S(G,\mathcal{I})\right] - \beta KL(q^*(G^{\mathcal{I}})||p(G^{\mathcal{I}}))\right] \\
=&\sum_{G\in\boldsymbol{\mathcal{G}}^{\mathcal{I}}} (q'(G) - q^*(G))S(G,\mathcal{I}) + \beta\left[KL(q^*(G,\mathcal{I})||p(G,\mathcal{I})) - KL(q'(G,\mathcal{I})||p(G,\mathcal{I}))\right] \\
=&\epsilon\left(S(G^*,\mathcal{I}^*) - S(\hat{G},\hat{\mathcal{I}})\right) + \beta\left[KL(q^*(G,\mathcal{I})||p(G,\mathcal{I})) - KL(q'(G,\mathcal{I})||p(G,\mathcal{I}))\right] \qquad \text{(C.25)}
\end{aligned}$$

By Theorem C.15, $\exists\lambda > 0, \lambda_R > 0$ such that $S(G^*,\mathcal{I}^*) > S(\hat{G},\hat{\mathcal{I}}), \forall G \not\simeq_{\mathcal{I}^*} G^*$ or $\mathcal{I} \neq \mathcal{I}^*$. Therefore, $S(G^*,\mathcal{I}^*) - S(\hat{G},\hat{\mathcal{I}}) = \Delta > 0$. If $KL(q'(G,\mathcal{I})||p(G,\mathcal{I})) - KL(q^*(G,\mathcal{I})||p(G,\mathcal{I})) \geq 0$, we already have $\boldsymbol{\mathcal{L}}(q') > \boldsymbol{\mathcal{L}}(q^*)$. Otherwise, we can pick

$$0 < \beta < \frac{\epsilon\Delta}{KL(q'(G,\mathcal{I})||p(G,\mathcal{I})) - KL(q^*(G,\mathcal{I})||p(G,\mathcal{I}))}$$

and $\boldsymbol{\mathcal{L}}(q') > \boldsymbol{\mathcal{L}}(q^*)$. Both cases contradict the fact that $\boldsymbol{\mathcal{L}}(q^*) \geq \boldsymbol{\mathcal{L}}(q')$. Therefore, we conclude that both $\hat{G} \simeq_{\mathcal{I}^*} G^*$ and $\hat{\mathcal{I}} = \mathcal{I}^*$ □

C.2.4 Identifiability of the True f-DAG

So far, our theory has covered the major interests of causal discovery. However, if we assume that there is a ground truth for the f-DAG, it is not obvious that identifying the causal DAG is equivalent to identifying the corresponding f-DAG. Identifying causal connections between nodes and latent factors may have important implications in applications such as computational biology, where genes often function together through specific pathways. Therefore, in this section, we introduce additional theoretical results regarding f-DAG identifiability.

The first question we would like to ask is: Is $\mathcal{I}$-Markov equivalence between f-DAGs equivalent to $\mathcal{I}$-Markov equivalence between their half-square graphs? To answer this question, we first need to introduce some notations and new concepts.

1. We denote $par(\cdot; D)$ and $chd(\cdot; D)$ as the set of parents and children of a vertex in an f-DAG, D.

2. We denote $\simeq$ as the Markov equivalence and $\simeq_{\mathcal{I}}$ as the $\mathcal{I}$-Markov equivalence relation.

3. For any factor f in an f-DAG G, we define the set $P_f(G) = \{i : chd(i) = \{f\}\}$ as the set of parents with f as the unique child in G.

4. For any factor f in an f-DAG G, we define the set $C_f(G) = \{i : par(i) = \{f\}\}$ as the set of children with only f as the unique parent in G.

In fact, not every f-DAG can be identified up to an $\mathcal{I}$-Markov equivalence class. We consider a subset of f-DAGs defined as follows.

Definition C.17. *Let $D = (V, F, E)$ be an f-DAG where $|F| = m$. Let $\mathcal{I}$ be a set of interventions. $\boldsymbol{\mathcal{D}}_m$ is defined as the set of f-DAGs with m factors and the following properties:*

1. *$\forall f \in F$, $P_f(D) \neq \emptyset$ and $C_f(D) \neq \emptyset$.*

2. *$\forall f_1, f_2 \in F, f_1 \neq f_2, f_1 \to f_2 \in D^2[F]$, $|P_{f_1}(D)| > 1$ (inclusively) or $|P_{f_2}(D)| > 1$.*

3. *$\forall f \in F$, if $|par(f; D)| > 1$, there is at most one factor $g \in par(f; D^2[F])$ such that $|par(g; D)| = 1$.*

Intuitively, the three additional conditions for f-DAGs mean

1. Any factor should have a unique parent and unique child that distinguish it from other factors.

2. There cannot be adjacent "chain" or "tree" structures in the f-DAG.

3. There should be enough v-structures in the f-DAG.

Definition C.18. *Let $G = (V, E)$ be a DAG which can be represented as a half-square graph of an f-DAG with m factors. $\boldsymbol{\mathcal{G}}_m$ is defined as the set of all DAGs having an identifiable f-DAG representation:*

$$\boldsymbol{\mathcal{G}}_m := \{G : \exists D \in \boldsymbol{\mathcal{D}}_m, G = D^2[V]\}$$

Now we present the following lemma.

Lemma C.19. *Let $D_1 = (V, F, E_1)$ and $D_2 = (V, F, E_2)$ be two f-DAGs on the same set of nodes, V, and factors, F, and $\mathcal{I}$ be a set of interventions. Let $\Xi = \{\xi_k : k \in [n^{\mathcal{I}}]\}$ be the intervention nodes. In addition, suppose $D_1, D_2 \in \boldsymbol{\mathcal{D}}_m$ defined as in Def. C.17. Then, under a permutation of factors, we have $D_1 \simeq_{\mathcal{I}} D_2 \iff D_1^2[V] \simeq_{\mathcal{I}} D_2^2[V]$.*

Proof. By Theorem C.9, we convert the proof of $\mathcal{I}$-Markov equivalence to proof of equal graph structure.

Let $D_1^{\mathcal{I}}$ and $D_2^{\mathcal{I}}$ be the corresponding extended f-DAGs.

First, we prove the forward direction by contradiction. Suppose $D_1 \simeq_{\mathcal{I}} D_2$ but $(D_1^{\mathcal{I}})^2[V] \not\simeq_{\mathcal{I}} (D_2^{\mathcal{I}})^2[V]$. Then, either there is an edge (i, j) in $(D_1^{\mathcal{I}})^2[V]$, (i, j) not in $(D_2^{\mathcal{I}})^2[V]$ or a v-structure $i \to k \leftarrow j$ mismatch between $(D_1^{\mathcal{I}})^2[V]$ and $(D_2^{\mathcal{I}})^2[V]$. In the former case, $\exists f \in F$ such that $i \to f \to j$ in D_1, but no such factor in D_2. This leads to a mismatch in skeleton between D_1 and D_2 and results in contrdiction.

In the latter case, without loss of generality, we assume $i \to k \leftarrow j$ in $(D_1^{\mathcal{I}})^2[V]$ but $i \to k \leftarrow j$ not in $(D_2^{\mathcal{I}})^2[V]$. In addition, we can also assume the skeletons of $(D_1^{\mathcal{I}})^2[V]$ and $(D_2^{\mathcal{I}})^2[V]$ match. Otherwise, we will be in the former case again. Let f_1 and f_2 be the factors such that $i \to f_1 \to k$ and $j \to f_2 \to k$ in $D_1^{\mathcal{I}}$.

If $f_1 = f_2$, we have a v-structure $i \to f_1 \leftarrow j$ in $D_1^{\mathcal{I}}$. Because $D_1 \simeq_{\mathcal{I}} D_2$ and $i \to k \leftarrow j$ not in $(D_2^{\mathcal{I}})^2[V]$, $i \to f_1 \leftarrow j$ and $k \to f_1$ must exist in $D_2^{\mathcal{I}}$. This implies two additional v-structures $k \to f_1 \leftarrow i$ and $k \to f_1 \leftarrow j$ in $D_2^{\mathcal{I}}$ but not in $D_1^{\mathcal{I}}$, contradicting the fact $D_1 \simeq_{\mathcal{I}} D_2$.

If $f_1 \neq f_2$, we have a v-structure $f_1 \to k \leftarrow f_2$ in $D_1^{\mathcal{I}}$ and $D_2^{\mathcal{I}}$. Because $(D_1^{\mathcal{I}})^2[V]$ and $(D_2^{\mathcal{I}})^2[V]$ share the same skeleton but not the v-structure $i \to k \leftarrow j$, either $f_1 \to i$ or $f_2 \to j$ exists in $D_2^{\mathcal{I}}$. Without loss of generality, we assume $f_1 \to i$ in $D_2^{\mathcal{I}}$, now that we have $k \leftarrow f_1 \to i$ in $D_2^{\mathcal{I}}$, there must be another factor f' such that $k \to f' \to i$ in $D_2^{\mathcal{I}}$. If $f' = f_2$, there will be a cycle between k and f_2. Thus, $f' \neq f_2$. Now we have a v-structure $f_1 \to i \leftarrow f'$ in $D_2^{\mathcal{I}}$ but not in $D_1^{\mathcal{I}}$, leading to a contraction.

Therefore, we conclude that $D_1 \simeq_{\mathcal{I}} D_2 \implies D_1^2[V] \simeq_{\mathcal{I}} D_2^2[V]$.

Next, we prove the reverse direction. We first claim and prove propositions about the three types of factors. Before presenting the propositions, we repeat important notations here:

1. $par(i; D)$: the set of all parents of node i in an f-DAG D.
2. $chd(i; D)$: the set of all children of node i in an f-DAG D.
3. $P_i(D) := \{h : chd(h; D) = \{i\}\}$: the set of parents **unique** to node i in an f-DAG D.
4. $C_i(D) := \{j : par(j; D) = \{i\}\}$: the set of children **unique** to node i in an f-DAG D.
5. $A \to f \to B$: all nodes in set A are connected to all nodes in set B via a factor f. We slightly abuse the notation here for conciseness.

Proposition C.20. *Given $(D_1^{\mathcal{I}})^2[V] \simeq (D_2^{\mathcal{I}})^2[V]$ under the assumptions of lemma C.19. Then, $\forall j \in F$ such that $par(j; D_1^{\mathcal{I}}) = \{i\}, chd(j; D_1^{\mathcal{I}}) = \{k\}$, $\exists j' \in F$ such that $P_{j'}(D_2^{\mathcal{I}}) = \{i\}, C_{j'}(D_2^{\mathcal{I}}) = \{k\}$ or $P_{j'}(D_2^{\mathcal{I}}) = \{k\}, C_{j'}(D_2^{\mathcal{I}}) = \{i\}$.*

Proof. Because $(D_1^{\mathcal{I}})^2[V]$ and $(D_2^{\mathcal{I}})^2[V]$ share the same skeleton, $\exists j' \in F$ such that $i \to j' \to k$ in $(D_2^{\mathcal{I}})^2[V]$ or $i \leftarrow j' \leftarrow k$ in $(D_2^{\mathcal{I}})^2[V]$. By assumption $P_j(D_1^{\mathcal{I}}) = par(j; D_1^{\mathcal{I}}) = \{i\}$ and $C_j(D_1^{\mathcal{I}}) = chd(j; D_1^{\mathcal{I}}) = \{k\}$. Now let's consider two cases.

Case I: $i \to j' \to k$ Suppose $\exists k', k' \neq k$ and $k' \in chd(j'; D_2^{\mathcal{I}})$. To match the skeleton in $(D_1^{\mathcal{I}})^2[V]$, we must have $k' \to f \to i$ in $D_1^{\mathcal{I}}$ (**Fig. C.1a**). Now consider f and j. By property 2 in lemma C.19, $|P_f(D_1^{\mathcal{I}})| > 1$ or $|P_j(D_1^{\mathcal{I}})| > 1$. Since $P_j(D_1^{\mathcal{I}}) = \{i\}$, we must have $|P_f(D_1^{\mathcal{I}})| > 1$. This implies i is a collider in $(D_1^{\mathcal{I}})^2[V]$ and $k' \to i$ in $(D_1^{\mathcal{I}})^2[V]$ and hence, in $(D_2^{\mathcal{I}})^2[V]$. This leads to a contradiction.

Case II: $k \to j' \to i$ Suppose $\exists i', i' \neq i$ and $i' \in chd(j'; D_2^{\mathcal{I}})$. Since $C_j(D_1^{\mathcal{I}}) = \{k\}$ and $P_j(D_1^{\mathcal{I}}) = \{i\}$, we must have $k \to f \to i'$ in $D_1^{\mathcal{I}}$ (**Fig. C.1b**). Now consider j and f. By property 2 in Def. C.17, $|P_f(D_1^{\mathcal{I}})| > 1$ or $|P_j(D_1^{\mathcal{I}})| > 1$. Since $par(j; D_1^{\mathcal{I}}) = \{i\}$, we must have $|P_f(D_1^{\mathcal{I}})| > 1$. Let $k' \neq k$ such that $k' \in P_f(D_1^{\mathcal{I}})$. Now i' is a collider in $(D_1^{\mathcal{I}})^2[V]$ and hence, $(D_2^{\mathcal{I}})^2[V]$. We cannot have $k' \to j' \to i'$ because this would cause the v-structure $k' \to i \leftarrow k$ in $(D_2^{\mathcal{I}})^2[V]$ and contradicts $i \to k$ in $(D_1^{\mathcal{I}})^2[V]$. Thus, $\exists f' \neq j'$, $k' \to f' \to i'$ in $D_2^{\mathcal{I}}$. This implies $i' \notin C_{f'}(D_2^{\mathcal{I}})$. We can pick $l' \in C_{f'}(D_2^{\mathcal{I}})$. It is clear that k is not connected to l' in $(D_2^{\mathcal{I}})^2[V]$. Otherwise, we would have $k \to f' \implies k \notin P_{j'}(D_2^{\mathcal{I}}) \implies i$ is a collider in $(D_2^{\mathcal{I}})^2[V] \implies k \to i$ in $D_1^{\mathcal{I}}$, which is a contradiction. Now consider l' in $D_1^{\mathcal{I}}$. We claim that $k' \to l'$ in $(D_1^{\mathcal{I}})^2[V]$. Otherwise, $\exists g \in F$ such that $l' \to g \to k'$ in $D_1^{\mathcal{I}}$ and $par(g; D_1^{\mathcal{I}}) = \{l'\}$,

which contradicts property 3 in Def. C.17. Now we can conclude that $k' \to f \to l'$. Since $k \in P_f(D_1^{\mathcal{I}})$, this implies $k \to l'$ in $(D_1^{\mathcal{I}})^2[V]$ and leads to a contradiction.

Thus, we conclude that $\exists j'$ such that $P_{j'}(G_2^{\mathcal{I}}) = \{i\}, C_{j'}(G_2^{\mathcal{I}}) = \{k\}$ or $P_{j'}(G_2^{\mathcal{I}}) = \{k\}, C_{j'}(G_2^{\mathcal{I}}) = \{i\}$. □

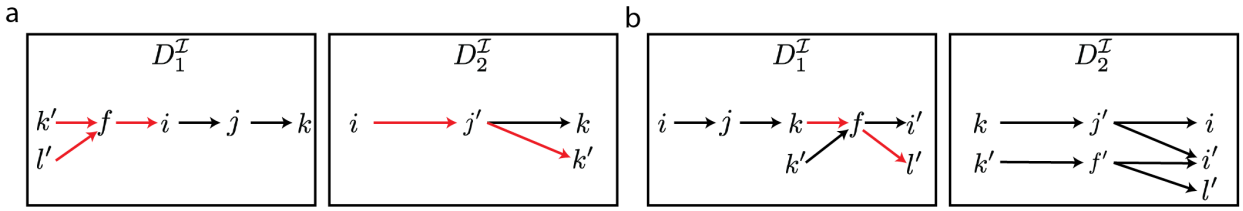

Figure C.1: **Illustration of Proposition C.20 (a)** Case I: $i \to j' \to k$ in $D_2^{\mathcal{I}}$. **(b)** Case II: $k \to j' \to i$ in $D_2^{\mathcal{I}}$. Contradictory edges are colored in red.

Proposition C.21. *Given* $(D_1^{\mathcal{I}})^2[V] \simeq (D_2^{\mathcal{I}})^2[V]$ *under the assumptions of lemma C.19. Then,* $\forall j \in F$ *such that* $par(j; D_1^{\mathcal{I}}) = P_j(D_1^{\mathcal{I}}) = \{i\}, |chd(j; D_1^{\mathcal{I}})| > 1$, $\exists j' \in F$, $par(j'; D_2^{\mathcal{I}}) = \{i\}, chd(j'; D_2^{\mathcal{I}}) = chd(j; D_1^{\mathcal{I}})$.

Proof. Because $(D_1^{\mathcal{I}})^2[V]$ and $(D_2^{\mathcal{I}})^2[V]$ share the same skeleton and $|chd(j; D_1^{\mathcal{I}})| > 1$, $\forall k \in chd(j; D_1^{\mathcal{I}})$, either (i, k) or (k, i) exists in $(D_2^{\mathcal{I}})^2[V]$. We first claim that the edges between i and $chd(j; D_1^{\mathcal{I}})$ in $(D_2^{\mathcal{I}})^2[V]$ should all be from i to $chd(j; D_1^{\mathcal{I}})$. To prove this, we consider any $k \in chd(j; D_1^{\mathcal{I}})$. If $k \notin C_j(D_1^{\mathcal{I}})$, $\exists f \in F, f \neq j, h \in V, h \neq i$ such that $h \to f \to k$ in $D_1^{\mathcal{I}}$. Hence, there is a v-structure $h \to k \leftarrow i$ in $(D_1^{\mathcal{I}})^2[V]$ and it has to exist in $(D_2^{\mathcal{I}})^2[V]$. This implies $i \to k$ in $(D_2^{\mathcal{I}})^2[V]$ (**Fig. C.2a**).

