Spring 2021

Unsupervised Machine Learning Algorithms to Characterize Single-Cell Heterogeneity and Perturbation Response

Daniel Bernard Burkhardt
Yale University Graduate School of Arts and Sciences, daniel.b.burkhardt@gmail.com

Abstract

Unsupervised Machine Learning Algorithms to Characterize Single-Cell Heterogeneity and Perturbation Response

Daniel Bernard Burkhardt

2021

Recent advances in microfluidic technologies facilitate the measurement of gene expression, DNA accessibility, protein content, or genomic mutations at unprecedented scale. The challenges imposed by the scale of these datasets are further exacerbated by non-linearity in molecular effects, complex interdependencies between features, and a lack of understanding of both data generating processes and sources of technical and biological noise. As a result, analysis of modern single-cell data requires the development of specialized computational tools. One solution to these problems is the use of manifold learning, a sub-field of unsupervised machine learning that seeks to model data geometry using a simplifying assumption that the underlying system is continuous and locally Euclidean. In this dissertation, I show how manifold learning is naturally suited for single-cell analysis and introduce three related algorithms for characterization of single-cell heterogeneity and perturbation response. I first describe Vertex Frequency Clustering, an algorithm that identifies groups of cells with similar responses to an experiment perturbation by analyzing the spectral representation of condition labels expressed as signals over a cell similarity graph. Next, I introduce MELD, an algorithm that expands on these ideas to estimate the density of each experimental sample over the graph to quantify the effect of an experimental perturbation at single cell resolution. Finally, I describe a neural network for archetypal analysis that represents the data as continuously distributed between a set of extrema. Each of these algorithms are demonstrated on a combination of real and synthetic datasets and are benchmarked against state-of-the-art algorithms.

Unsupervised Machine Learning Algorithms to Characterize Single-Cell Heterogeneity and Perturbation Response

A Dissertation
Presented to the Faculty of the Graduate School
of
Yale University
in Candidacy for the Degree of
Doctor of Philosophy

by
Daniel Bernard Burkhardt

Dissertation Director: Smita Krishnaswamy

June 2021

Acknowledgments

This dissertation is the result of six years of learning facilitated by countless mentors, teachers, and peers. I would like to thank my dissertation supervisor and advisor, Dr. Smita Krishnaswamy. Dr. Krishnaswamy provided me both the freedom to pursue my scientific interests and the support to learn and grow as a scientist. She taught me that creativity is at the heart of quantitative reasoning. Without her guidance, none of this work would be possible. Thank you. I would also like to thank Dr. Antonio Giraldez, who co-advised the first two years of my dissertation research and taught me the importance of clarity in scientific communication both written and presented. I also would like to thank my thesis committee mentors: my committee chair, Dr. James Noonan, Dr. Marc Hammarlund, Dr. Ronald Coifman, and Dr. Nenad Sestan. These mentors ensured the scientific rigor of my research and provided crucial guidance throughout my PhD. I would also like to thank my many collaborators. Inside the Krishnaswamy lab, I have benefited greatly from Scott Gigante, Alexander Tong, Matthew Amodio, Dr. Guy Wolf, Dr. David van Dijk, and Dr. Kevin Moon. Thank you also to Dr. James Noonan and his lab, Dr. Kevan Herold and his lab, Dr. Mandar Mazumdar and his lab, Dr. Shanqin Guo and her lab, Dr. Natalia Ivanova and her lab, Dr. Roy Decker, Dr. Allison Campbell, Dr. Guilin Wang and the Yale Center for Genome Analysis. Thank you also to all my informal mentors. Dr. Charles Vejnar was an invaluable teacher of proper programming and computer infrastructure design. Cassandra Kontur helped me make great slides and along with Dr. Valerie Tornini helped keep my writing clean. Thank you also to all my colleagues in the Genetics Department community, to the amazing administrators in the business office, and our Registrar Debbie Lossi-Sullivan who together keep our research running. Finally, thank you to my family and my friends who have kept me sane. You are the best.

If I have seen further it is by standing on the shoulders of Giants.