Now we prove $i \to C_j(D_1^{\mathcal{I}})$ in $(D_2^{\mathcal{I}})^2[V]$ by contradiction. If this is not the case, there are two cases:

Case I $\exists k_1, k_2 \in C_j(D_1^{\mathcal{I}}), k_1 \neq k_2, f_1, f_2 \in F$ such that $k_1 \to f_1 \to i, k_2 \to f_2 \to i$ in $D_2^{\mathcal{I}}$. There will be a v-structure $k_1 \to i \leftarrow k_2, k_1, k_2 \in A$, but such a v-structure does not exist in $(D_1^{\mathcal{I}})^2[V]$ (**Fig. C.2b**). Therefore, we have a contradiction.

Case II There is only one $h \in C_j(D_1^{\mathcal{I}}), f' \in F$ such that $h \to f' \to i$ in $D_2^{\mathcal{I}}$. Since $|chd(j; D_2^{\mathcal{I}})| > 1$, $\exists k \neq h, k \in chd(j; D_2^{\mathcal{I}})$ and $j' \in F$ such that $i \to j' \to k$ in $D_2^{\mathcal{I}}$. By property 2 of Def. C.17, $|P_{f'}(D_1^{\mathcal{I}})| > 1$ or $|P_{j'}(D_1^{\mathcal{I}})| > 1$. If $|P_{f'}(D_1^{\mathcal{I}})| > 1$, i will be a collider and $h \to i$ is in $(D_2^{\mathcal{I}})^2[V]$, but $i \to h$ in $(D_1^{\mathcal{I}})^2[V]$ and we have a mismatch in the v structure. Therefore, the only possibility is $|P_{j'}(D_1^{\mathcal{I}})| > 1$ and let $i' \neq i$ be one of them (**Fig. C.2c**). Because $par(j; D_1^{\mathcal{I}}) = \{i\}$, $i' \not\to j$ in $D_1^{\mathcal{I}}$ and consequently, $\exists f \in F$ such that $i' \to f \to k$ in

$D_1^{\mathcal{I}}$. Since $j \to k$ and $f \to k$ in $D_1^{\mathcal{I}}$, $k \notin C_f(D_1^{\mathcal{I}})$. Thus, we can pick $l \in C_f(D_1^{\mathcal{I}}), l \neq k$. We claim that $i' \to l$ in $(D_2^{\mathcal{I}})^2[V]$. Otherwise, we will have $l \to g \to i'$ and $par(g; D_2^{\mathcal{I}}) = \{l\}$, just like case II in the proof of proposition C.20. Thus, we have $i \to l$ in $(D_2^{\mathcal{I}})^2[V]$ but $i \not\to l$ in $(D_1^{\mathcal{I}})^2[V]$.

So we conclude that $\forall k \in chd(j; D_1^{\mathcal{I}}), \exists j' \in F$ such that $i \to j' \to k$ in $D_2^{\mathcal{I}}$.

Now that all edges between i and $chd(j; D_2^{\mathcal{I}})$ start from i, the remaining piece is to show such j' is unique. If $\exists J \subseteq F, |J| > 1$ such that $\forall k \in chd(j; D_1^{\mathcal{I}})$, $\exists j' \in J$, $i \to j' \to k$ in $D_2^{\mathcal{I}}$, $i \notin P_{j'}(D_2^{\mathcal{I}})$ $\forall j' \in J$. Now consider any $k \in C_j(D_1^{\mathcal{I}})$. Suppose $i \to j' \to k$ in $D_2^{\mathcal{I}}$. Because $i \notin P_{j'}(D_2^{\mathcal{I}})$, $\exists h \in P_{j'}(D_2^{\mathcal{I}})$ such that $h \to k$ in $D_2^{\mathcal{I}}$. Now, we have the v-structure $h \to k \leftarrow i$ in $(D_2^{\mathcal{I}})^2[V]$ and $(D_1^{\mathcal{I}})^2[V]$. However, $h \to k$ cannot exist in $(D_1^{\mathcal{I}})^2[V]$ because $h \notin par(j; D_1^{\mathcal{I}})$ and $k \in P_j(D_1^{\mathcal{I}})$ (**Fig. C.2d**). We have a contradiction.

Thus, we know that $\exists$ unique j' such that $i \to j' \to chd(j; D_1^{\mathcal{I}})$ in $D_2^{\mathcal{I}}$ and consequently $chd(j; D_1^{\mathcal{I}}) \subseteq chd(j'; D_2^{\mathcal{I}})$. The last step is to prove $chd(j'; D_2^{\mathcal{I}}) = chd(j; D_1^{\mathcal{I}})$, which only requires proving $chd(j'; D_2^{\mathcal{I}}) \subseteq chd(j; D_1^{\mathcal{I}})$. Suppose $\exists k' \in chd(j'; D_2^{\mathcal{I}})$ such that $j' \notin chd(j; D_1^{\mathcal{I}})$, (i, k') is in $(D_2^{\mathcal{I}})^2[V]$ but not $(D_1^{\mathcal{I}})^2[V]$. To match the skeleton, (k', i) must exist in $(D_1^{\mathcal{I}})^2[V]$. Then, $\exists f \in F$ such that $k' \to f \to i$ in $D_1^{\mathcal{I}}$ (Fig. C.2(e)).

We claim that $k' \in C_{j'}(D_2^{\mathcal{I}})$. Otherwise, k' is a collider in $(D_2^{\mathcal{I}})^2[V]$ and we must have a v-structure with $i \to k'$ in $(D_2^{\mathcal{I}})^2[V]$ but not in $(D_1^{\mathcal{I}})^2[V]$. Now we can assume $k' \in C_{j'}(D_2^{\mathcal{I}})$. By property 2 in Def. C.17, $|P_j(D_1^{\mathcal{I}})| > 1$ or $|P_f(D_1^{\mathcal{I}})| > 1$. Because $|par(jD_1^{\mathcal{I}})| = 1$ by assumption, $|P_f(D_1^{\mathcal{I}})| > 1$ and this implies i is a collider with k' as its parent in $D_1^{\mathcal{I}}$ (**Fig.C.2(e)**). However, this cannot happen in $D_2^{\mathcal{I}}$. We have a contradiction.

Finally, we conclude that for any factor in $D_1^{\mathcal{I}}$ under the conditions in proposition C.21, there exists a factor in $D_2^{\mathcal{I}}$ with the same parent and children. □

Proposition C.22. *If $(D_1^{\mathcal{I}})^2[V] \simeq_{\mathcal{I}} (D_2^{\mathcal{I}})^2[V]$ under the assumptions of lemma C.19, then $\forall j \in F$ such that $|par(j; D_1^{\mathcal{I}})| > 1$, $\exists j' \in F$ such that $par(j; D_1^{\mathcal{I}}) = par(j'; D_2^{\mathcal{I}})$ and $chd(j; D_1^{\mathcal{I}}) = chd(j'; D_2^{\mathcal{I}})$.*

Proof. The proof is trivial when $|F| = 1$. Now we assume $|F| > 1$. We first prove $\exists j', P_j(D_1^{\mathcal{I}}) = P_{j'}(D_2^{\mathcal{I}})$ and $C_j(D_1^{\mathcal{I}}) = C_{j'}(D_2^{\mathcal{I}})$.

Since $|par(j; D_1^{\mathcal{I}})| > 1$, any node in $C_j(D_1^{\mathcal{I}})$ is a collider in $(D_1^{\mathcal{I}})^2[V]$. Thus, $\forall i \in P_j(D_1^{\mathcal{I}}), k \in C_j(D_1^{\mathcal{I}}), (i, k) \in (D_2^{\mathcal{I}})^2[V]$. Thus, $P_j(D_1^{\mathcal{I}})$ is still connected to $C_j(D_1^{\mathcal{I}})$ in $(D_2^{\mathcal{I}})^2[V]$ via a set of factors $F_C = \{f_1, \ldots, f_M\}$ in $D_2^{\mathcal{I}}$. Now we consider two cases.

Case I First, we consider $M > 1$. $\forall f_m \in F_C$, we define $P_m = par(f_m; D_2^{\mathcal{I}}) \cap P_j(D_1^{\mathcal{I}})$ and $C_m = chd(f_m; D_2^{\mathcal{I}}) \cap C_j(D_1^{\mathcal{I}})$. By our assumption $P_m \neq \emptyset, C_m \neq \emptyset, \forall m$.

Pick any two factors f_1 and f_2. We consider the following two conditions

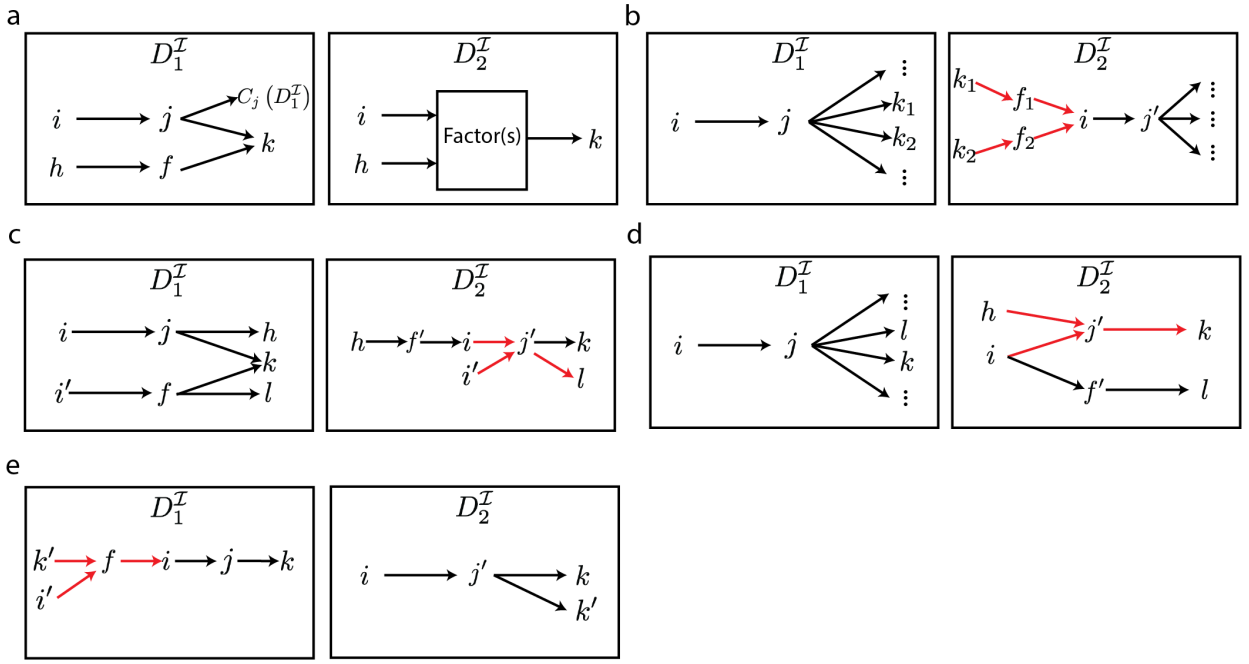

Figure C.2: **Illustration of Cases in Proposition C.21 (a)** $k \in chd(j), k \notin C_j(D_1^{\mathcal{I}})$. **(b)** Contradictory Case I: $\exists k_1, k_2 \in C_j(D_1^{\mathcal{I}})$ s.t. $k_1 \to f_1 \to i, k_2 \to f_2 \to i$ in $D_2^{\mathcal{I}}$ **(c)** Contradictory case II: $\exists$ unique h s.t. $h \to f' \to i$ in $D_2^{\mathcal{I}}$. **(d)** Contradictory Case: there exists multiple factors connecting i to $chd(j; D_1^{\mathcal{I}})$. **(e)** Contradictory Case: $chd(j; D_1^{\mathcal{I}}) \subset chd(j'; D_2^{\mathcal{I}})$. Red edges indicate contradictory v-structures.

1. $\exists i \in P_j(D_1^{\mathcal{I}})$ such that $i \to f_1$ and $i \to f_2$ in $D_2^{\mathcal{I}}$.

2. $\exists k \in C_j(D_1^{\mathcal{I}})$ such that $f_1 \to k$ and $f_2 \to k$ in $D_2^{\mathcal{I}}$.

Each condition can be either true or false. The combination of these two conditions give us four different subcases.

Subcase I-1: Condition (1) and (2) are True.

This means i is a common parent of f_1, f_2 in $D_2^{\mathcal{I}}$. Now consider $P_{f_1}(D_2^{\mathcal{I}})$ and $P_{f_2}(D_2^{\mathcal{I}})$. $P_{f_1}(D_2^{\mathcal{I}})$ and $\{i\}$ are disjoint and they form v-structures at any node in $C_{f_1}(D_2^{\mathcal{I}})$ in $D_2^{\mathcal{I}}$. Similarly, $P_{f_2}(D_2^{\mathcal{I}})$ and $\{i\}$ form v-structures at any node in $C_{f_2}(D_2^{\mathcal{I}})$. We also have $P_{f_1}(D_2^{\mathcal{I}})$ and $P_{f_2}(D_2^{\mathcal{I}})$ form v-structures at k in $D_2^{\mathcal{I}}$. By the definition of k, $P_{f_1}(D_2^{\mathcal{I}}) \cup P_{f_2}(D_2^{\mathcal{I}})$ is connected to j in $D_1^{\mathcal{I}}$. Because both $P_{f_1}(D_2^{\mathcal{I}})$ and $P_{f_2}(D_2^{\mathcal{I}})$ are connected to j in $D_1^{\mathcal{I}}$, we have $P_{f_1}(D_2^{\mathcal{I}})$, $P_{f_2}(D_2^{\mathcal{I}})$ form v-structures at $C_{f_1}(D_2^{\mathcal{I}})$ in $(D_1^{\mathcal{I}})^2[V]$, but such a v-structure cannot exist in $(D_2^{\mathcal{I}})^2[V]$ by the definition of $C_{f_1}(D_2^{\mathcal{I}})$. We reach contradiction. **Fig. C.3a** shows this scenario.

Subcase I-2: Condition (1) is False but (2) True.

We claim that $|par(f_1; D_2^{\mathcal{I}})| > 1$. Otherwise, by proposition C.21, $\exists j'$ in $D_1^{\mathcal{I}}$ such that

$par(j') = par(f_1) = P_1, chd(j') = chd(f_1)$. Because $C_1 \subset C_j(D_1^{\mathcal{I}})$, the only possibility is $j' = j$. But we know $|par(j'; D_1^{\mathcal{I}})| = |P_1| = 1$ and $|par(j; D_1^{\mathcal{I}})| > 1$. This leads to a contradiction. Hence, $|par(f_1; D_2^{\mathcal{I}})| > 1$. Similarly, $|par(f_2; D_2^{\mathcal{I}})| > 1$. Consequently, we must have v-structures at $C_{f_1}(D_2^{\mathcal{I}})$ and $C_{f_2}(D_2^{\mathcal{I}})$ in $D_2^{\mathcal{I}}$. This implies the edge connections between P_1 and $C_{f_1}(D_2^{\mathcal{I}})$ are all from P_1 to $C_{f_1}(D_2^{\mathcal{I}})$. Similarly, edge connections between P_2 and $C_{f_2}(D_2^{\mathcal{I}})$ are all from P_2 to $C_{f_2}(D_2^{\mathcal{I}})$. By the definition of P_1 and P_2, we have $j \to C_{f_1}(D_2^{\mathcal{I}})$ and $j \to C_{f_2}(D_2^{\mathcal{I}})$. This means P_1 and P_2 form v-structures at $C_{f_1}(D_2^{\mathcal{I}})$ in $(D_1^{\mathcal{I}})^2[V]$. But such v-structure(s) cannot exist in $(D_2^{\mathcal{I}})^2[V]$. Again we have a contradiction. **Fig. C.3b** shows the scenario.

Subcase I-3: Condition (1) is True but (2) False.

This implies C_1 and C_2 are disjoint sets. Now consider $P_{f_1}(D_2^{\mathcal{I}})$ and $P_{f_2}(D_2^{\mathcal{I}})$. $P_{f_1}(D_2^{\mathcal{I}})$ and i form v-structures at C_1 in $(D_2^{\mathcal{I}})^2[V]$. Hence, such a v-structure must exist in $(D_1^{\mathcal{I}})^2[V]$. By the definition of C_1, $P_{f_1}(D_2^{\mathcal{I}}) \to j$ in $D_1^{\mathcal{I}}$. Similarly, $P_{f_2}(D_2^{\mathcal{I}}) \to j$ in $D_1^{\mathcal{I}}$. Again, we have $P_{f_1}(D_2^{\mathcal{I}})$ and $P_{f_2}(D_2^{\mathcal{I}})$ for v-structures at C_1, which contradicts the fact that C_1 and C_2 are disjoint. **Fig. C.3c** shows the scenario.

Subcase I-4: Condition (1) and (2) are false.

This implies C_1 and C_2 are disjoint and P_1 and P_2 are disjoint. By proposition C.20 and C.21, $|par(f_1; D_2^{\mathcal{I}})| > 1$ and $|par(f_2; D_2^{\mathcal{I}})| > 1$ under the same argument as in subcase I-2. Hence, we have v-structures from $par(f_1; D_2^{\mathcal{I}})$ to C_1 and $par(f_2; D_2^{\mathcal{I}})$ to C_2. This means the edges must point from P_1 to C_1 and from P_2 to C_2 in $(D_1^{\mathcal{I}})^2[V]$. By the definition of P_1, P_2, C_1, C_2, they are all connected via j in $D_1^{\mathcal{I}}$. Hence, P_1 is also connected to C_2, but in $(D_2^{\mathcal{I}})^2[V]$ this is not true. We reach contradiction again. **Fig. C.3d** shows the scenario.

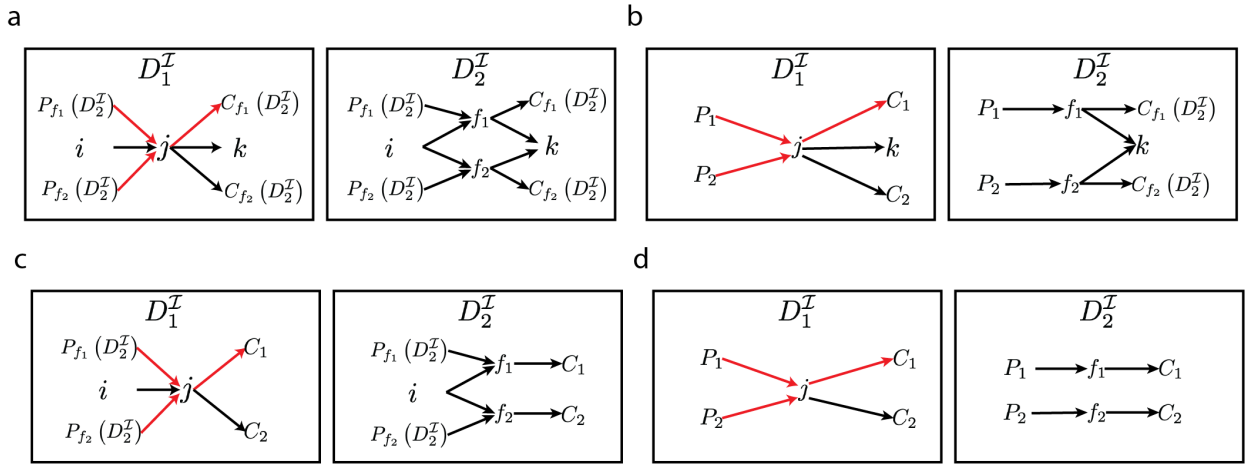

Figure C.3: **Illustration of Cases in Case I of Proposition C.22 (a)** Subcase I-1. **(b)** Subcase I-2. **(c)** Subcase I-3. **(d)** Subcase I-4. Red edges indicate contradictory v-structures.

Thus, we know that $M > 1$ is impossible.

Case II $M = 1$. There are two parts: prove equality of unique parents and equality of unique children.

Let $F_C = \{f\}$. We first prove $P_j(D_1^{\mathcal{I}}) \subseteq P_f(D_2^{\mathcal{I}})$. Suppose $\exists i \in P_j(D_1^{\mathcal{I}})$, $i \notin P_f(D_2^{\mathcal{I}})$. At the beginning of this proof, we concluded that $P_j(D_1^{\mathcal{I}})$ must connect to $C_j(D_1^{\mathcal{I}})$ in $(D_2^{\mathcal{I}})^2[V]$ due to the presence of v-structures. Since $i \in P_j(D_1^{\mathcal{I}}), i \notin P_f(D_2^{\mathcal{I}})$, the only possibility is i being connected to another factor $f' \neq f$ in $D_2^{\mathcal{I}}$. This further implies $i \notin P_{f'}(D_2^{\mathcal{I}})$.

Now consider the v-structure formed by i, any node in $P_{f'}(D_2^{\mathcal{I}})$ and any node in $C_{f'}(D_2^{\mathcal{I}})$. Due to such a v-structure, connection between i and $C_{f'}(D_2^{\mathcal{I}})$ should be from i to $C_{f'}(D_2^{\mathcal{I}})$ in $(D_1^{\mathcal{I}})^2[V]$. Because $i \in P_j(D_1^{\mathcal{I}})$, $P_j(D_1^{\mathcal{I}})$ should all be connected to $C_{f'}(D_2^{\mathcal{I}})$ via j. In short, we have $j \to C_{f'}(D_2^{\mathcal{I}})$ in $D_1^{\mathcal{I}}$.

On the other hand, i, $P_f(D_2^{\mathcal{I}})$ and $C_f(D_2^{\mathcal{I}}) \cup C_j(D_1^{\mathcal{I}})$ form v-structures in $(D_2^{\mathcal{I}})^2[V]$. Hence, $P_f(D_2^{\mathcal{I}})$ is connected to $C_j(D_1^{\mathcal{I}})$ via j in $D_1^{\mathcal{I}}$.

Based on the reasoning above, we have (1) $j \to C_{f'}(D_2^{\mathcal{I}})$ and (2) $P_f(D_2^{\mathcal{I}}) \to j$ in $D_1^{\mathcal{I}}$. This implies $P_f(D_2^{\mathcal{I}}) \to C_{f'}$ in $(D_1^{\mathcal{I}})^2[V]$. Due to the v-structure formed by i, $P_{f'}(D_2^{\mathcal{I}})$ and $C_{f'}(D_2^{\mathcal{I}})$ in $(D_2^{\mathcal{I}})^2[V]$, we also have $P_{f'}(D_2^{\mathcal{I}}) \to C_{f'}(D_2^{\mathcal{I}})$ in $(D_1^{\mathcal{I}})^2[V]$. The previous two facts further imply we have a v-structure $P_f(D_2^{\mathcal{I}}) \to C_{f'}(D_2^{\mathcal{I}}) \leftarrow P_{f'}(D_2^{\mathcal{I}})$ in $(D_1^{\mathcal{I}})^2[V]$, but such v-structure is not present in $(D_2^{\mathcal{I}})^2[V]$. Thus, we reach a contradiction. **Fig. C.4a** shows the scenario.

To conclude, so far we have proven $P_j(D_1^{\mathcal{I}}) \subseteq P_f(D_2^{\mathcal{I}})$.

Next, we prove $P_f(D_2^{\mathcal{I}}) \subseteq P_j(D_1^{\mathcal{I}})$. Suppose $\exists i \in P_f(D_2^{\mathcal{I}}), i \notin P_j(D_1^{\mathcal{I}})$. First, by our previous argument, $P_j(D_1^{\mathcal{I}}) \to j \to C_j(D_1^{\mathcal{I}})$ in $D_1^{\mathcal{I}}$ and $P_j(D_1^{\mathcal{I}}) \to f \to C_j(D_1^{\mathcal{I}})$ in $D_2^{\mathcal{I}}$. By our assumption, we also have $i \to j \to C_j(D_1^{\mathcal{I}})$in $D_1^{\mathcal{I}}$ due to the v-structures at $C_j(D_1^{\mathcal{I}})$. However, since $i \notin P_j(D_1^{\mathcal{I}})$, $\exists j' \neq j, i \to j'$ in $D_1^{\mathcal{I}}$. Apparently, $i \notin P_{j'}(D_1^{\mathcal{I}})$, so we can consider the v-structure formed by $i, P_{j'}(D_1^{\mathcal{I}}), C_{j'}(D_1^{\mathcal{I}})$. This implies $i \to C_{j'}(D_1^{\mathcal{I}})$ in $(D_2^{\mathcal{I}})^2[V]$. Because $i \in P_f(D_2^{\mathcal{I}})$, we must have $f \to C_{j'}(D_1^{\mathcal{I}})$ in $D_2^{\mathcal{I}}$. This means $P_j(D_1^{\mathcal{I}}), i, C_{j'}(D_1^{\mathcal{I}})$ form v-structures at $C_{j'}(D_1^{\mathcal{I}})$ in $(D_2^{\mathcal{I}})^2[V]$ and hence, $P_j(D_1^{\mathcal{I}}) \to C_{j'}(D_1^{\mathcal{I}})$ in $(D_1^{\mathcal{I}})^2[V]$. However, there is no connection between $P_j(D_1^{\mathcal{I}})$ and $C_{j'}(D_1^{\mathcal{I}})$. We reach a contradiction. Now we can conclude that $P_j(D_1^{\mathcal{I}}) = P_f(D_2^{\mathcal{I}})$. **Fig. C.4b** shows the scenario.

Next, we can proceed to prove $C_j(D_1^{\mathcal{I}}) = C_f(D_2^{\mathcal{I}})$. First, we prove $C_j(D_1^{\mathcal{I}}) \subseteq C_f(D_2^{\mathcal{I}})$ by contradiction. Suppose $\exists k \in C_j(D_1^{\mathcal{I}}), k \notin C_f(D_2^{\mathcal{I}})$. By the assumption of case II, $P_j(D_1^{\mathcal{I}})$ is connected to $C_j(D_1^{\mathcal{I}})$ via a unique factor, f, in $D_2^{\mathcal{I}}$. We have $P_j(D_1^{\mathcal{I}}) \to f \to k$ in $D_2^{\mathcal{I}}$. Due to the v-structures at $C_j(D_1^{\mathcal{I}})$ in $(D_1^{\mathcal{I}})^2[V]$, all nodes in $par(j; D_1^{\mathcal{I}})$ must connect to $C_j(D_1^{\mathcal{I}})$

in $D_2^{\mathcal{I}}$. Therefore, we have a stronger conclusion:

$$\forall i \in par(j; D_1^{\mathcal{I}}), i \to f \to C_j(D_1^{\mathcal{I}}) \cup C_f(D_2^{\mathcal{I}}) \text{ in } D_2^{\mathcal{I}}$$

Because $k \notin C_f(D_2^{\mathcal{I}})$, there must be another factor $f' \neq f$ such that $f' \to k$ in $D_2^{\mathcal{I}}$. Now pick $l \in P_{f'}(D_2^{\mathcal{I}})$ and $m \in C_f(D_2^{\mathcal{I}})$. Since $|par(f; D_2^{\mathcal{I}})| > 1, |par(j; D_1^{\mathcal{I}})| > 1$, we can pick $i' \in par(f; D_2^{\mathcal{I}}), i' \neq i$. We have the v-structures $i \to k \leftarrow l$, $i' \to k \leftarrow l$ and $i \to m \leftarrow i'$. in $(D_2^{\mathcal{I}})^2[V]$. Because $f \neq f'$, l and m are not connected in $(D_2^{\mathcal{I}})^2[V]$. On the other hand, because $k \in C_j(D_1^{\mathcal{I}})$ and the previous v-structures must match, we must have $l \to j \to m$ in $D_1^{\mathcal{I}}$ and hence, l and m are connected. This leads to a contradiction. **Fig. C.4c** shows the scenario.

Next, we prove $C_f(D_2^{\mathcal{I}}) \subseteq C_j(D_1^{\mathcal{I}})$ using a similar argument. Suppose $\exists k \in C_f(D_2^{\mathcal{I}}), k \notin C_j(D_1^{\mathcal{I}})$. Consider $i \in P_f(D_2^{\mathcal{I}}), i' \in par(f; D_2^{\mathcal{I}})$, we have v-structures $i \to k \leftarrow i'$ and $i \to m \leftarrow i', \forall m \in C_j(D_1^{\mathcal{I}})$ in $(D_2^{\mathcal{I}})^2[V]$. This implies $\{i, i'\} \to C_j(D_1^{\mathcal{I}}) \cup \{k\}$ in $(D_1^{\mathcal{I}})^2[V]$. By the definition of $C_f(D_2^{\mathcal{I}})$ and our previous conclusion that $P_j(D_1^{\mathcal{I}}) = P_f(D_2^{\mathcal{I}})$, we must have $i \to j \to C_j(D_1^{\mathcal{I}}) \cup \{k\}$ in $D_1^{\mathcal{I}}$. Because $k \notin C_j(D_1^{\mathcal{I}})$, $\exists j' \neq j$ such that $j' \to k$ in $D_1^{\mathcal{I}}$. Now pick $l \in P_{j'}(D_1^{\mathcal{I}})$. We have the v-structure $l \to k \leftarrow i$ in $(D_1^{\mathcal{I}})^2[V]$. Hence, such a v-structure must exist in $(D_2^{\mathcal{I}})^2[V]$. Because $k \in C_f(D_2^{\mathcal{I}})$, we must have $l \to f \to k$ in $D_2^{\mathcal{I}}$. This implies the v-structure l is connected to any $m \in C_j(D_2^{\mathcal{I}})$ in $(D_2^{\mathcal{I}})^2[V]$, but such connection does not exist in $(D_1^{\mathcal{I}})^2[V]$ since $l \in P_{j'}(D_1^{\mathcal{I}}), m \in C_j(D_1^{\mathcal{I}}), j' \neq j$. **Fig. C.4d** shows the scenario.

From the proof above, we have $\forall j$, $\exists f$, $P_j(D_1^{\mathcal{I}}) = P_f(D_2^{\mathcal{I}})$ and $C_j(D_1^{\mathcal{I}}) = C_f(D_2^{\mathcal{I}})$.

Because $|par(j; D_1^{\mathcal{I}})| > 1$ and $|chd(j; D_1^{\mathcal{I}})| > 1$, a v-structure forms $\forall i_1, i_2 \in par(j; D_1^{\mathcal{I}}), k \in chd(j; D_1^{\mathcal{I}})$. Hence, $\forall i \in par(j; D_1^{\mathcal{I}}), k \in chd(j; D_1^{\mathcal{I}})$, the direction of edge (i, k) must be from i to k. Because we have $P_j(D_1^{\mathcal{I}}) = P_f(D_2^{\mathcal{I}})$ and $C_j(D_1^{\mathcal{I}}) = C_f(D_2^{\mathcal{I}})$ and any $i \in par(j; D_1^{\mathcal{I}})$ must form v-structures at any $k \in C_j(D_1^{\mathcal{I}})$ in $(D_2^{\mathcal{I}})^2[V]$, we have $i \to k$ in both $(D_1^{\mathcal{I}})^2[V]$ and $(D_2^{\mathcal{I}})^2[V]$. The only possibility for (i, k) exist in $(D_2^{\mathcal{I}})^2[V]$ is $i \to f \to k$. This implies $par(j; D_1^{\mathcal{I}}) \subseteq par(f; D_2^{\mathcal{I}})$. Similarly, $chd(j; D_1^{\mathcal{I}}) \subseteq chd(f; D_2^{\mathcal{I}})$. $\forall i' \in par(f; D_2^{\mathcal{I}})$, $i' \in par(j; D_1^{\mathcal{I}})$ because otherwise $(i', k), \forall k \in C_j(D_1^{\mathcal{I}})$ will not be in $(D_1^{\mathcal{I}})^2[V]$. Therefore, we have $par(f; D_2^{\mathcal{I}}) \subseteq par(j; D_1^{\mathcal{I}})$ and similarly $chd(f; D_2^{\mathcal{I}}) \subseteq chd(j; D_1^{\mathcal{I}})$.

Finally, we conclude that $\forall j \in F, \exists f \in F$, $par(f; D_2^{\mathcal{I}}) = par(j; D_1^{\mathcal{I}})$ and similarly $par(f; D_2^{\mathcal{I}}) = par(j; D_1^{\mathcal{I}})$. □

Propositions C.20-C.22 provide insights about extra conditions in order for two f-DAGs to be $\mathcal{I}$-Markov equivalent. The proof of lemma C.19 will be straightforward given these propositions.

Given the conditions in lemma C.19, we have two f-DAGs with the same set of nodes and factors. Now let's consider any factor $j \in F$. For any extended f-DAG, D, and interventions,

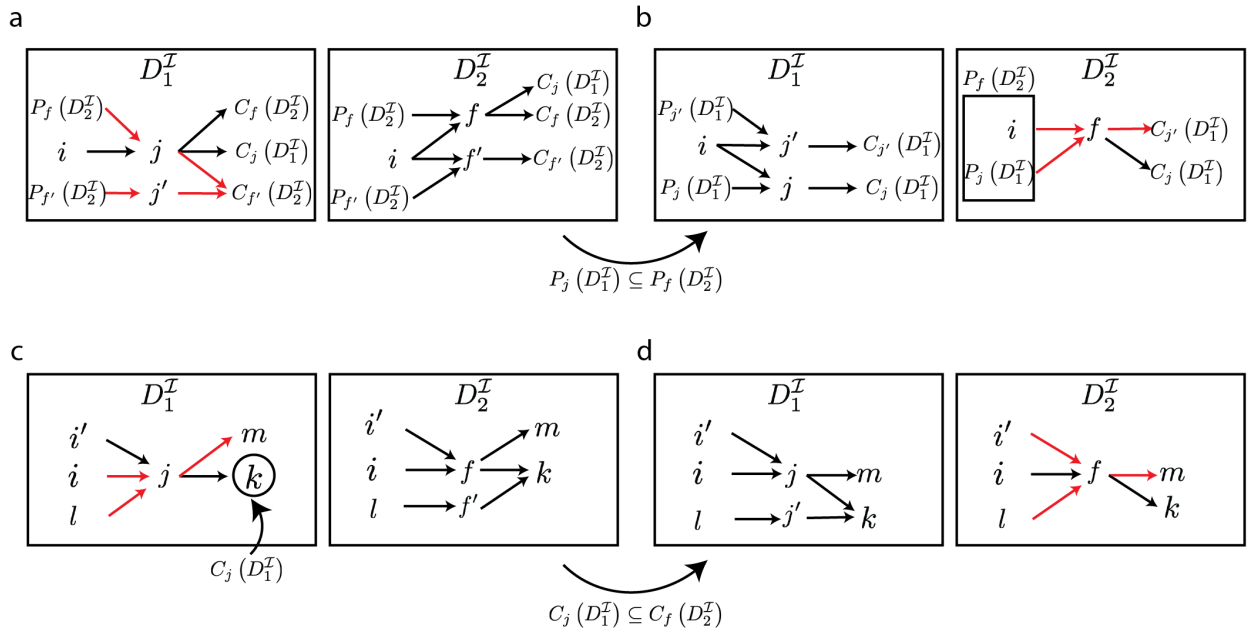

Figure C.4: **Illustration of Cases in Case II of Proposition C.22. (a)** Contradictory case when $P_j(D_1^{\mathcal{I}}) \not\subseteq P_f(D_2^{\mathcal{I}})$. **(b)** Contradictory case when $P_j(D_1^{\mathcal{I}}) \subseteq P_f(D_2^{\mathcal{I}})$ but $P_f(D_2^{\mathcal{I}}) \not\subseteq P_j(D_1^{\mathcal{I}})$. **(c)** Contradictory case when $C_j(D_1^{\mathcal{I}}) \not\subseteq C_f(D_2^{\mathcal{I}})$. **(d)** Contradictory case when $C_j(D_1^{\mathcal{I}}) \subseteq C_f(D_2^{\mathcal{I}})$ but $C_f(D_2^{\mathcal{I}}) \not\subseteq C_j(D_1^{\mathcal{I}})$. Red edges indicate contradictory v-structures.