Contents

1 Introduction **1**

2 Vertex Frequency Clustering **10**

2.1 Introduction 11

2.1.1 Prior Works 12

2.1.2 Notation 13

2.2 Background 13

2.2.1 Graph Signal Processing 13

2.2.2 Vertex-Frequency Analysis 14

2.3 Vertex-Frequency Clustering 15

2.3.1 Problem Statement 15

2.3.2 Frame Selection 16

2.3.3 VFC 17

2.4 Synthetic Examples 18

2.4.1 Gaussian Mixtures 18

2.4.2 Anomaly Detection 20

2.5 Analysis of single-cell RNA sequencing 21

2.6 Discussion 23

3 Quantifying the effect of experimental perturbations at single-cell resolution **24**

3.1 Introduction . . . 25
3.2 Results . . . 27
3.2.1 Overview of the MELD algorithm . . . 27
3.2.2 Calculating sample-associated density estimates . . . 30
3.2.3 Using sample-associated relative likelihood to quantify differences between experimental conditions . . . 32
3.2.4 VFC identifies cell populations affected by a perturbation . . . 33
3.2.5 Quantitative validation of the MELD and VFC algorithms . . . 37
3.2.6 The sample-associated relative likelihood identifies a biologically relevant signature of T cell activation . . . 38
3.2.7 VFC characterizes subtype response to *chd* mutagenesis . . . 40
3.2.8 Identifying the effect of IFNγ stimulation on pancreatic islet cells 44
3.2.9 Analysis of donor-specific composition . . . 48
3.3 Discussion . . . 48
3.4 Author Contributions . . . 50
3.5 Competing Interests . . . 51
3.6 Methods . . . 51
3.6.1 Computation of the sample-associated density estimate . . . 51
3.6.2 Vertex-frequency clustering . . . 70
3.6.3 Parameter search for the MELD algorithm . . . 77
3.6.4 Processing and analysis of the T-cell datasets . . . 77
3.6.5 Processing and analysis of the zebrafish dataset . . . 78
3.6.6 Generation, processing and analysis of the pancreatic islet data . . 79
3.6.7 Quantitative comparisons . . . 80
3.7 Data availability . . . 82
3.8 Code availability . . . 83
3.9 Supplementary Notes . . . 83