$\mathcal{I}$, We define the following three types of factors:

1. $j : |par(j; D^{\mathcal{I}})| = |chd(j; D^{\mathcal{I}})| = 1$
2. $j : |par(j; D^{\mathcal{I}})| = 1, |chd(j; D^{\mathcal{I}})| > 1$
3. $j : |par(j; D^{\mathcal{I}})| > 1$

We can partition F into three subsets F_1, F_2, F_3 for $D_1^{\mathcal{I}}$ and F_1', F_2', F_3' for $D_2^{\mathcal{I}}$. Because $(D_1^{\mathcal{I}})^2[V] \simeq_{\mathcal{I}} (D_2^{\mathcal{I}})^2[V]$, by proposition C.20, $\forall j \in F_1$, $\exists j' \in F_1'$ with the same parents and children and $\forall j' \in F_1'$, $\exists j \in F_1$ with the same parents and children. Hence, there is a bijection between F_1 and F_1'. Similarly, we have bijections from F_2 to F_2' by proposition C.21 and F_3 to F_3' by proposition C.22. Finally, we have a bijection $\phi : F \to F$ from all factors in $D_1^{\mathcal{I}}$ to factors in $D_2^{\mathcal{I}}$. $\forall f \in F_2 \cup F_3$, f and $\phi(f)$ share the same parent(s) and children. $\forall f \in F_1$, f and $\phi(f)$, either $par(f; D_1^{\mathcal{I}}) = par(\phi(f); D_2^{\mathcal{I}}), chd(f; D_1^{\mathcal{I}}) = chd(\phi(f); D_2^{\mathcal{I}})$ or $par(f; D_1^{\mathcal{I}}) = chd(\phi(f); D_2^{\mathcal{I}}), chd(f; D_1^{\mathcal{I}}) = par(\phi(f); D_2^{\mathcal{I}})$. Since $|par(f; D_1^{\mathcal{I}})| = |chd(f; D_1^{\mathcal{I}})| = 1$, flipping the parent and child maintains the same skeleton and does not introduce v-structure.

Finally, we conclude that $D_1^{\mathcal{I}}$ and $D_2^{\mathcal{I}}$ share the same skeleton and v-structures and $D_1^2[V] \simeq_{\mathcal{I}} D_2^2[V] \implies D_1 \simeq_{\mathcal{I}} D_2$ □

Next, we discuss the identifiability of an f-DAG. The following theorem summarizes our conclusion.

Theorem C.23 (Identifiability of the f-DAG via ELBO maximization)**.** *Let $\{X_i : i \in [n]\}$ be a set of causally related random variables whose joint distribution follows a causal graphical model with a causal DAG G^* and $\mathcal{I}^* = \{I_k^* : k \in [n^{\mathcal{I}}]\}$ be a set of interventions where $I_1 = \emptyset$. Suppose $G^* \in \boldsymbol{\mathcal{G}}_m$ and there is a true f-DAG representation, $D^* \in \boldsymbol{\mathcal{D}}_m$, such that $(D^*)^2[V] = G^*$. Let $q(G; \boldsymbol{\Lambda})$ be a variational distribution on $\boldsymbol{\mathcal{G}}_m$. Define the Bayesian score function as*

$$S_{\mathcal{I}^*}(q) := \mathbb{E}_{q(G;\boldsymbol{\Lambda})}\left[S_{\mathcal{I}^*}(G)\right] - \beta KL(q(G;\boldsymbol{\Lambda})||p(G))$$

where $S_{\mathcal{I}^*}(G) = \sup_{\boldsymbol{\Phi}} \sum_{k=1}^{n^{\mathcal{I}}} \mathbb{E}_{p^{(k)}(\boldsymbol{X})}\left[\log f^{(k)}(\boldsymbol{X}|G;\boldsymbol{\Phi})\right] - \lambda|G|$ is the score function. Suppose $q^*(G;\boldsymbol{\Lambda}) = \arg\max_{q(G;\boldsymbol{\Lambda})} S_{\mathcal{I}^*}(q)$ and $\hat{G} = \arg\max_G q^*(G;\boldsymbol{\Lambda})$. Let $\hat{G}$. Then, under the same assumptions of Theorem C.12, namely sufficient capacity, $\mathcal{I}$-faithfulness, positivity and finite entropy, and for sufficiently small $\lambda > 0$ and $\beta > 0$, $\hat{D} = \arg\max_D q^*(D^2[V];\boldsymbol{\Lambda})$ is $\mathcal{I}^*$-Markov equivalent to D^* under a permutation of factors.

Proof. The proof is a direct result of previous results.

First, we have $G^* \in \boldsymbol{\mathcal{G}}_m$. Therefore, by Theorem C.14, $q^*(G) = \arg\max_{q(G;\boldsymbol{\Lambda}):supp(q)\subseteq\boldsymbol{\mathcal{G}}_m} \boldsymbol{\mathcal{L}}(q)$ has a mode $\hat{G} = (\hat{D})^2[V]$ such that $\hat{G} \simeq_{\mathcal{I}^*} G^*$. Next, since $\hat{D}, D^* \in \boldsymbol{\mathcal{D}}_m$, by lemma C.19, $\hat{D} \simeq_{\mathcal{I}^*} D^*$ under a permutation of factors. □

C.3 Supplementary Results

C.3.1 Results on Toy Datasets.

We benchmarked existing methods on simulated data using both SPN-FG and previous f-DAG simulation method from Lopez et al.[151]. In a preliminary study, we tested all methods on simple toy datasets simulated with 16 nodes and 2 factors. We changed the sparsity penalty in ENCO but it produced mainly zero adjacency matrix except for one datast with 0.13 F1 score. Hence, we report zero F1 scores here as a placeholder. **Table C.1, C.2** show the result.

C.3.2 Availability of Benchmark Results.

We conducted benchmark studies on a variety of data simulation settings at a larger scale, with 100 nodes and 10 factors. We classify the simulations by (1) SEM - linear vs. nonlinear, (2) factor graph model - SPN-FG vs. regular f-DAG and (3) type of intervention (hard vs. soft).

Table C.1: **Performance on Simulated Datasets with Linear SEM on 16 Nodes.** Best performance is in bold text and second best is underlined.

Metric	Method	Linear (FG)			Linear (SPN-FG)		
		D1	D2	D3	D1	D2	D3
SHD↓	DCDI	12	<u>4</u>	26	14	12	25
	DCDFG	48	33	31	43	43	56
	ENCO	27	24	28	28	54	29
	SDCD	<u>11</u>	16	<u>3</u>	12	15	5
	ABCDEFG	**0**	**0**	**0**	12	**0**	**0**
	ABCDEFG (SPN)	**0**	10	12	**5**	21	<u>1</u>
F1↑	DCDI	<u>0.842</u>	<u>0.923</u>	0.678	0.793	<u>0.876</u>	0.679
	DCDFG	0.529	0.190	0.644	0.566	0.650	0.509
	ENCO	0.000	0.000	0.000	0.000	0.000	0.000
	SDCD	0.825	0.750	<u>0.949</u>	<u>0.818</u>	0.842	0.918
	ABCDEFG	**1.000**	**1.000**	**1.000**	0.806	**1.000**	**1.000**
	ABCDEFG (SPN)	**1.000**	0.828	0.824	**0.912**	0.753	<u>0.983</u>

Table C.2: **Performance on Simulated Datasets with Non-linear SEM on 16 Nodes.** Best performance is in bold text and second best is underlined.

Metric	Method	Nonlinear (FG)			Nonlinear (SPN-FG)		
		D1	D2	D3	D1	D2	D3
SHD↓	DCDI	14	**7**	**4**	**2**	28	14
	DCDFG	48	18	<u>5</u>	46	36	17
	ENCO	27	18	29	37	41	29
	SDCD	<u>4</u>	**7**	6	8	<u>16</u>	**5**
	ABCDEFG	**2**	<u>12</u>	13	<u>3</u>	**12**	<u>9</u>
	ABCDEFG (SPN)	26	28	26	22	17	25
F1↑	DCDI	0.781	**0.759**	**0.935**	**0.964**	0.682	0.774
	DCDFG	0.529	N/A	<u>0.915</u>	0.477	0.667	0.691
	ENCO	0.000	0.000	0.000	0.000	0.128	0.000
	SDCD	<u>0.931</u>	**0.759**	0.889	0.833	0.795	**0.915**
	ABCDEFG	**0.964**	<u>0.500</u>	0.800	<u>0.949</u>	**0.842**	<u>0.857</u>
	ABCDEFG (SPN)	0.675	0.333	0.690	0.718	<u>0.805</u>	0.683

C.3.3 Experiment Settings

In this section, we report the hyperparameters used in our simulation study. Because ABCDEFG has many hyperparameters, we did not comprehensively tune each of them. Instead, we fixed hyperparameters across the same SEM model type. Here, we report some key hyperparameter values. For the other hyperparameters, our python program contains default values and we used the same value in all experiments. Table C.3 summarizes the most important hyperparameters. In addition, we unexhaustively tuned the L1 regularization coefficient by trying two different values per simulation type. We also have a separate L1 regularization coefficient for the intervention-to-node bipartite graph in simulation with unknown intervention targets.

Table C.3.3 lists the set of best parameters we chose for each simulation type. For conciseness, we name a simulation type by a sequence of four attributes: (1) targeted (T) vs. untargeted (U), (2) canonical f-DAG (FG) vs. SPN-FG (SPNFG), (3) linear (L) vs nonlinear (N) SEM, and (4) hard (H) vs. soft (S) intervention, separated by "-".

Table C.3: **Default Hyper-Parameter Setting of ABCDEFG in a Simulation Study.**

Parameter Name	Default Value
Batch Size	128
Hidden Dimension	1000
Number of Epochs	1000
Number of Hidden Layers	1
Width Bound of SPN (max_copies)	8
Learning Rate (VAE)	5×10^{-4}
Learning Rate (f-DAG Model)	5×10^{-3}
KL Div. Coeff. (β)	1×10^{-8}
Gaussian Noise Level	0.05
VAE Weight L2 Reg.	1×10^{-3}
Latent Factor Prior	$\mathcal{N}(\mathbf{0}, 10^{-3} \cdot \mathbf{I})$
SPN Parallelism	Factor

C.3.4 Time and Memory Consumption

All simulated datasets with known intervention targets contain 25k samples and those with unknown intervention targets contain 30k samples. With a batch size of 128, we were able to train our model on a server with 2 2x 2.9 GHz Intel Xeon Gold 6226R, 16 GB of RAM and an NVIDIA A40 GPU with 48GB of memory. The training time of ABCDEFG is shown

Table C.4: **Hyper-Parameter Setting of ABCDEFG in a Simulation Study.**

Simulation Type	L1 Reg.	L1 Reg. (Intv.)	Activation Function
T-FG-L-H	1.0	N/A	Identity
T-FG-L-S	0.01	N/A	Identity
T-FG-N-H	1.0	N/A	Tanh
T-FG-N-S	0.01	N/A	Tanh
T-SPNFG-L-H	0.01	N/A	Identity
T-SPNFG-L-S	1e-4	N/A	Identity
T-SPNFG-N-H	0.01	N/A	Tanh
T-SPNFG-N-S	1e-4	N/A	Tanh
U-FG-L-H	1.0	10.0	Identity
U-FG-L-S	0.01	0.1	Identity
U-FG-N-H	1.0	10.0	Tanh
U-FG-N-S	0.01	0.1	Tanh
U-SPNFG-L-H	0.01	0.1	Identity
U-SPNFG-L-S	1e-4	1e-3	Identity
U-SPNFG-N-H	0.01	0.1	Tanh
U-SPNFG-N-S	1e-4	1e-3	Tanh

in Fig. C.5. Since datasets are of similar sizes, the training time is stable across different simulations. Training ABCDEFG with SPN-FG consumes more time due to a larger number of parameters and extra time for forward and backward through the network layers.

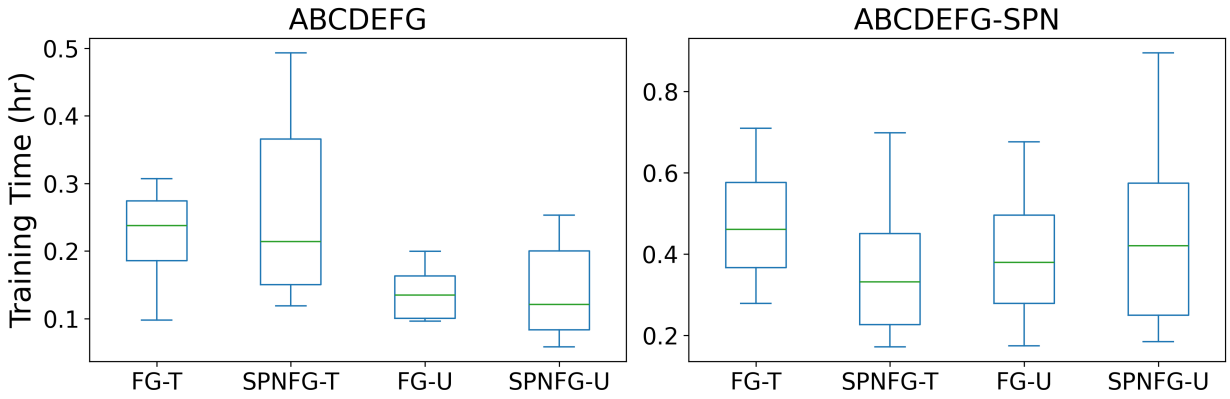

Figure C.5: **Training Time of ABCDEFG.** Each box represents one type of simulation. We groups simulation regarding the ground truth graph type and known vs. unknown intervention targets. We use the suffix "-T" for known intervention targets and "-U" for unknown ones.

C.4 DAG Evaluation

To evaluate the "DAGness" of causal graphs inferred from all methods, we define an acyclicity distance as the minimum number of edge removal from a directed graph to make it acyclic. It is formally defined as follows.

Definition C.24. *Let $G = (V, E)$ be a directed graph and denote $\setminus$ as the set difference operator. The acyclicity distance of G is defined as*

$$d(G) := \min_{\Delta E \subseteq E:\ (V, E \setminus \Delta E)\ \textit{is acyclic}} |\Delta E|$$

Calculating the acyclicity distance is equivalent to the minimum feedback arc set (FAS) problem and is proven to be NP-hard[231]. Instead of calculating the exact minimum FAS, we calculated an upper and lower bound. The first step is to obtain all strongly connected components (SCC) of a graph. We applied the algorithm by Nuutila[253, 254], implemented in the python networkx package, to find all SCCs. Next, we calculated upper and lower bounds for each SCC separately. For the upper bound, we applied a heuristic algorithm by Eades, Lin and Smyth[232], which finds a FAS, and use the set size as an upper bound. We reformulate the original algorithm in **Algorithm 1, 2** to fit our context. For any node i in SCC, either $\exists j$ such that (i, j) and (j, i) in the graph or no such j exists. In the former case, any FAS must contain at least one of (i, j) and (j, i). In the latter case, i is inside a larger loop and the larger loop needs at least one edge deletion. We include the details in **Algorithm 3**. Because the lower bound is loose, we also evaluated an asymptotic lower bound using a recently published result[233]:

$$P\left(Y^* \geq M\left(\frac{1}{2} - \sqrt{\frac{n\log(n)}{2M}}\right)\right) \geq 1 - 3\sqrt{n}e^{-n} \tag{C.26}$$

The authors assume the Erdos-Renyi graph model with a total of n nodes and M edges with 0.5 probability of being in either direction. Here, Y^* is the total number of upper-trangular edges and $Y^* = M$ means the graph is a DAG. We can see that this asymptotic lower bound holds with exponentially increasing probability as the number of nodes increases.

Algorithm 1: A Greedy Algorithm for the FAS Problem

Input : G: directed graph.

Output: d: upper bound of acyclicity distance.

```
Function greedy_fas(G):
    δ ← 0
    E ← G.edges
    rank ← HashTable()
    head ← 0
    tail ← |G.nodes| − 1
    while G.nodes ≠ ∅ do
        //Remove all sources and sinks.
        for u ∈ G.nodes do
            if in_degree(u) = 0 then
                rank[u] ← head
                head ← head + 1
                remove_node(G, u)
            end
            if out_degree(u) = 0 then
                rank[u] ← tail
                tail ← tail − 1
                remove_node(G, u)
            end
        end
        if |G.nodes| > 0 then
            u ← arg max_v |out_degree(v) − in_degree(v)|
            rank[u] ← head
            head ← head + 1
        end
    end
    //Count number of edges for deletion.
    for (u, v) ∈ E, u < v do
        if rank[u] > rank[v] then
            δ ← δ + 1
        end
    end
    return δ
```

Algorithm 2: Upper bound of acyclicity distance

Input : G: directed graph.

Output: d: upper bound of acyclicity distance.

```
1 Function dist_upper_bound(G):
      //Initialize the distance
2     d ← 0
      //Apply Tarjan's[253] or Nuutila's[254] algorithm
3     SCC ← find_all_scc(G)
      //Find a lower bound for each SCC.
4     for scc in SCC do
5         G_sub ← sub_graph(G, scc)
6         d ← d+greedy_fas(G_sub)
7     end
8     return d
```

Algorithm 3: Lower bound of acyclicity distance

Input : G: directed graph.

Output: d: acyclicity distance.

```
1  Function dist_lower_bound(G):
       //Initialize the distance
2      d ← 0
       //Apply Tarjan's[253] or Nuutila's[254] algorithm
3      SCC ← find_all_scc(G)
       //Find a lower bound for each SCC.
4      for scc in SCC do
5          if |scc| > 1 then
6              count ← 0
7              G_sub ← sub_graph(G, scc)
8              V ← ∅ //Keeps track of visited nodes.
9
10             for (u, v) in G_sub.edges, u < v do
11                 if (v, u) in G_sub.edges then
12                     count ← count + 1
13                     if u not in V then
14                         add(V, u)
15                     end
16                     if v not in V then
17                         add(V, v)
18                     end
19                 end
20             end
               //Remove at least one edge if not all nodes are in length-2
                 loops.
21             if |V| > 1 then
22                 count ← count + 1
23             end
24             d ← d + count
25         end
26     end
27     return d
```

C.5 Preprocessing single cell perturbation data

The data used for single cell perturbation is downloaded from Amin et al.[51] and we followed the preprocessing steps described by Lopez et al.[151]. For each untargeted perturbation, we removed the description words like "high","low","early",etc, and only retained the name of each biomolecule as the perturbation. We used scanpy to select the top 1000 highly variable genes as input of our model, and used 10 factors. We performed gene ontology analysis using the online tool at the Gene Ontology Website.

BIBLIOGRAPHY

[1] Nadav Yayon, Veronika R Kedlian, Lena Boehme, Chenqu Suo, Brianna T Wachter, Rebecca T Beuschel, Oren Amsalem, Krzysztof Polanski, Simon Koplev, Elizabeth Tuck, et al. A spatial human thymus cell atlas mapped to a continuous tissue axis. *Nature*, 635(8039):708–718, 2024.

[2] Ron Sender, Shai Fuchs, and Ron Milo. Revised estimates for the number of human and bacteria cells in the body. *PLoS biology*, 14(8):e1002533, 2016.

[3] Francis Crick. Central dogma of molecular biology. *Nature*, 227(5258):561–563, 1970.

[4] Alec Radford. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[5] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020.

[6] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8188–8197, 2020.

[7] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[8] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022.

[9] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.

[10] Dzmitry Bahdanau. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[11] Junbo Zhao, Yoon Kim, Kelly Zhang, Alexander Rush, and Yann LeCun. Adversarially regularized autoencoders. In *International conference on machine learning*, pages 5902–5911. PMLR, 2018.

[12] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

[13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186, 2019.

[14] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[15] Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

[16] Benjamin Sanchez-Lengeling and Alán Aspuru-Guzik. Inverse molecular design using machine learning: Generative models for matter engineering. *Science*, 361(6400):360–365, 2018.

[17] Shion Honda, Shoi Shi, and Hiroki R Ueda. Smiles transformer: Pre-trained molecular fingerprint for low data drug discovery. *arXiv preprint arXiv:1911.04738*, 2019.

[18] Moksh Jain, Emmanuel Bengio, Alex Hernandez-Garcia, Jarrid Rector-Brooks, Bonaventure FP Dossou, Chanakya Ajit Ekbote, Jie Fu, Tianyu Zhang, Michael Kilgour, Dinghuai Zhang, et al. Biological sequence design with gflownets. In *International Conference on Machine Learning*, pages 9786–9801. PMLR, 2022.

[19] Eric S. Lander, Lauren M. Linton, Bruce Birren, Chad Nusbaum, Michael C. Zody, Jennifer Baldwin, Keri Devon, Ken Dewar, Michael Doyle, William FitzHugh, Roel Funke, Diane Gage, Katrina Harris, Andrew Heaford, John Howland, Lisa Kann, Jessica Lehoczky, Rosie LeVine, Paul McEwan, Kevin McKernan, James Meldrim, Jill P. Mesirov, Cher Miranda, William Morris, Jerome Naylor, Christina Raymond, Mark Rosetti, Ralph Santos, Andrew Sheridan, Carrie Sougnez, Nicole Stange-Thomann, Nikola Stojanovic, Aravind Subramanian, Dudley Wyman, Jane Rogers, John Sulston, Rachael Ainscough, Stephan Beck, David Bentley, John Burton, Christopher Clee, Nigel Carter, Alan Coulson, Rebecca Deadman, Panos Deloukas, Andrew Dunham, Ian Dunham, Richard Durbin, Lisa French, Darren Grafham, Simon Gregory, Tim Hubbard, Sean Humphray, Adrienne Hunt, Matthew Jones, Christine Lloyd, Amanda McMurray, Lucy Matthews, Simon Mercer, Sarah Milne, James C. Mullikin, Andrew Mungall, Robert Plumb, Mark Ross, Ratna Shownkeen, Sarah Sims, Robert H. Waterston, Richard K. Wilson, LaDeana W. Hillier, John D. McPherson,

Marco A. Marra, Elaine R. Mardis, Lucinda A. Fulton, Asif T. Chinwalla, Kymberlie H. Pepin, Warren R. Gish, Stephanie L. Chissoe, Michael C. Wendl, Kim D. Delehaunty, Tracie L. Miner, Andrew Delehaunty, Jason B. Kramer, Lisa L. Cook, Robert S. Fulton, Douglas L. Johnson, Patrick J. Minx, Sandra W. Clifton, Trevor Hawkins, Elbert Branscomb, Paul Predki, Paul Richardson, Sarah Wenning, Tom Slezak, Norman Doggett, Jan-Fang Cheng, Anne Olsen, Susan Lucas, Christopher Elkin, Edward Uberbacher, Marvin Frazier, Richard A. Gibbs, Donna M. Muzny, Steven E. Scherer, John B. Bouck, Erica J. Sodergren, Kim C. Worley, Catherine M. Rives, James H. Gorrell, Michael L. Metzker, Susan L. Naylor, Raju S. Kucherlapati, David L. Nelson, George M. Weinstock, Yoshiyuki Sakaki, Asao Fujiyama, Masahira Hattori, Tetsushi Yada, Atsushi Toyoda, Takehiko Itoh, Chiharu Kawagoe, Hidemi Watanabe, Yasushi Totoki, Todd Taylor, Jean Weissenbach, Roland Heilig, William Saurin, Francois Artiguenave, Philippe Brottier, Thomas Bruls, Eric Pelletier, Catherine Robert, Patrick Wincker, André Rosenthal, Matthias Platzer, Gerald Nyakatura, Stefan Taudien, Andreas Rump, Douglas R. Smith, Lynn Doucette-Stamm, Marc Rubenfield, Keith Weinstock, Hong Mei Lee, JoAnn Dubois, Huanming Yang, Jun Yu, Jian Wang, Guyang Huang, Jun Gu, Leroy Hood, Lee Rowen, Anup Madan, Shizen Qin, Ronald W. Davis, Nancy A. Federspiel, A. Pia Abola, Michael J. Proctor, Bruce A. Roe, Feng Chen, Huaqin Pan, Juliane Ramser, Hans Lehrach, Richard Reinhardt, W. Richard McCombie, Melissa de la Bastide, Neilay Dedhia, Helmut Blöcker, Klaus Hornischer, Gabriele Nordsiek, Richa Agarwala, L. Aravind, Jeffrey A. Bailey, Alex Bateman, Serafim Batzoglou, Ewan Birney, Peer Bork, Daniel G. Brown, Christopher B. Burge, Lorenzo Cerutti, Hsiu-Chuan Chen, Deanna Church, Michele Clamp, Richard R. Copley, Tobias Doerks, Sean R. Eddy, Evan E. Eichler, Terrence S. Furey, James Galagan, James G. R. Gilbert, Cyrus Harmon, Yoshihide Hayashizaki, David Haussler, Henning Hermjakob, Karsten Hokamp, Wonhee Jang, L. Steven Johnson, Thomas A. Jones, Simon Kasif, Arek Kaspryzk, Scot Kennedy, W. James Kent, Paul Kitts, Eugene V. Koonin, Ian Korf, David Kulp, Doron Lancet, Todd M. Lowe, Aoife McLysaght, Tarjei Mikkelsen, John V. Moran, Nicola Mulder, Victor J. Pollara, Chris P. Ponting, Greg Schuler, Jörg Schultz, Guy Slater, Arian F. A. Smit, Elia Stupka, Joseph Szustakowki, Danielle Thierry-Mieg, Jean Thierry-Mieg, Lukas Wagner, John Wallis, Raymond Wheeler, Alan Williams, Yuri I. Wolf, Kenneth H. Wolfe, Shiaw-Pyng Yang, Ru-Fang Yeh, Francis Collins, Mark S. Guyer, Jane Peterson, Adam Felsenfeld, Kris A. Wetterstrand, Richard M. Myers, Jeremy Schmutz, Mark Dickson, Jane Grimwood, David R. Cox, Maynard V. Olson, Rajinder Kaul, Christopher Raymond, Nobuyoshi Shimizu, Kazuhiko Kawasaki, Shinsei Minoshima, Glen A. Evans, Maria Athanasiou, Roger Schultz, Aristides Patrinos, Michael J. Morgan, International Human Genome Sequencing Consortium, Center for Genome Research: Whitehead Institute for Biomedical Research, The Sanger Centre:, Washington University Genome Sequencing Center, US DOE Joint Genome Institute:, Baylor College of Medicine Human Genome Sequencing Center:, RIKEN Genomic Sciences Center:, Genoscope, CNRS UMR-8030:, Institute of Molecular Biotechnology: Department of Genome Analysis, GTC Sequencing Center:, Beijing Genomics Institute/Human Genome Center:, The Institute for Systems Biology: Multimegabase Sequencing Center, Stanford Genome Technology Center:, University of Oklahoma's Ad-

vanced Center for Genome Technology:, Max Planck Institute for Molecular Genetics:, Lita Annenberg Hazen Genome Center: Cold Spring Harbor Laboratory, GBF-German Research Centre for Biotechnology:, also includes individuals listed under other headings): *Genome Analysis Group (listed in alphabetical order, US National Institutes of Health: Scientific management: National Human Genome Research Institute, Stanford Human Genome Center:, University of Washington Genome Center:, Keio University School of Medicine: Department of Molecular Biology, University of Texas Southwestern Medical Center at Dallas:, US Department of Energy: Office of Science, and The Wellcome Trust:. Initial sequencing and analysis of the human genome. *Nature*, 409(6822):860–921, 2001.

[20] Mingye Hong, Shuang Tao, Ling Zhang, Li-Ting Diao, Xuanmei Huang, Shaohui Huang, Shu-Juan Xie, Zhen-Dong Xiao, and Hua Zhang. Rna sequencing: new technologies and applications in cancer research. *Journal of hematology & oncology*, 13:1–16, 2020.

[21] Xinmin Li and Cun-Yu Wang. From bulk, single-cell to spatial rna sequencing. *International journal of oral science*, 13(1):36, 2021.

[22] Dragomirka Jovic, Xue Liang, Hua Zeng, Lin Lin, Fengping Xu, and Yonglun Luo. Single-cell rna sequencing technologies and applications: A brief overview. *Clinical and translational medicine*, 12(3):e694, 2022.

[23] Fuchou Tang, Catalin Barbacioru, Yangzhou Wang, Ellen Nordman, Clarence Lee, Nanlan Xu, Xiaohui Wang, John Bodeau, Brian B Tuch, Asim Siddiqui, et al. mrna-seq whole-transcriptome analysis of a single cell. *Nature methods*, 6(5):377–382, 2009.

[24] Jillian J Goetz and Jeffrey M Trimarchi. Transcriptome sequencing of single cells with smart-seq. *Nature biotechnology*, 30(8):763–765, 2012.

[25] Diego Adhemar Jaitin, Ephraim Kenigsberg, Hadas Keren-Shaul, Naama Elefant, Franziska Paul, Irina Zaretsky, Alexander Mildner, Nadav Cohen, Steffen Jung, Amos Tanay, et al. Massively parallel single-cell rna-seq for marker-free decomposition of tissues into cell types. *Science*, 343(6172):776–779, 2014.

[26] Evan Z Macosko, Anindita Basu, Rahul Satija, James Nemesh, Karthik Shekhar, Melissa Goldman, Itay Tirosh, Allison R Bialas, Nolan Kamitaki, Emily M Martersteck, et al. Highly parallel genome-wide expression profiling of individual cells using nanoliter droplets. *Cell*, 161(5):1202–1214, 2015.

[27] Todd M Gierahn, Marc H Wadsworth, Travis K Hughes, Bryan D Bryson, Andrew Butler, Rahul Satija, Sarah Fortune, J Christopher Love, and Alex K Shalek. Seq-well: portable, low-cost rna sequencing of single cells at high throughput. *Nature methods*, 14(4):395–398, 2017.

[28] Grace XY Zheng, Jessica M Terry, Phillip Belgrader, Paul Ryvkin, Zachary W Bent, Ryan Wilson, Solongo B Ziraldo, Tobias D Wheeler, Geoff P McDermott, Junjie Zhu,

et al. Massively parallel digital transcriptional profiling of single cells. *Nature communications*, 8(1):14049, 2017.

[29] Josephine Bageritz and Gianmarco Raddi. Single-cell rna sequencing with drop-seq. *Single Cell Methods: Sequencing and Proteomics*, pages 73–85, 2019.

[30] Patrik L Ståhl, Fredrik Salmén, Sanja Vickovic, Anna Lundmark, José Fernández Navarro, Jens Magnusson, Stefania Giacomello, Michaela Asp, Jakub O Westholm, Mikael Huss, et al. Visualization and analysis of gene expression in tissue sections by spatial transcriptomics. *Science*, 353(6294):78–82, 2016.

[31] Sophia K Longo, Margaret G Guo, Andrew L Ji, and Paul A Khavari. Integrating single-cell and spatial transcriptomics to elucidate intercellular tissue dynamics. *Nature Reviews Genetics*, 22(10):627–644, 2021.

[32] Luyi Tian, Fei Chen, and Evan Z Macosko. The expanding vistas of spatial transcriptomics. *Nature Biotechnology*, 41(6):773–782, 2023.

[33] Lambda Moses and Lior Pachter. Museum of spatial transcriptomics. *Nature methods*, 19(5):534–546, 2022.

[34] Anjali Rao, Dalia Barkley, Gustavo S França, and Itai Yanai. Exploring tissue architecture using spatial transcriptomics. *Nature*, 596(7871):211–220, 2021.

[35] F Ann Ran, Patrick D Hsu, Jason Wright, Vineeta Agarwala, David A Scott, and Feng Zhang. Genome engineering using the crispr-cas9 system. *Nature protocols*, 8(11):2281–2308, 2013.

[36] Lei S Qi, Matthew H Larson, Luke A Gilbert, Jennifer A Doudna, Jonathan S Weissman, Adam P Arkin, and Wendell A Lim. Repurposing crispr as an rna-guided platform for sequence-specific control of gene expression. *Cell*, 152(5):1173–1183, 2013.

[37] Matthew H Larson, Luke A Gilbert, Xiaowo Wang, Wendell A Lim, Jonathan S Weissman, and Lei S Qi. Crispr interference (crispri) for sequence-specific control of gene expression. *Nature protocols*, 8(11):2180–2196, 2013.

[38] Luke A Gilbert, Matthew H Larson, Leonardo Morsut, Zairan Liu, Gloria A Brar, Sandra E Torres, Noam Stern-Ginossar, Onn Brandman, Evan H Whitehead, Jennifer A Doudna, et al. Crispr-mediated modular rna-guided regulation of transcription in eukaryotes. *Cell*, 154(2):442–451, 2013.

[39] Pablo Perez-Pinera, D Dewran Kocak, Christopher M Vockley, Andrew F Adler, Ami M Kabadi, Lauren R Polstein, Pratiksha I Thakore, Katherine A Glass, David G Ousterout, Kam W Leong, et al. Rna-guided gene activation by crispr-cas9–based transcription factors. *Nature methods*, 10(10):973–976, 2013.

[40] Morgan L Maeder, Samantha J Linder, Vincent M Cascio, Yanfang Fu, Quan H Ho, and J Keith Joung. Crispr rna–guided activation of endogenous human genes. *Nature methods*, 10(10):977–979, 2013.

[41] Antonia A Dominguez, Wendell A Lim, and Lei S Qi. Beyond editing: repurposing crispr–cas9 for precision genome regulation and interrogation. *Nature reviews Molecular cell biology*, 17(1):5–15, 2016.

[42] Diego Adhemar Jaitin, Assaf Weiner, Ido Yofe, David Lara-Astiaso, Hadas Keren-Shaul, Eyal David, Tomer Meir Salame, Amos Tanay, Alexander van Oudenaarden, and Ido Amit. Dissecting immune circuits by linking crispr-pooled screens with single-cell rna-seq. *Cell*, 167(7):1883–1896, 2016.

[43] Atray Dixit, Oren Parnas, Biyu Li, Jenny Chen, Charles P. Fulco, Livnat Jerby-Arnon, Nemanja D. Marjanovic, Danielle Dionne, Tyler Burks, Raktima Raychowdhury, Britt Adamson, Thomas M. Norman, Eric S. Lander, Jonathan S. Weissman, Nir Friedman, and Aviv Regev. Perturb-seq: Dissecting molecular circuits with scalable single-cell rna profiling of pooled genetic screens. *Cell*, 167(7):1853–1866.e17, 2016.