4 Finding Archetypal Spaces Using Neural Networks **106**

4.1 Previous work and Background . . . 107

4.2 Introduction . . . 108

4.3 Methods . . . 111

4.3.1 Problem setup . . . 111

4.3.2 The AAnet Framework . . . 112

4.3.3 Code availability . . . 115

4.4 Results . . . 115

4.4.1 Archetypes from a triangle projected onto a sphere . . . 115

4.4.2 Finding archetypes of image translations . . . 116

4.4.3 Generating from the data geometry with AAnet . . . 117

4.4.4 AAnet identifies reproducible archetypes . . . 119

4.4.5 Optimal number of archetypes . . . 120

4.4.6 Latent noise for tight archetypes . . . 121

4.4.7 Runtime . . . 121

4.4.8 Visualizing the archetypal space . . . 122

4.4.9 Characterization of tumor-infiltrating lymphocytes using single-cell sequencing . . . 123

4.4.10 AAnet identifies archetypal states of gut microbiomes . . . 124

4.5 Conclusion . . . 125

4.6 Supplement . . . 126

4.6.1 Neural network parameters . . . 126

4.6.2 Parameters for other methods . . . 127

5 Conclusion **131**

Bibliography **133**

List of Figures

1.1 Application of manifold learning to single-cell biology 4

1.2 Visualization of eigenvectors on scRNA-seq data 5

2.1 Comparison of VFC to other graph clustering methods. 19

2.2 Anomaly detection with VFC 21

2.3 VFC captures biologically relevant populations of beta cells 22

3.1 Overview of the MELD algorithm 28

3.2 Overview of VFC and MELD for single-cell perturbation analysis 36

3.3 Quantitative comparison of the sample-associated relative likelihood and VFC. 39

3.4 MELD recovers signature of TCR activation. 41

3.5 Characterizing chordin Cas9 mutagenesis with MELD. 43

3.6 MELD characterizes the response to IFNγ in pancreatic islet cells. 47

3.7 A step-by-step visual representation of the MELD algorithm 91

3.8 Vertex-Frequency clustering with MELD algorithm schematic 92

3.9 Identifying gene signatures using MELD 93

3.10 Overview of a pipeline for single-cell analysis using MELD 94

3.11 Result of down-sampling on MELD 95

3.12 VFC accurately identifies cell populations affected by a perturbation . . . 96

3.13 Quantitative comparison of clustering algorithms using zebrafish data . . 97
3.14 Quantitative analysis of Cas9 perturbations in T cells 98
3.15 Analysis of replicates within the zebrafish data 99
3.16 Characterization of vertex-frequency clusters in the zebrafish data 100
3.17 Analysis of pancreatic islet cells from three donors 101
3.18 Analysis of islet cell profiles across donors 102
3.19 Source Separation and Parameter Analysis with the MELD filter 103
3.20 Selecting parameters for MELD 104

4.1 Illustrative representation of AAnet 112
4.2 Quantitative analysis of AAnet performance 116
4.3 Comparison of AA methods on dSprites dataset 117
4.4 AAnet samples from data geometry despite uneven sampling 118
4.5 AAnet reproducability analysis 120
4.6 Effect of latent noise on AAnet representation 121
4.7 Comparison of fast MDS interpolation for visualization 123
4.8 AAnet analysis of scRNA-seq data 128
4.9 AAnet analysis of gut microbiome data 129
4.10 Extended results of AA methods on dSprites data 130

List of Tables

3.1 Quantitative comparison of methods for label smoothing over a graph . . 105

3.2 Quantitative comparison of clustering methods to identify the cell types affected by a simulated experimental perturbation using real world data . 105

Chapter 1

Introduction

Recent advances in microfluidic technology facilitate the measurement of gene expression [1, 2], DNA accessibility [3, 4], protein content [5, 6], and genomic mutations [7, 8] across tens of thousands of single-cells. These technologies mark a revolutionary change from the previous half century of single-cell research where techniques such as flow cytometry recorded only one or two dozen features per cell. Standard cytometry analysis involves iterative gating of cells into high or low expression on a per-maker basis. However, the recent increase in number of features provided by modern single-cell techniques requires the development of novel tools for data analysis.

The past several years have seen an explosion in the development of algorithms for single-cell analysis tasks including dimensionality reduction [9–14], data denoising [15–18], data integration [19–24], clustering [25–30], and trajectory inference [31]. Although some methods are purpose built for a specific data modality, an emerging trend is the application of generalizable algorithms, some of which were developed outside the biological sciences. For example, t-SNE, a non-linear dimensionality reduction algorithm, was originally applied to images of handwritten digits and household objects [32]. However, t-SNE is widely applied in the single-cell literature with little adaptation of the original algorithm [10]. To avoid confusion in this dissertation, I will refer primarily to methods

as being applied to single-cell RNA-sequencing (scRNA-seq) data. However, like tSNE, many of these methods can be applied to many other kinds of data across scientific domains.

These generalizable algorithms stand in contrast to specialized statistical methods purpose-build for a specific data modality. To understand these differences, we can compare t-SNE to Zero-Inflated Factor Analysis (ZIFA), a dimensionality reduction algorithm designed specifically for scRNA-seq [33]. ZIFA is latent variable model where expression values in a cell are given by a linear combination of latent factors plus Gaussian noise. However, observing over abundance of zero values in the data matrices of single-cell datasets, ZIFA adds a zero-inflation term that has the potential to set expression of a gene to zero in any cell. While the addition of a zero-inflation term may allow ZIFA to better fit scRNA-seq data, this statistical model is biologically implausible. Zero-inflation implies a molecular process that inhibits all mRNA copies of genes from being captured during library preparation. Although gene-specific reverse transcription and amplification biases exist [34], these biases should act independently at the transcript level. Each transcript should have some independent probability of being detected, whereas zero-inflation acts at the gene level. As such, zero-inflated models are no longer widely used in current single-cell literature [35, 36].