[44] Shiqi Xie, Jialei Duan, Boxun Li, Pei Zhou, and Gary C Hon. Multiplexed engineering and analysis of combinatorial enhancer activity in single cells. *Molecular cell*, 66(2):285–299, 2017.

[45] Paul Datlinger, André F Rendeiro, Christian Schmidl, Thomas Krausgruber, Peter Traxler, Johanna Klughammer, Linda C Schuster, Amelie Kuchler, Donat Alpar, and Christoph Bock. Pooled crispr screening with single-cell transcriptome readout. *Nature methods*, 14(3):297–301, 2017.

[46] Daniel Schraivogel, Andreas R Gschwind, Jennifer H Milbank, Daniel R Leonce, Petra Jakob, Lukas Mathur, Jan O Korbel, Christoph A Merten, Lars Velten, and Lars M Steinmetz. Targeted perturb-seq enables genome-scale genetic screens in single cells. *Nature methods*, 17(6):629–635, 2020.

[47] Joseph M Replogle, Thomas M Norman, Albert Xu, Jeffrey A Hussmann, Jin Chen, J Zachery Cogan, Elliott J Meer, Jessica M Terry, Daniel P Riordan, Niranjan Srinivas, Ian T. Fiddes, Joseph G. Arthur, Luigi J. Alvarado, Katherine A. Pfeiffer, Tarjei S. Mikkelsen, Jonathan S. Weissman, and Britt Adamson. Combinatorial single-cell crispr screens by direct guide rna capture and targeted sequencing. *Nature biotechnology*, 38(8):954–961, 2020.

[48] Chris J Frangieh, Johannes C Melms, Pratiksha I Thakore, Kathryn R Geiger-Schuller, Patricia Ho, Adrienne M Luoma, Brian Cleary, Livnat Jerby-Arnon, Shruti Malu, Michael S Cuoco, et al. Multimodal pooled perturb-cite-seq screens in patient models define mechanisms of cancer immune evasion. *Nature genetics*, 53(3):332–341, 2021.

[49] Jase Gehring, Jong Hwee Park, Sisi Chen, Matthew Thomson, and Lior Pachter. Highly multiplexed single-cell rna-seq by dna oligonucleotide tagging of cellular proteins. *Nature biotechnology*, 38(1):35–38, 2020.

[50] Sanjay R Srivatsan, José L McFaline-Figueroa, Vijay Ramani, Lauren Saunders, Junyue Cao, Jonathan Packer, Hannah A Pliner, Dana L Jackson, Riza M Daza, Lena

Christiansen, et al. Massively multiplex chemical transcriptomics at single-cell resolution. *Science*, 367(6473):45–51, 2020.

[51] Neal D. Amin, Kevin W. Kelley, Konstantin Kaganovsky, Massimo Onesto, Jin Hao, Yuki Miura, James P. McQueen, Noah Reis, Genta Narazaki, Tommy Li, Shravanti Kulkarni, Sergey Pavlov, and Sergiu P. Pașca. Generating human neural diversity with a multiplexed morphogen screen in organoids. *Cell Stem Cell*, 31(12):1831–1846.e9, 2024.

[52] Yunhao Wang, Yue Zhao, Audrey Bollas, Yuru Wang, and Kin Fai Au. Nanopore sequencing technology, bioinformatics and applications. *Nature biotechnology*, 39(11):1348–1365, 2021.

[53] Chris Seymour, Oxford Nanopore Technologies Ltd. Bonito: A pytorch basecaller for oxford nanopore reads, 2019. Oxford Nanopore Technologies, Ltd. Public License, v. 1.0.

[54] Yuwei Bao, Jack Wadden, John R Erb-Downward, Piyush Ranjan, Weichen Zhou, Torrin L McDonald, Ryan E Mills, Alan P Boyle, Robert P Dickson, David Blaauw, et al. Squigglenet: real-time, direct classification of nanopore signals. *Genome biology*, 22:1–16, 2021.

[55] Marc Pagès-Gallego and Jeroen de Ridder. Comprehensive benchmark and architectural analysis of deep learning models for nanopore sequencing basecalling. *Genome Biology*, 24(1):71, 2023.

[56] Delaney K Sullivan, Kyung Hoi Min, Kristján Eldjárn Hjörleifsson, Laura Luebbert, Guillaume Holley, Lambda Moses, Johan Gustafsson, Nicolas L Bray, Harold Pimentel, A Sina Booeshaghi, et al. kallisto, bustools and kb-python for quantifying bulk, single-cell and single-nucleus rna-seq. *Nature Protocols*, pages 1–21, 2024.

[57] Peter V Kharchenko, Lev Silberstein, and David T Scadden. Bayesian approach to single-cell differential expression analysis. *Nature methods*, 11(7):740–742, 2014.

[58] Greg Finak, Andrew McDavid, Masanao Yajima, Jingyuan Deng, Vivian Gersuk, Alex K Shalek, Chloe K Slichter, Hannah W Miller, M Juliana McElrath, Martin Prlic, et al. Mast: a flexible statistical framework for assessing transcriptional changes and characterizing heterogeneity in single-cell rna sequencing data. *Genome biology*, 16:1–13, 2015.

[59] Sheida Nabavi, Daniel Schmolze, Mayinuer Maitituoheti, Sadhika Malladi, and Andrew H Beck. Emdomics: a robust and powerful method for the identification of genes differentially expressed between heterogeneous classes. *Bioinformatics*, 32(4):533–541, 2016.

[60] Vasilis Ntranos, Lynn Yi, Páll Melsted, and Lior Pachter. A discriminative learning approach to differential expression analysis for single-cell rna-seq. *Nature methods*, 16(2):163–166, 2019.

[61] F Alexander Wolf, Philipp Angerer, and Fabian J Theis. Scanpy: large-scale single-cell gene expression data analysis. *Genome biology*, 19:1–5, 2018.

[62] Rahul Satija, Jeffrey A Farrell, David Gennert, Alexander F Schier, and Aviv Regev. Spatial reconstruction of single-cell gene expression data. *Nature biotechnology*, 33(5):495–502, 2015.

[63] Andrew Butler, Paul Hoffman, Peter Smibert, Efthymia Papalexi, and Rahul Satija. Integrating single-cell transcriptomic data across different conditions, technologies, and species. *Nature biotechnology*, 36(5):411–420, 2018.

[64] Tim Stuart, Andrew Butler, Paul Hoffman, Christoph Hafemeister, Efthymia Papalexi, William M Mauck, Yuhan Hao, Marlon Stoeckius, Peter Smibert, and Rahul Satija. Comprehensive integration of single-cell data. *cell*, 177(7):1888–1902, 2019.

[65] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.

[66] Vincent A Traag, Ludo Waltman, and Nees Jan Van Eck. From louvain to leiden: guaranteeing well-connected communities. *Scientific reports*, 9(1):1–12, 2019.

[67] Xun Zhu, Travers Ching, Xinghua Pan, Sherman M Weissman, and Lana Garmire. Detecting heterogeneity in single-cell rna-seq data by non-negative matrix factorization. *PeerJ*, 5:e2888, 2017.

[68] Joshua D Welch, Velina Kozareva, Ashley Ferreira, Charles Vanderburg, Carly Martin, and Evan Z Macosko. Single-cell multi-omic integration compares and contrasts features of brain cell identity. *Cell*, 177(7):1873–1887, 2019.

[69] Jialin Liu, Chao Gao, Joshua Sodicoff, Velina Kozareva, Evan Z Macosko, and Joshua D Welch. Jointly defining cell types from multiple single-cell datasets using liger. *Nature protocols*, 15(11):3632–3662, 2020.

[70] Thomas D Sherman, Tiger Gao, and Elana J Fertig. Cogaps 3: Bayesian non-negative matrix factorization for single-cell analysis with asynchronous updates and sparse data structures. *BMC bioinformatics*, 21:1–6, 2020.

[71] April R Kriebel and Joshua D Welch. Uinmf performs mosaic integration of single-cell multi-omic datasets using nonnegative matrix factorization. *Nature communications*, 13(1):780, 2022.

[72] Laleh Haghverdi, Florian Buettner, and Fabian J Theis. Diffusion maps for high-dimensional single-cell analysis of differentiation data. *Bioinformatics*, 31(18):2989–2998, 2015.

[73] Romain Lopez, Jeffrey Regier, Michael B Cole, Michael I Jordan, and Nir Yosef. Deep generative modeling for single-cell transcriptomics. *Nature methods*, 15(12):1053–1058, 2018.

[74] Christopher Heje Grønbech, Maximillian Fornitz Vording, Pascal N Timshel, Casper Kaae Sønderby, Tune H Pers, and Ole Winther. scvae: variational auto-encoders for single-cell gene expression data. *Bioinformatics*, 36(16):4415–4422, 2020.

[75] Jiarui Ding and Aviv Regev. Deep generative model embedding of single-cell rna-seq profiles on hyperspheres and hyperbolic spaces. *Nature communications*, 12(1):2554, 2021.

[76] Mohammad Lotfollahi, Mohsen Naghipourfar, Malte D Luecken, Matin Khajavi, Maren Büttner, Marco Wagenstetter, Žiga Avsec, Adam Gayoso, Nir Yosef, Marta Interlandi, et al. Mapping single-cell data to reference atlases by transfer learning. *Nature biotechnology*, 40(1):121–130, 2022.

[77] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[78] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[79] Etienne Becht, Leland McInnes, John Healy, Charles-Antoine Dutertre, Immanuel WH Kwok, Lai Guan Ng, Florent Ginhoux, and Evan W Newell. Dimensionality reduction for visualizing single-cell data using umap. *Nature biotechnology*, 37(1):38–44, 2019.

[80] Cole Trapnell, Davide Cacchiarelli, Jonna Grimsby, Prapti Pokharel, Shuqiang Li, Michael Morse, Niall J Lennon, Kenneth J Livak, Tarjei S Mikkelsen, and John L Rinn. The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells. *Nature Biotechnology*, 32:381–386, 2014.

[81] Sean C. Bendall, Kara L. Davis, El ad David Amir, Michelle D. Tadmor, Erin F. Simonds, Tiffany J. Chen, Daniel K. Shenfeld, Garry P. Nolan, and Dana Pe'er. Single-cell trajectory detection uncovers progression and regulatory coordination in human b cell development. *Cell*, 157(3):714–725, 2014.

[82] Junyue Cao, Malte Spielmann, Xiaojie Qiu, Xingfan Huang, Daniel M Ibrahim, Andrew J Hill, Fan Zhang, Stefan Mundlos, Lena Christiansen, Frank J Steemers, et al. The single-cell transcriptional landscape of mammalian organogenesis. *Nature*, 566(7745):496–502, 2019.

[83] Laleh Haghverdi, Maren Büttner, F Alexander Wolf, Florian Buettner, and Fabian J Theis. Diffusion pseudotime robustly reconstructs lineage branching. *Nature methods*, 13(10):845–848, 2016.

[84] Joshua D. Welch, Alexander J. Hartemink, and Jan F. Prins. Slicer: inferring branched, nonlinear cellular trajectories from single cell rna-seq data. *Genome Biology*, 17:106, 2016.

[85] Huidong Chen, Luca Albergante, Jonathan Y. Hsu, Caleb A. Lareau, Giosuè Lo Bosco, Jihong Guan, Shuigeng Zhou, Alexander N. Gorban, Daniel E. Bauer, Martin J. Aryee, David M. Langenau, Andrei Zinovyev, Jason D. Buenrostro, Guo-Cheng Yuan, and Luca Pinello. Single-cell trajectories reconstruction, exploration and mapping of omics data with stream. *Nature Communications*, 10:1903, 2019.

[86] Manu Setty, Vaidotas Kiseliovas, Jacob Levine, Adam Gayoso, Linas Mazutis, and Dana Pe'er. Characterization of cell fate probabilities in single-cell data with palantir. *Nature Biotechnology*, 37:451–460, 2019.

[87] Xiaojie Qiu, Qi Mao, Ying Tang, Li Wang, Raghav Chawla, Hannah A Pliner, and Cole Trapnell. Reversed graph embedding resolves complex single-cell trajectories. *Nature methods*, 14(10):979–982, 2017.

[88] Geoffrey Schiebinger, Jian Shu, Marcin Tabaka, Brian Cleary, Vidya Subramanian, Aryeh Solomon, Joshua Gould, Siyan Liu, Stacie Lin, Peter Berube, Lia Lee, Jenny Chen, Justin Brumbaugh, Philippe Rigollet, Konrad Hochedlinger, Rudolf Jaenisch, Aviv Regev, and Eric S. Lander. Optimal-transport analysis of single-cell gene expression identifies developmental trajectories in reprogramming. *Cell*, 176(4):928–943.e22, 2019.

[89] Gioele La Manno, Ruslan Soldatov, Amit Zeisel, Emelie Braun, Hannah Hochgerner, Viktor Petukhov, Katja Lidschreiber, Maria E Kastriti, Peter Lönnerberg, Alessandro Furlan, et al. Rna velocity of single cells. *Nature*, 560(7719):494–498, 2018.

[90] Ben Calderhead, Mark Girolami, and Neil Lawrence. Accelerating bayesian inference over nonlinear differential equations with gaussian processes. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 21. Curran Associates, Inc., 2009.

[91] Frank Dondelinger, Dirk Husmeier, Simon Rogers, and Maurizio Filippone. Ode parameter inference using adaptive gradient matching with gaussian processes. In Carlos M. Carvalho and Pradeep Ravikumar, editors, *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, volume 31 of *Proceedings of Machine Learning Research*, pages 216–228, Scottsdale, Arizona, USA, 29 Apr–01 May 2013. PMLR.

[92] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[93] Cagatay Yildiz, Markus Heinonen, and Harri Lahdesmaki. Ode2vae: Deep generative second order odes with bayesian neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[94] Hengguan Huang, Hongfu Liu, Hao Wang, Chang Xiao, and Ye Wang. Strode: Stochastic boundary ordinary differential equation. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 4435–4445. PMLR, 18–24 Jul 2021.

[95] Geoffrey Roeder, Paul Grant, Andrew Phillips, Neil Dalchau, and Edward Meeds. Efficient amortised Bayesian inference for hierarchical and nonlinear dynamical systems. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 4445–4455. PMLR, 09–15 Jun 2019.

[96] Nico S Gorbach, Stefan Bauer, and Joachim M Buhmann. Scalable variational inference for dynamical systems. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[97] Sanmitra Ghosh, Paul Birrell, and Daniela De Angelis. Variational inference for nonlinear ordinary differential equations. In Arindam Banerjee and Kenji Fukumizu, editors, *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 2719–2727. PMLR, 13–15 Apr 2021.

[98] Volker Bergen, Marius Lange, Stefan Peidli, F Alexander Wolf, and Fabian J Theis. Generalizing rna velocity to transient cell states through dynamical modeling. *Nature biotechnology*, 38(12):1408–1414, 2020.

[99] Mingze Gao, Chen Qiao, and Yuanhua Huang. Unitvelo: temporally unified rna velocity reinforces single-cell trajectory inference. *Nature Communications*, 13(1):6586, 2022.

[100] Haotian Cui, Hassaan Maan, Maria C. Vladoiu, Jiao Zhang, Michael D. Taylor, and Bo Wang. Deepvelo: deep learning extends rna velocity to multi-lineage systems with cell-specific kinetics. *Genome Biology*, 25:27, 2024.

[101] Shengyu Li, Pengzhi Zhang, Weiqing Chen, Lingqun Ye, Kristopher W. Brannan, Nhat-Tu Le, Jun-ichi Abe, John P. Cooke, and Guangyu Wang. A relay velocity model infers cell-dependent rna velocity. *Nature Biotechnology*, 2023.

[102] Yichen Gu, David T Blaauw, and Joshua Welch. Variational mixtures of ODEs for inferring cellular gene expression dynamics. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 7887–7901. PMLR, 17–23 Jul 2022.

[103] Yichen Gu, David Blaauw, and Joshua D. Welch. Bayesian inference of rna velocity from multi-lineage single-cell data. *bioRxiv*, 2022.

[104] Adam Gayoso, Philipp Weiler, Mohammad Lotfollahi, Dominik Klein, Justin Hong, Aaron Streets, Fabian J. Theis, and Nir Yosef. Deep generative modeling of transcriptional dynamics for rna velocity analysis in single cells. *Nature Methods*, 2023.

[105] Qian Qin, Eli Bingham, Gioele La Manno, David M. Langenau, and Luca Pinello. Pyro-velocity: probabilistic rna velocity inference from single-cell data. *bioRxiv*, 2022.

[106] Qian Li. sctour: a deep learning architecture for robust inference and accurate prediction of cellular dynamics. *Genome Biology*, 24:149, 2023.

[107] Samuel G. Rodriques, Robert R. Stickels, Aleksandrina Goeva, Carly A. Martin, Evan Murray, Charles R. Vanderburg, Joshua Welch, Linlin M. Chen, Fei Chen, and Evan Z. Macosko. Slide-seq: A scalable technology for measuring genome-wide expression at high spatial resolution. *Science*, 363(6434):1463–1467, 2019.

[108] Chun-Seok Cho, Jingyue Xi, Yichen Si, Sung-Rye Park, Jer-En Hsu, Myungjin Kim, Goo Jun, Hyun Min Kang, and Jun Hee Lee. Microscopic examination of spatial transcriptome using seq-scope. *Cell*, 184(13):3559–3572.e22, 2021.

[109] Ao Chen, Sha Liao, Mengnan Cheng, Kailong Ma, Liang Wu, Yiwei Lai, Xiaojie Qiu, Jin Yang, Jiangshan Xu, Shijie Hao, Xin Wang, Huifang Lu, Xi Chen, Xing Liu, Xin Huang, Zhao Li, Yan Hong, Yujia Jiang, Jian Peng, Shuai Liu, Mengzhe Shen, Chuanyu Liu, Quanshui Li, Yue Yuan, Xiaoyu Wei, Huiwen Zheng, Weimin Feng, Zhifeng Wang, Yang Liu, Zhaohui Wang, Yunzhi Yang, Haitao Xiang, Lei Han, Baoming Qin, Pengcheng Guo, Guangyao Lai, Pura Muñoz-Cánoves, Patrick H. Maxwell, Jean Paul Thiery, Qing-Feng Wu, Fuxiang Zhao, Bichao Chen, Mei Li, Xi Dai, Shuai Wang, Haoyan Kuang, Junhou Hui, Liqun Wang, Ji-Feng Fei, Ou Wang, Xiaofeng Wei, Haorong Lu, Bo Wang, Shiping Liu, Ying Gu, Ming Ni, Wenwei Zhang, Feng Mu, Ye Yin, Huanming Yang, Michael Lisby, Richard J. Cornall, Jan Mulder, Mathias Uhlén, Miguel A. Esteban, Yuxiang Li, Longqi Liu, Xun Xu, and Jian Wang. Spatiotemporal transcriptomic atlas of mouse organogenesis using dna nanoball-patterned arrays. *Cell*, 185(10):1777–1792.e21, 2022.

[110] Andrew J. C. Russell, Jackson A. Weir, Naeem M. Nadaf, Matthew Shabet, Vipin Kumar, Sandeep Kambhampati, Ruth Raichur, Giovanni J. Marrero, Sophia Liu, Karol S. Balderrama, Charles R. Vanderburg, Vignesh Shanmugam, Luyi Tian, J. Bryan Iorgulescu, Charles H. Yoon, Catherine J. Wu, Evan Z. Macosko, and Fei Chen. Slide-tags enables single-nucleus barcoding for multimodal spatial genomics. *Nature*, 625:101–109, 2024.

[111] Kok Hao Chen, Alistair N. Boettiger, Jeffrey R. Moffitt, Siyuan Wang, and Xiaowei Zhuang. Spatially resolved, highly multiplexed rna profiling in single cells. *Science*, 348(6233):aaa6090, 2015.

[112] Xiao Wang, William E. Allen, Matthew A. Wright, Emily L. Sylwestrak, Nikolay Samusik, Sam Vesuna, Kathryn Evans, Cindy Liu, Charu Ramakrishnan, Jia Liu,

Garry P. Nolan, Felice-Alessio Bava, and Karl Deisseroth. Three-dimensional intact-tissue sequencing of single-cell transcriptional states. *Science*, 361(6400):eaat5691, 2018.

[113] Amanda Janesick, Robert Shelansky, Andrew D Gottscho, Florian Wagner, Stephen R. Williams, Morgane Rouault, Ghezal Beliakoff, Carolyn A. Morrison, Michelli F. Oliveira, Jordan T. Sicherman, Andrew Kohlway, Jawad Abousoud, Tingsheng Yu Drennon, Seayar H. Mohabbat, 10x Development Teams, and Sarah E. B. Taylor. High resolution mapping of the breast cancer tumor microenvironment using integrated single cell, spatial and in situ analysis of ffpe tissue. *Nature Communications*, 14:8353, 2023.

[114] Valentine Svensson, Sarah A Teichmann, and Oliver Stegle. Spatialde: identification of spatially variable genes. *Nature methods*, 15(5):343–346, 2018.

[115] Shiquan Sun, Jiaqiang Zhu, and Xiang Zhou. Statistical analysis of spatial expression patterns for spatially resolved transcriptomic studies. *Nature methods*, 17(2):193–200, 2020.

[116] Jiaqiang Zhu, Shiquan Sun, and Xiang Zhou. Spark-x: non-parametric modeling enables scalable and robust detection of spatial expression patterns for large spatial transcriptomic studies. *Genome biology*, 22(1):184, 2021.

[117] Roser Vento-Tormo, Mirjana Efremova, Rachel A Botting, Margherita Y Turco, Miquel Vento-Tormo, Kerstin B Meyer, Jong-Eun Park, Emily Stephenson, Krzysztof Polański, Angela Goncalves, et al. Single-cell reconstruction of the early maternal–fetal interface in humans. *Nature*, 563(7731):347–353, 2018.

[118] Damien Arnol, Denis Schapiro, Bernd Bodenmiller, Julio Saez-Rodriguez, and Oliver Stegle. Modeling cell-cell interactions from spatial molecular data with spatial variance component analysis. *Cell reports*, 29(1):202–211, 2019.

[119] Suoqin Jin, Christian F Guerrero-Juarez, Lihua Zhang, Ivan Chang, Raul Ramos, Chen-Hsiang Kuan, Peggy Myung, Maksim V Plikus, and Qing Nie. Inference and analysis of cell-cell communication using cellchat. *Nature communications*, 12(1):1088, 2021.

[120] Xiaojie Qiu, Daniel Y Zhu, Yifan Lu, Jiajun Yao, Zehua Jing, Kyung Hoi Min, Mengnan Cheng, Hailin Pan, Lulu Zuo, Samuel King, et al. Spatiotemporal modeling of molecular holograms. *Cell*, 187(26):7351–7373, 2024.

[121] Xiaojie Qiu, Yan Zhang, Jorge D Martin-Rufino, Chen Weng, Shayan Hosseinzadeh, Dian Yang, Angela N Pogson, Marco Y Hein, Kyung Hoi Joseph Min, Li Wang, et al. Mapping transcriptomic vector fields of single cells. *Cell*, 185(4):690–711, 2022.

[122] Alexander Aivazidis, Fani Memi, Vitalii Kleshchevnikov, Brian Clarke, Oliver Stegle, and Omer Ali Bayraktar. Model-based inference of rna velocity modules improves cell fate prediction. *bioRxiv*, pages 2023–08, 2023.

[123] Peijie Zhou, Federico Bocci, Tiejun Li, and Qing Nie. Spatial transition tensor of single cells. *Nature Methods*, pages 1–10, 2024.

[124] Uthsav Chitra, Brian J Arnold, Hirak Sarkar, Kohei Sanno, Cong Ma, Sereno Lopez-Darwin, and Benjamin J Raphael. Mapping the topography of spatial gene expression with interpretable deep learning. *Nature Methods*, pages 1–12, 2025.

[125] Yichen Gu, Jialin Liu, Chen Li, and Joshua D Welch. Mapping cell fate transition in space and time. In *International Conference on Research in Computational Molecular Biology*, pages 417–420. Springer, 2024.

[126] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

[127] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

[128] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

[129] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[130] Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks? In *International Conference on Learning Representations*, 2022.

[131] Jiaxuan You, Bowen Liu, Zhitao Ying, Vijay Pande, and Jure Leskovec. Graph convolutional policy network for goal-directed molecular graph generation. *Advances in neural information processing systems*, 31, 2018.

[132] Oliver Wieder, Stefan Kohlbacher, Mélaine Kuenemann, Arthur Garon, Pierre Ducrot, Thomas Seidel, and Thierry Langer. A compact review of molecular property prediction with graph neural networks. *Drug Discovery Today: Technologies*, 37:1–12, 2020.

[133] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. Graph neural networks for social recommendation. In *The world wide web conference*, pages 417–426, 2019.

[134] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. Graph neural networks in recommender systems: a survey. *ACM Computing Surveys*, 55(5):1–37, 2022.

[135] Jiachen Li, Siheng Chen, Xiaoyong Pan, Ye Yuan, and Hong-Bin Shen. Cell clustering for spatial transcriptomics data with graph neural networks. *Nature Computational Science*, 2(6):399–408, 2022.

[136] Yahui Long, Kok Siong Ang, Mengwei Li, Kian Long Kelvin Chong, Raman Sethi, Chengwei Zhong, Hang Xu, Zhiwei Ong, Karishma Sachaphibulkij, Ao Chen, et al. Spatially informed clustering, integration, and deconvolution of spatial transcriptomics with graphst. *Nature Communications*, 14(1):1155, 2023.

[137] Peter Spirtes and Clark Glymour. An algorithm for fast recovery of sparse causal graphs. *Social Science Computer Review*, 9(1):62–72, 1991.

[138] Markus Kalisch and Peter Bühlmann. Estimating high-dimensional directed acyclic graphs with the pc-algorithm. *Journal of Machine Learning Research*, 8(22):613–636, 2007.

[139] Antti Hyttinen, Frederick Eberhardt, and Patrik O. Hoyer. Learning linear cyclic causal models with latent variables. *Journal of Machine Learning Research*, 13(109):3387–3439, 2012.

[140] Peter L Spirtes, Christopher Meek, and Thomas S Richardson. Causal inference in the presence of latent variables and selection bias. *arXiv preprint arXiv:1302.4983*, 2013.

[141] R. W. Robinson. Counting unlabeled acyclic digraphs. In Charles H. C. Little, editor, *Combinatorial Mathematics V*, pages 28–43, Berlin, Heidelberg, 1977. Springer Berlin Heidelberg.

[142] David Maxwell Chickering. Learning bayesian networks is np-complete. *Learning from data: Artificial intelligence and statistics V*, pages 121–130, 1996.

[143] Frederick Eberhardt, Clark Glymour, and Richard Scheines. On the number of experiments sufficient and in the worst case necessary to identify all causal relations among n variables. In *Conference on Uncertainty in Artificial Intelligence*, 2005.

[144] Antti Hyttinen, Frederick Eberhardt, and Patrik O. Hoyer. Experiment selection for causal discovery. *Journal of Machine Learning Research*, 14(93):3041–3071, 2013.

[145] Karren Yang, Abigail Katcoff, and Caroline Uhler. Characterizing and learning equivalence classes of causal DAGs under interventions. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 5541–5550. PMLR, 10–15 Jul 2018.

[146] Amiremad Ghassami, Alan Yang, Negar Kiyavash, and Kun Zhang. Characterizing distribution equivalence and structure learning for cyclic and acyclic directed graphs. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3494–3504. PMLR, 13–18 Jul 2020.

[147] Xun Zheng, Bryon Aragam, Pradeep K Ravikumar, and Eric P Xing. Dags with no tears: Continuous optimization for structure learning. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[148] Hao-Chih Lee, Matteo Danieletto, Riccardo Miotto, Sarah T Cherng, and Joel T Dudley. Scaling structural learning with no-bears to infer causal transcriptome networks. In *Pacific Symposium on Biocomputing 2020*, pages 391–402. World Scientific, 2019.

[149] Philippe Brouillard, Sébastien Lachapelle, Alexandre Lacoste, Simon Lacoste-Julien, and Alexandre Drouin. Differentiable causal discovery from interventional data. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21865–21877. Curran Associates, Inc., 2020.

[150] Ignavier Ng, AmirEmad Ghassami, and Kun Zhang. On the role of sparsity and dag constraints for learning linear dags. *Advances in Neural Information Processing Systems*, 33:17943–17954, 2020.

[151] Romain Lopez, Jan-Christian Hütter, Jonathan K. Pritchard, and Aviv Regev. Large-scale differentiable causal discovery of factor graphs. In *Advances in Neural Information Processing Systems*, 2022.

[152] Muralikrishnna G Sethuraman, Romain Lopez, Rahul Mohan, Faramarz Fekri, Tommaso Biancalani, and Jan-Christian Huetter. Nodags-flow: Nonlinear cyclic causal structure learning. In Francisco Ruiz, Jennifer Dy, and Jan-Willem van de Meent, editors, *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pages 6371–6387. PMLR, 25–27 Apr 2023.

[153] Achille Nazaret, Justin Hong, Elham Azizi, and David Blei. Stable differentiable causal discovery. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 37413–37445. PMLR, 21–27 Jul 2024.

[154] Phillip Lippe, Taco Cohen, and Efstratios Gavves. Efficient neural causal discovery without acyclicity constraints. In *International Conference on Learning Representations*, 2022.

[155] Raanan Y Rohekar, Yaniv Gurwicz, and Shami Nisimov. Causal interpretation of self-attention in pre-trained transformers. *Advances in Neural Information Processing Systems*, 36, 2024.

[156] Chris Cundy, Aditya Grover, and Stefano Ermon. Bcd nets: Scalable variational approaches for bayesian causal discovery. *Advances in Neural Information Processing Systems*, 34:7095–7110, 2021.

[157] Alexander Hägele, Jonas Rothfuss, Lars Lorch, Vignesh Ram Somnath, Bernhard Schölkopf, and Andreas Krause. Bacadi: Bayesian causal discovery with unknown interventions. In *International Conference on Artificial Intelligence and Statistics*, pages 1411–1436. PMLR, 2023.

[158] Yashas Annadani, Nick Pawlowski, Joel Jennings, Stefan Bauer, Cheng Zhang, and Wenbo Gong. Bayesdag: Gradient-based posterior inference for causal discovery. *Advances in Neural Information Processing Systems*, 36:1738–1763, 2023.

[159] Yue Yu, Tian Gao, Naiyu Yin, and Qiang Ji. Dags with no curl: An efficient dag structure learning approach. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12156–12166. PMLR, 18–24 Jul 2021.

[160] Mohammad Lotfollahi, F Alexander Wolf, and Fabian J Theis. scgen predicts single-cell perturbation responses. *Nature methods*, 16(8):715–721, 2019.

[161] Hengshi Yu and Joshua D Welch. Perturbnet predicts single-cell responses to unseen chemical and genetic perturbations. *bioRxiv*, 2022.

[162] Leon Hetzel, Simon Böhm, Niki Kilbertus, Stephan Günnemann, Mohammad Lotfollahi, and Fabian J Theis. Predicting cellular responses to novel drug perturbations at a single-cell resolution. In *NeurIPS 2022*, 2022.

[163] Yusuf Roohani, Kexin Huang, and Jure Leskovec. Predicting transcriptional outcomes of novel multigene perturbations with gears. *Nature Biotechnology*, pages 1–9, 2023.

[164] Zoe Piran, Niv Cohen, Yedid Hoshen, and Mor Nitzan. Disentanglement of single-cell data with biolord. *Nature Biotechnology*, pages 1–6, 2024.

[165] Mireya Plass, Jordi Solana, F. Alexander Wolf, Salah Ayoub, Aristotelis Misios, Petar Glažar, Benedikt Obermayer, Fabian J. Theis, Christine Kocks, and Nikolaus Rajewsky. Cell type atlas and lineage tree of a whole complex animal by single-cell transcriptomics. *Science*, 360(6391):eaaq1723, 2018.

[166] Aaron J Wilk, Arjun Rustagi, Nancy Q Zhao, Jonasel Roque, Giovanny J Martínez-Colón, Julia L McKechnie, Geoffrey T Ivison, Thanmayi Ranganath, Rosemary Vergara, Taylor Hollis, et al. A single-cell atlas of the peripheral immune response in patients with severe covid-19. *Nature medicine*, 26(7):1070–1076, 2020.

[167] Monika Litviňuková, Carlos Talavera-López, Henrike Maatz, Daniel Reichart, Catherine L Worth, Eric L Lindberg, Masatoshi Kanda, Krzysztof Polanski, Matthias Heinig, Michael Lee, et al. Cells of the adult human heart. *Nature*, 588(7838):466–472, 2020.

[168] Gennady Gorin, Valentine Svensson, and Lior Pachter. Protein velocity and acceleration from single-cell multiomics experiments. *Genome Biology*, 21:39, 2020.

[169] Chen Li, Maria Virgilio, Kathleen L. Collins, and Joshua D. Welch. Multi-omic single-cell velocity models epigenome–transcriptome interactions and improves cell fate prediction. *Nature Biotechnology*, 41:387–398, 03 2023.

[170] Volker Bergen, Ruslan A Soldatov, Peter V Kharchenko, and Fabian J Theis. Rna velocity—current challenges and future perspectives. *Molecular Systems Biology*, 17(8):e10282, 2021.

[171] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[172] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, Lille, France, 07–09 Jul 2015. PMLR.

[173] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

[174] Diederik P. Kingma and Jimmy Ba. Adam: a method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[175] Kaspar Märtens and Christopher Yau. Basisvae: Translation-invariant feature-level clustering with variational autoencoders. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 2928–2937. PMLR, 26–28 Aug 2020.

[176] Tobias Jahnke and Wilhelm Huisinga. Solving the chemical master equation for monomolecular reaction systems analytically. *Journal of Mathematical Biology*, 54:1–26, 2007.

[177] JifanWu Yichong Li, TiejunShi and Peijie Zhou. On the mathematics of rna velocity i: theoretical analysis. *CSIAM Transactions on Applied Mathematics*, 2(1):1–55, 2021.

[178] Jack Edmonds. Optimal branchings. *Journal of Research of National Bureau of Standards*, 71B:233–240, 1967.

[179] Aimée Bastidas-Ponce, Sophie Tritschler, Leander Dony, Katharina Scheibner, Marta Tarquis-Medina, Ciro Salinno, Silvia Schirge, Ingo Burtscher, Anika Böttcher, Fabian J. Theis, Heiko Lickert, Mostafa Bakhti, Allon Klein, and Barbara Treutlein. Comprehensive single cell mRNA profiling reveals a detailed roadmap for pancreatic endocrinogenesis. *Development*, 146(12), 06 2019.

[180] Gioele La Manno, Kimberly Siletti, Alessandro Furlan, Daniel Gyllborg, Elin Vinsland, Alejandro Mossi Albiach, Christoffer Mattsson Langseth, Irina Khven, Alex R. Lederer, Lisa M. Dratva, Anna Johnsson, Mats Nilsson, Peter Lönnerberg, and Sten Linnarsson. Molecular architecture of the developing mouse brain. *Nature*, 596(7870):92–96, 2021.