This example serves to illustrate the potential pitfalls of applying restrictive statistical models applied to a nascent data modality. This is not to say that statistical models have no place in single-cell analysis. However, until noise and data generative processes are better understood, the strong assumptions of these models pose a risk of misrepresenting the underlying biology.

One class of generalizable algorithms come from the framework of manifold learning, which models the cellular landscape or manifold in order to understand the potential states that cells can occupy [37]. A manifold is a mathematical model that describes a space that is smooth, differentiable, and locally Euclidean. Measured cellular features,

for example gene counts in scRNA-seq space define a space with tens of thousands of dimensions. This is called the ambient space or the feature space. Relationships between genes are complex and potentially non-linear meaning that the ambient space is likely globally non-Euclidean. However, we understand that cells smoothly progress between states rather than jumping discretely as an electron might change orbitals. This suggests that the landscape of cellular states is smooth and locally Euclidean, like the classic depictions of Waddington's Landscape [38] (**Figure 1.1a**). Unlike those original drawings, we understand the dimensionality of the space of viable cellular states to be high dimensional, but not as high dimensional as the ambient gene space. For example, intrinsic dimensionality in the gene space is reduced by informational redundancy resulting from gene interactions and co-regulation of gene modules. This suggests the cellular manifold has lower dimensionality than the number of genes or proteins in a cell. These constraints on the gene space mean the cellular manifold or landscape can be computed from scRNA-seq data using manifold learning techniques.

Characterizing manifold geometry from discretely sampled data points can be done by calculating distances along a graph computed from the data (**Figure 1.1b**). For droplet-based scRNA-seq, these discretely sampled points are snapshot gene profiles measured using Unique Molecular Identifiers, or UMIs. From these profiles, it is possible to compute a cell similarity graph where nodes represent cells and edges connect closely connected points as calculated using a kernel function. Kernel functions are most commonly encountered when performing kernel density estimation (KDE), where they are used to measure the amount of data in a region of the feature space. These functions can also be used to build a graph by measuring the similarity between data points. Here similarity is the inverse measure of distance. These similarities can be used to calculate an adjacency matrix for a graph. The choice of kernel method preserves local neighborhoods in the data. Distances are then calculated along edges of the graph. It has been shown that in the limit as the number of points grows large distances along a graph calculated over data

sampled from a manifold approximate distances along that manifold [39]. This becomes possible with single-cell methods because the number of data points scales from the tens of thousands to millions.

Manifold theory provides many tools for characterizing data. For example, geometric diffusion has been used in scRNA-seq for data visualization [41, 42], denoising [43], batch normalization [44], and trajectory inference [45]. Data diffusion is related to Markov processes, which are used to model random walks. Applied to a graph, a Markov diffusion operator can be obtained using a row-normalized affinity matrix where the values in each row sums to 1. Eigenvectors of this matrix provide paths through the data that suggest transitions between cellular states that are analogous to the valleys in the Waddington Landscape. Representations of data using these eigenvectors are called Diffusion Maps [41]. **Figure 1.2** shows a PHATE visualization of the data colored by the first two non-trivial