[181] Lei Jiang, Benedicte Brackeva, Zhidong Ling, Gertjan Kramer, Johannes M. Aerts, Frans Schuit, Bart Keymeulen, Daniel Pipeleers, Frans Gorus, and Geert A. Martens. Potential of protein phosphatase inhibitor 1 as biomarker of pancreatic -cell injury in vitro and in vivo. *Diabetes*, 62(8):2683–2688, 07 2013.

[182] Luis Rodrigo Cataldo, Neelanjan Vishnu, Tania Singh, Ludivine Bertonnier-Brouty, Sara Bsharat, Cheng Luan, Erik Renström, Rashmi B. Prasad, Malin Fex, Hindrik Mulder, and Isabella Artner. The mafa-target gene ppp1r1a regulates glp1r-mediated amplification of glucose-stimulated insulin secretion in -cells. *Metabolism*, 118:154734, 2021.

[183] Atsuko Kubo, Hiroaki Misonou, Makoto Matsuyama, Akane Nomori, Satoko Wada-Kakuda, Akihiko Takashima, Mitsuhiro Kawata, Shigeo Murayama, Yasuo Ihara, and Tomohiro Miyasaka. Distribution of endogenous normal tau in the mouse brain. *Journal of Comparative Neurology*, 527(5):985–998, 2019.

[184] Leigh Van Valen. Multivariate structural statistics in natural history. *Journal of Theoretical Biology*, 45(1):235–247, 1974.

[185] Chengxiang Qiu, Junyue Cao, Beth K. Martin, Tony Li, Ian C. Welsh, Sanjay Srivatsan, Xingfan Huang, Diego Calderon, William Stafford Noble, Christine M. Disteche, Stephen A. Murray, Malte Spielmann, Cecilia B. Moens, Cole Trapnell, and Jay Shendure. Systematic reconstruction of cellular trajectories across mouse embryogenesis. *Nature Genetics*, 54:328–341, 2019.

[186] Steven W Lane, David A Williams, and Fiona M Watt. Modulating the stem cell niche for tissue regeneration. *Nature biotechnology*, 32(8):795–803, 2014.

[187] Jason R Rock, Mark W Onaitis, Emma L Rawlins, Yun Lu, Cheryl P Clark, Yan Xue, Scott H Randell, and Brigid LM Hogan. Basal cells as stem cells of the mouse trachea and human airway epithelium. *Proceedings of the National Academy of Sciences*, 106(31):12771–12775, 2009.

[188] Ramsés Ayala, Tianzhi Shu, and Li-Huei Tsai. Trekking across the brain: The journey of neuronal migration. *Cell*, 128(1):29–43, 2007.

[189] Kim Thrane, Hanna Eriksson, Jonas Maaskola, Johan Hansson, and Joakim Lundeberg. Spatially resolved transcriptomics enables dissection of genetic heterogeneity in stage iii cutaneous malignant melanoma. *Cancer Research*, 78(20):5970–5979, 10 2018.

[190] Reuben Moncada, Dalia Barkley, Florian Wagner, Marta Chiodin, Joseph C. Devlin, Maayan Baron, Cristina H. Hajdu, Diane M. Simeone, and Itai Yanai. Integrating microarray-based spatial transcriptomics and single-cell rna-seq reveals tissue architecture in pancreatic ductal adenocarcinomas. *Nature Biotechnology*, 38:333–342, 03 2020.

[191] Amanda Janesick, Robert Shelansky, Andrew D Gottscho, Florian Wagner, Morgane Rouault, Ghezal Beliakoff, Michelli Faria de Oliveira, Andrew Kohlway, Jawad

Abousoud, Carolyn A Morrison, et al. High resolution mapping of the breast cancer tumor microenvironment using integrated single cell, spatial and in situ analysis of ffpe tissue. *bioRxiv*, pages 2022–10, 2022.

[192] Sungwoo Bae, Hongyoon Choi, and Dong Soo Lee. Discovery of molecular features underlying the morphological landscape by integrating spatial transcriptomic data with deep features of tissue images. *Nucleic acids research*, 49(10):e55–e55, 2021.

[193] Yuchen Liang, Guowei Shi, Runlin Cai, Yuchen Yuan, Ziying Xie, Long Yu, Yingjian Huang, Qian Shi, Lizhe Wang, Jun Li, et al. Prost: quantitative identification of spatially variable genes and domain detection in spatial transcriptomics. *Nature Communications*, 15(1):600, 2024.

[194] Dominik Klein, Giovanni Palla, Marius Lange, Michal Klein, Zoe Piran, Manuel Gander, Laetitia Meng-Papaxanthos, Michael Sterr, Lama Saber, Changying Jing, et al. Mapping cells through time and space with moscot. *Nature*, pages 1–11, 2025.

[195] Thomas N. Kipf and Max Welling. Variational graph auto-encoders, 2016.

[196] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

[197] Matan Gavish and David L. Donoho. The optimal hard threshold for singular values is $4/\sqrt{3}$. *IEEE Transactions on Information Theory*, 60(8):5040–5053, 2014.

[198] Trevor Hastie and Werner Stuetzle. Principal curves. *Journal of the American statistical association*, 84(406):502–516, 1989.

[199] Louis Faure, Ruslan Soldatov, Peter V Kharchenko, and Igor Adameyko. scFates: a scalable python package for advanced pseudotime and bifurcation analysis from single-cell data. *Bioinformatics*, 39(1):btac746, 11 2022.

[200] Dylan M Cable, Evan Murray, Luli S Zou, Aleksandrina Goeva, Evan Z Macosko, Fei Chen, and Rafael A Irizarry. Robust decomposition of cell type mixtures in spatial transcriptomics. *Nature biotechnology*, 40(4):517–526, 2022.

[201] Katherine Rhodes, Kenneth A Barr, Joshua M Popp, Benjamin J Strober, Alexis Battle, and Yoav Gilad. Human embryoid bodies as a novel system for genomic studies of functionally diverse cell types. *Elife*, 11:e71361, 2022.

[202] Taylor B. Arnold. *ggimg: Graphics Layers for Plotting Image Data with ggplot2*, 2020. R package version 0.1.0.

[203] Simona Lodato and Paola Arlotta. Generating neuronal diversity in the mammalian cerebral cortex. *Annual review of cell and developmental biology*, 31:699–720, 2015.

[204] James M. Wells and Jason R. Spence. How to make an intestine. *Development*, 141(4):752–760, 02 2014.

[205] Hannah M. Kolev and Klaus H. Kaestner. Mammalian intestinal development and differentiation—the state of the art. *Cellular and Molecular Gastroenterology and Hepatology*, 16(5):809–821, 2023.

[206] Hergen Spits. Development of $\alpha\beta$ t cells in the human thymus. *Nature Reviews Immunology*, 2(10):760–772, 2002.

[207] Gail Pearse. Normal structure, function and histology of the thymus. *Toxicologic pathology*, 34(5):504–514, 2006.

[208] Fangfang Qu, Wenjia Li, Jian Xu, Ruifang Zhang, Jincan Ke, Xiaodie Ren, Xiaogao Meng, Lexin Qin, Jingna Zhang, Fangru Lu, et al. Three-dimensional molecular architecture of mouse organogenesis. *Nature Communications*, 14(1):4599, 2023.

[209] Joanne M. Britto, Karen J. Tait, Leigh A. Johnston, Vicki E. Hammond, Michael Kalloniatis, and Seong-Seng Tan. Altered Speeds and Trajectories of Neurons Migrating in the Ventricular and Subventricular Zones of the Reeler Neocortex. *Cerebral Cortex*, 21(5):1018–1027, 09 2010.

[210] Jialin Liu, Hiroaki Manabe, Weizhou Qian, Yichen Wang, Yichen Gu, Angel Ka Yan Chu, Gaurav Gadhvi, Yuxuan Song, Noriaki Ono, and Joshua D Welch. Cytosignal detects locations and dynamics of ligand-receptor signaling at cellular resolution from spatial transcriptomic data. *bioRxiv*, pages 2024–03, 2024.

[211] Abhishek Sampath Kumar, Luyi Tian, Adriano Bolondi, Amèlia Aragonés Hernández, Robert Stickels, Helene Kretzmer, Evan Murray, Lars Wittler, Maria Walther, Gabriel Barakat, Leah Haut, Yechiel Elkabetz, Evan Z. Macosko, Léo Guignard, Fei Chen, and Alexander Meissner. Spatiotemporal transcriptomic maps of whole mouse embryos at the onset of organogenesis. *Nature Genetics*, 55:1176–1185, 2023.

[212] Andrew J. Copp, Nicholas D. E. Greene, and Jennifer N. Murdoch. The genetic basis of mammalian neurulation. *Nature Reviews Genetics*, 4:784–793, 2003.

[213] Kira Zeevaert, Mohamed H Elsafi Mabrouk, Wolfgang Wagner, and Roman Goetzke. Cell mechanics in embryoid bodies. *Cells*, 9(10):2270, 2020.

[214] Joshua M Brickman and Palle Serup. Properties of embryoid bodies. *Wiley Interdisciplinary Reviews: Developmental Biology*, 6(2):e259, 2017.

[215] Kenneth A Barr, Katherine L Rhodes, and Yoav Gilad. Embryoid bodies facilitate comparative analysis of gene expression in humans and chimpanzees across dozens of cell types. *bioRxiv*, pages 2022–07, 2022.

[216] Shinichiro Ogawa, Yoh-ichi Tagawa, Akiko Kamiyoshi, Akihiro Suzuki, Jun Nakayama, Yasuhiko Hashikura, and Shinichi Miyagawa. Crucial roles of mesodermal cell lineages in a murine embryonic stem cell–derived in vitro liver organogenesis system. *Stem cells*, 23(7):903–913, 2005.

[217] Alessandro Magli, Erin Schnettler, Scott A Swanson, Luciene Borges, Kirsta Hoffman, Ron Stewart, James A Thomson, Susan A Keirstead, and Rita CR Perlingeiro. Pax3 and tbx5 specify whether pdgfrα+ cells assume skeletal or cardiac muscle fate in differentiating embryonic stem cells. *Stem Cells*, 32(8):2072–2083, 2014.

[218] Min Su Lee, Dae-Hyun Jun, Chang-Il Hwang, Seung Soo Park, Jason Jongho Kang, Hyun-Seok Park, Jihoon Kim, Ju Han Kim, Jeong-Sun Seo, and Woong-Yang Park. Selection of neural differentiation-specific genes by comparing profiles of random differentiation. *Stem Cells*, 24(8):1946–1955, 2006.

[219] Hans Jorg Fehling, Georges Lacaud, Atsushi Kubo, Marion Kennedy, Scott Robertson, Gordon Keller, and Valerie Kouskoff. Tracking mesoderm induction and its specification to the hemangioblast during embryonic stem cell differentiation. 2003.

[220] Olga Gordeeva, Andrey Gordeev, and Pavel Erokhov. Archetypal architecture construction, patterning, and scaling invariance in a 3d embryoid body differentiation model. *Frontiers in Cell and Developmental Biology*, 10:852071, 2022.

[221] David Maxwell Chickering. Optimal structure identification with greedy search. *Journal of machine learning research*, 3(Nov):507–554, 2002.

[222] Fei Fu and Qing Zhou. Learning sparse causal gaussian networks with experimental intervention: Regularization and coordinate descent. *Journal of the American Statistical Association*, 108(501):288–300, 2013.

[223] Mauro Scanagatta, Cassio P de Campos, Giorgio Corani, and Marco Zaffalon. Learning bayesian networks with thousands of variables. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[224] Young Woong Park and Diego Klabjan. Bayesian network learning via topological order. *Journal of Machine Learning Research*, 18(99):1–32, 2017.

[225] Joseph Ramsey, Madelyn Glymour, Ruben Sanchez-Romero, and Clark Glymour. A million variables and more: the fast greedy equivalence search algorithm for learning high-dimensional graphical causal models, with an application to functional magnetic resonance images. *International journal of data science and analytics*, 3:121–129, 2017.

[226] Biwei Huang, Kun Zhang, Yizhu Lin, Bernhard Schölkopf, and Clark Glymour. Generalized score functions for causal discovery. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1551–1560, 2018.

[227] Hong-Mei Zhang, Hu Chen, Wei Liu, Hui Liu, Jing Gong, Huili Wang, and An-Yuan Guo. AnimalTFDB: a comprehensive animal transcription factor database. *Nucleic Acids Research*, 40(D1):D144–D149, 11 2011.

[228] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017.

[229] Hoifung Poon and Pedro Domingos. Sum-product networks: A new deep architecture. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 689–690. IEEE, 2011.

[230] Andy Shih and Stefano Ermon. Probabilistic circuits for variational inference in discrete graphical models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 4635–4646. Curran Associates, Inc., 2020.

[231] Richard M Karp. Reducibility among combinatorial problems. In *50 Years of Integer Programming 1958-2008: from the Early Years to the State-of-the-Art*, pages 219–241. Springer, 2009.

[232] Peter Eades, Xuemin Lin, and William F Smyth. A fast and effective heuristic for the feedback arc set problem. *Information processing letters*, 47(6):319–323, 1993.

[233] Harvey Diamond, Mark Kon, and Louise Raphael. Asymptotic lower bounds for the feedback arc set problem in random graphs. *arXiv preprint arXiv:2409.16443*, 2024.

[234] Samuel I Holt, Zhaozhi Qian, and Mihaela van der Schaar. Neural laplace: Learning diverse classes of differential equations in the laplace domain. In *International Conference on Machine Learning*, pages 8811–8832. PMLR, 2022.

[235] Qianying Cao, Somdatta Goswami, and George Em Karniadakis. Laplace neural operator for solving differential equations. *Nature Machine Intelligence*, 6(6):631–640, 2024.

[236] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. pmlr, 2015.

[237] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

[238] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[239] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[240] Christina V Theodoris, Ling Xiao, Anant Chopra, Mark D Chaffin, Zeina R Al Sayed, Matthew C Hill, Helene Mantineo, Elizabeth M Brydon, Zexian Zeng, X Shirley Liu, et al. Transfer learning enables predictions in network biology. *Nature*, 618(7965):616–624, 2023.

[241] Haiyang Bian, Yixin Chen, Xiaomin Dong, Chen Li, Minsheng Hao, Sijie Chen, Jinyi Hu, Maosong Sun, Lei Wei, and Xuegong Zhang. scmulan: a multitask generative pre-trained language model for single-cell analysis. In *International Conference on Research in Computational Molecular Biology*, pages 479–482. Springer, 2024.

[242] Haotian Cui, Chloe Wang, Hassaan Maan, Kuan Pang, Fengning Luo, Nan Duan, and Bo Wang. scgpt: toward building a foundation model for single-cell multi-omics using generative ai. *Nature Methods*, 21(8):1470–1480, 2024.

[243] Minsheng Hao, Jing Gong, Xin Zeng, Chiming Liu, Yucheng Guo, Xingyi Cheng, Taifeng Wang, Jianzhu Ma, Xuegong Zhang, and Le Song. Large-scale foundation model on single-cell transcriptomics. *Nature methods*, 21(8):1481–1491, 2024.

[244] Raanan Y. Rohekar, Yaniv Gurwicz, and Shami Nisimov. Causal interpretation of self-attention in pre-trained transformers. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 31450–31465. Curran Associates, Inc., 2023.

[245] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[246] Tiejun Li, Jifan Shi, Yichong Wu, and Peijie Zhou. On the mathematics of rna velocity i: Theoretical analysis. *bioRxiv*, 2020.

[247] Blanca Pijuan-Sala, Jonathan A. Griffiths, Carolina Guibentif, Tom W. Hiscock, Wajid Jawaid, Fernando J. Calero-Nieto, Carla Mulas, Ximena Ibarra-Soria, Richard C. V. Tyser, Debbie Lee Lian Ho, Wolf Reik, Shankar Srinivas, Benjamin D. Simons, Jennifer Nichols, John C. Marioni, and Berthold Göttgens. A single-cell molecular map of mouse gastrulation and early organogenesis. *Nature*, 566(7745):490–495, 2019.

[248] Hannah Hochgerner, Amit Zeisel, Peter Lönnerberg, and Sten Linnarsson. Conserved properties of dentate gyrus neurogenesis across postnatal development revealed by single-cell rna sequencing. *Nature Neuroscience*, 21(2):290–299, 2018.

[249] Melania Barile, Ivan Imaz-Rosshandler, Isabella Inzani, Shila Ghazanfar, Jennifer Nichols, John C. Marioni, Carolina Guibentif, and Berthold Göttgens. Coordinated changes in gene expression kinetics underlie both mouse and human erythroid maturation. *Genome Biology*, 22:197, 2021.

[250] Peter V Kharchenko. The triumphs and limitations of computational methods for scrna-seq. *Nature Methods*, 18(7):723–732, 2021.

[251] Nico Battich, Joep Beumer, Buys de Barbanson, Lenno Krenning, Chloé S. Baron, Marvin E. Tanenbaum, Hans Clevers, and Alexander van Oudenaarden. Sequencing metabolically labeled transcripts in single cells reveals mrna turnover strategies. *Science*, 367(6482):1151–1156, 2020.

[252] Massively parallel and time-resolved rna sequencing in single cells with scnt-seq. *Nature Methods*, 17:991–1001, 2020.

[253] Robert Tarjan. Depth-first search and linear graph algorithms. *SIAM journal on computing*, 1(2):146–160, 1972.

[254] Esko Nuutila and Eljas Soisalon-Soininen. On finding the strongly connected components in a directed graph. *Information processing letters*, 49(1):9–14, 1994.