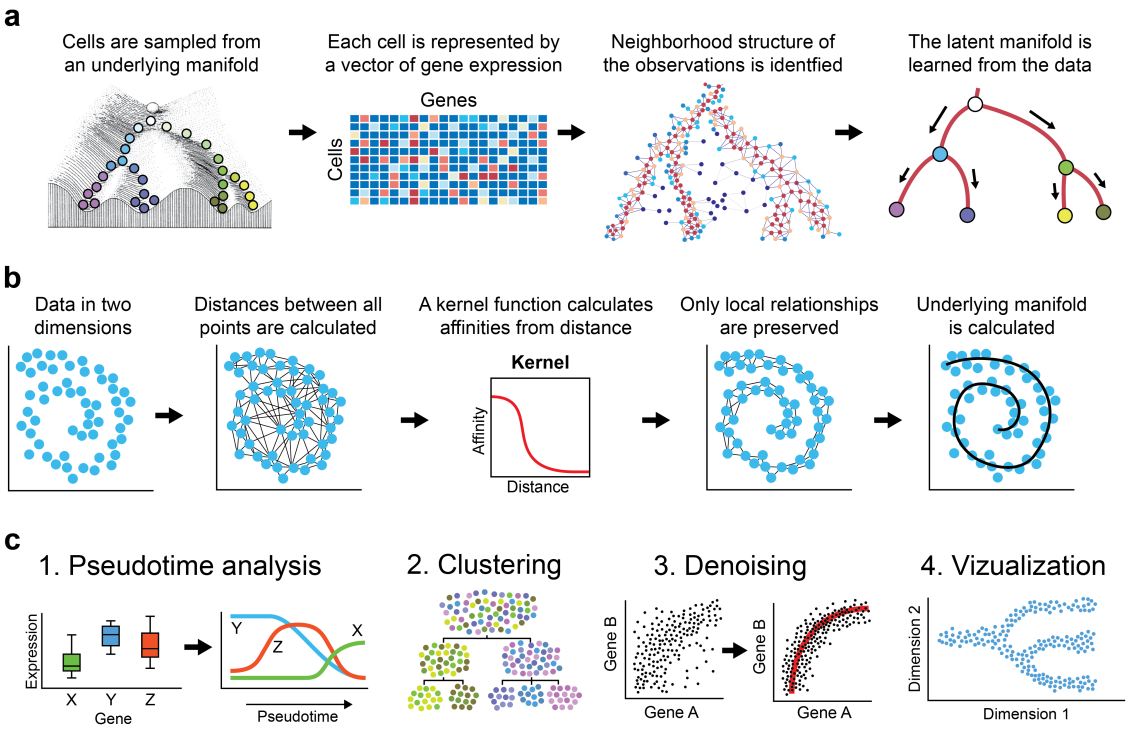

Figure 1.1: Application of manifold learning to single-cell biology. (**a**) In the Waddington Landscape model, cells are likely to be found in states of low free energy depicted as valleys and progress from states of high to low free energy. (**b**) We can recreate these paths from data sampled from this manifold using a graph. (**c**) The graph representation is useful for many downstream analysis tasks.

eigenvectors of the diffusion operator on a graph constructed from 392 cells generated from trans-differentiating fibroblasts by Treutlein et al. [40]. The first non-trivial eigenvector, $\psi 2$, has a maximum near the mouse embryonic fibroblasts (MEF) and minima at the two terminally differentiated clusters, Neuron and Myocyte. Thus, this eigenvector traces the developmental path through trans-differentiation. The next eigenvector, $\psi 3$, has a maximum at the Myocyte cluster and a minimum at the neuronal state. Note, this eigenvector traces a path through the data, but not one that follows a natural developmental

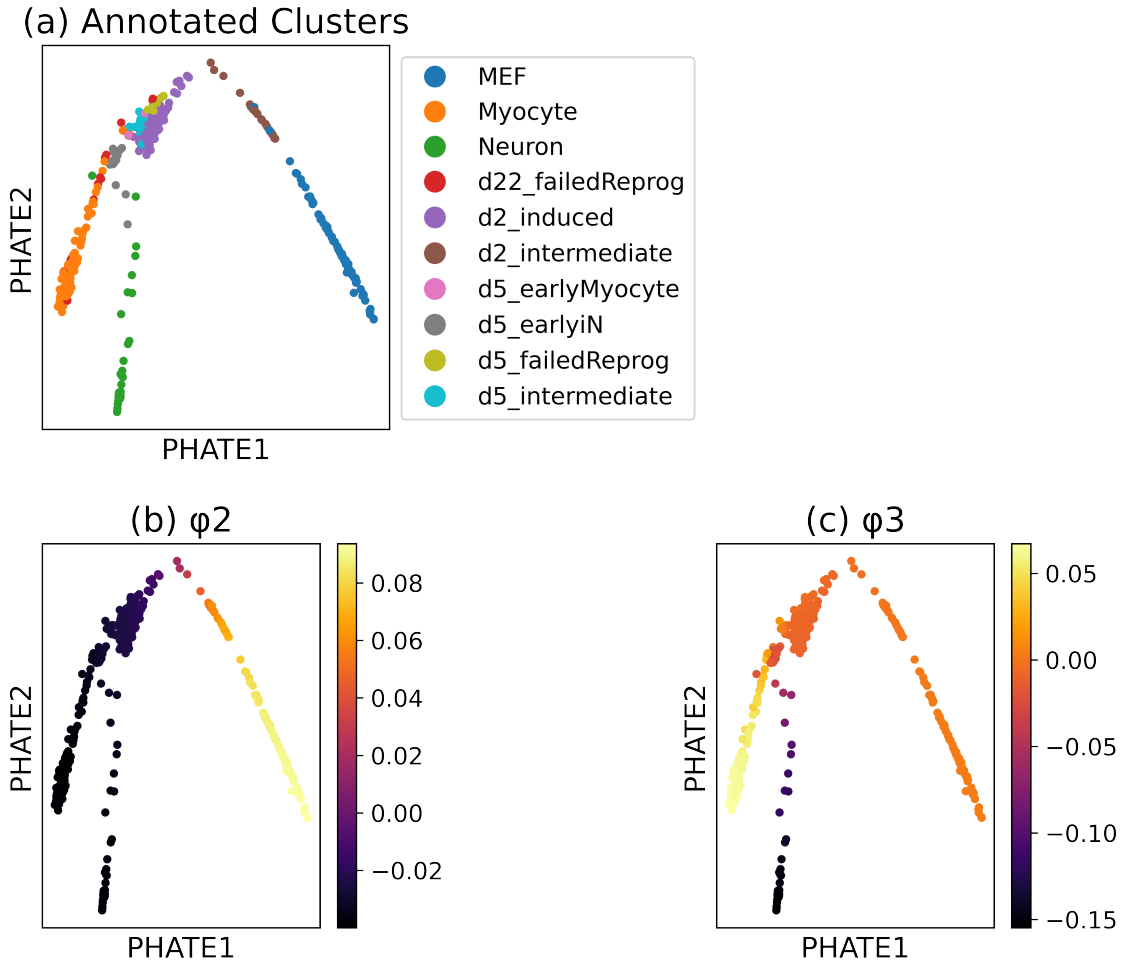

Figure 1.2: Eigenvectors reveal paths through data. (**a**) A PHATE embedding of 329 trans-differentiated fibroblasts colored by cluster labels identified by Treutlein et al. [40]. Cells from the earliest time point are Mouse Embryonic Fibroblasts (MEF) and transition into either neurons or myocytes. (**b, c**) The first two non-trivial eigenvectors of the diffusion operator reveal paths through the data.

progression. To interpret the results of manifold learning techniques, it is often crucial to incorporate prior knowledge about the system under investigation. In this case, we know the transition starts from MEF and terminates at Neuron and Myocyte for $\psi 2$. We also know that each of the extrema of $\psi 3$ indicate a terminal state and that there is no process going from Neuron to Myocyte in this system. This information helps orient inferences made from manifold and diffusion methods.

Another framework for single-cell analysis is deep learning using neural networks. Neural networks are computing systems that consist of complex arrangements and interconnections between simple units called neurons. Each neuron receives a series of inputs that are multiplied by weights and summed. This sum is then passed through a non-linear activation function and used as input to another set of neurons in the next layer. Most neural network arrangements, called architectures, consist of layers of neurons. Generally, all neurons within a layer receive the same set of inputs, but each neuron has different weights associated with each input and therefore activates in response to different patterns of input. Modern neural networks consist of many layers. For example Inception v3, the record-breaking image classification network from Google, is 48 layers deep, hence the term deep learning.

The power of deep learning originates from these networks' potential to learn complex non-linear representations of data. Unlike Principal Components Analysis (PCA) or Diffusion Maps, which apply set transformations of the data to obtain useful representations, neural networks learn representations implicitly as they are optimized to perform a specific task. One of the most fundamental representation learning networks, the autoencoder, is trained to reproduce it's input as output while constraining the network to encode information into a bottleneck layer consisting of many fewer neurons than input features. A common cost function for the autoencoder is mean-squared error between the input and the output. This function is used to calculate how well the network performs a task. Thus, training the network consists of random initialization of weights and biases

for each neuron, followed by evaluation of the network on a batch of input data. Finally, the output is compared to the input using the cost function and the gradient of the cost with respect to each of the weights and biases is calculated. These gradients are then backpropogated through the network such that the cost function is minimized. The steps after initialization are repeated until the network converges, *i.e.* subsequent updates to the network parameters stop improving the cost function.

Although the autoencoder is trained based on it's performance at recreating it's input, the utility of the network is the representation learned in the bottleneck layer. In the standard autoencoder, there is no constraint on the representation in the bottleneck layer. However, the literature in deep learning is built on iterative advances on existing designs. A common approach to modifying an existing neural network is to add constraints. Constraints are terms added to the cost function that alter the representations learned by a network. For example, one could constrain the sum of the weights and biases in a layer of a network using an $L2$ regularization. This term adds the sum of squared weights in a layer of the network to the cost function. Penalizing large weights can prevent a network from overfitting [46]. There are almost as many constraints as neural network papers and they can be used for everything from data denoising [47] to image classification [48]

A powerful autoencoder for single-cell analysis is SAUCIE, a Sparse Autoencoder for Unsupervised Clustering, Imputation and Embedding and batch normalization [19]. SAUCIE contains several novel constraints that allow the network to perform each of these tasks across training modes. For example, one such innovation is Information Dimension (ID) regularization that promotes binary activation of neurons in a layer of the network. This binary encoding for each cell is then used to aggregate cells into clusters for annotation. Another variation of the autoencoder for biological data is the single-cell Variational Inference (scVI) network [22]. scVI is derived from the Variational Autoeconder, which introduces a constraint on the bottleneck representation to encode each point to a Gaussian ball and offers a probabilistic interpretation for the representation. scVI builds on this

framework using four networks to learn both a probabilistic latent representation and a batch-specific noise estimate in the bottleneck layer. Two more networks are used to learn cell-specific scaling, counts, and dropout parameters that are fed to a generative model that recreates the original data. scVI has been further extended across six different tools for various analysis tasks. The flexibility of neural networks provides many opportunities to develop tools for single-cell analysis.

These two frameworks of manifold theory and deep learning can be combined to suggest novel methods to summarize data. Traditional biological data analysis has been dominated by discrete cluster descriptions. A classic example is the collection of discrete haematopoietic states that can be found in a standard immunology textbook. Recent advances in single-cell sequencing have revealed that this process is rather a continuous spectrum of states [49]. More recently, continuum-data summarizations are now proving highly advantageous. One example is archetypal analysis [50, 51], which is a technique that fits a convex hull to the data where the corners of the convex hull represent extrema and other data points are convex mixtures of the extrema. In contrast to clustering that assumes cells occupy distinct and disconnected spaces, archetypal analysis describes a spectrum of cell states.

In this thesis, I will describe three algorithms for single-cell analysis. The first, Vertex Frequency Clustering (VFC) uses graph structure and metadata variables, such as experimental condition, to identify populations of cell with similar spectral characteristics across those labels. When applied to condition labels, VFC identifies population of cells with similar responses to an experimental perturbation. Next, I will present MELD, a method for comparative analysis of single-cell datasets collected from different experimental conditions. Rather than characterize the effect of an experimental treatment at the level of clusters, MELD measures this effect continuously across a graph representation of the data. This provides a single-cell estimate of treatment effect measured as the relative likelihood of observing a cell in each condition. Finally, I will describe AAnet, an autoencoder

with novel regularizations the provides a scalable and non-linear approach to the task of archetypal analysis.

Chapter 2

Vertex Frequency Clustering

Authors

Daniel B. Burkhardt [1*], *Jay S. Stanley III* [2*], *Guy Wolf* [4†], *and Smita Krishnaswamy* [1,3†]

[1] Department of Genetics,

[2] Computational Biology and Bioinformatics Program,

[3] Department of Computer Science, Yale University

[4] Department of Mathematics and Statistics, Université de Montréal

[*,†] These authors contributed equally

Abstract

We propose a novel approach for clustering the vertices of a graph. The method, Vertex-Frequency Clustering (VFC), considers the local harmonic content of one or many graph signals, forming partitions based on spectral features in the input signal. The method can be related to spectral clustering, and the length scale over which frequencies are considered is tunable. This allows one to cluster data based on intrinsic graph geometry in the context of signal dynamics. VFC is useful for unravelling active regions in a signal, collecting sets of similar observations, or detecting anomalies. We demonstrate the utility of VFC

in synthetic and biological data, and show how VFC can be used to identify observations with similar feature sets and signal profiles.

Contribution

I identified the motivating problem for this chapter, trying to identify populations of data with similar distributions of signal distribution. I worked with Jay Stanley to adapt the vertex frequency analysis of Shuman et al. [52] for single-cell data. I led the analysis of the biological and simulated data and designed the figures. Jay and I co-wrote the paper. I am the primary author of the introduction, single-cell analysis, and discussion.

2.1 Introduction

Many modern datasets naturally occur as weighted graphs, such as social networks, sensor networks, or road connectivity maps. While existing graph based clustering methods [26, 29, 53–55] can partition graphs into localized modules, they often do not take into account *signals* on the graph, which can greatly inform clustering in many real-world situations. For instance, partitionings of road maps can be enhanced by taking into account traffic flow patterns, groupings of neurons can become more functionally relevant if accounting for firing or activity signals, and social network groupings can become more coherent if taking into account features of people such as their political beliefs or age demographics. In these examples, the graph is fixed based on some underlying data structure, either geography or anatomy, while the signals are acquired through some independent process. Here, we tackle this problem by using tools from graph signal processing. In particular, we use the frequency domain information of signals on a graph by using a localized graph Fourier transform at each vertex, leading to a vertex-frequency clustering algorithm that provides more informative and interpretable clusterings of large graphs.

We propose a clustering method that considers graph structure in the context of one or multiple graph signals. The method, which we call Vertex-Frequency Clustering (VFC), is constructed through a multiscale representation of the input signal using an extension of the Short Time Fourier Transform (STFT) of classical Euclidean signal processing. This representation uses graph frequencies to localize the frequency content of the graph signal at each node in the graph. By clustering over this representation, VFC partitions the data in the context of signal behavior, separating populations that are smooth or constant in the signal from those where the signal is strongly trending, oscillating, or divergent. In simulations, we use VFC to identify partitions that disentangle the signal of a simulated experimental perturbation, and we use VFC partitions to detect regions of anomalous frequency signatures in graph signals. In these ground-truth datasets we show that VFC achieves superior qualitative and quantitative performance to existing methods. Finally, we apply VFC to separate biologically relevant populations of pancreatic beta cells based on their response to an experimental stimulus.

2.1.1 Prior Works

Generalization of classical signal processing notions to irregular domains is a focus of graph signal processing [56]. In particular, time-frequency frames have been of key interest for extension to the graph setting, where the vertex domain (i.e. the set of nodes in a graph) is treated analogously to the time domain. Eigenvectors of the graph Laplacian are treated as the harmonics of the graph. The seminal work of [41] established diffusion wavelets, a set of orthonormal wavelet frames for irregular domains composed of a sequence of decompositions and low rank approximations. Later works by [57] treated vertex-frequency analysis by defining wavelets directly in the graph spectral domain. Subsequently, a flurry of works have developed wavelets with desirable features such as tight frames [58] that adapt to the density of Laplacian eigenvalues [59], and fast decomposi-

tions [60].

The first approach to incorporating graph signal frequency for clustering was proposed in [52], and it has direct roots to spectral clustering [61]. The notion of using vertex-frequency analysis for localizing signal components was discussed in [62], but clustering was not the focus of this chapter. Here, we extend the work of [52], which proposed the windowed graph Fourier transform (WGFT).

2.1.2 Notation

Let $\mathbf{v} = [v(i)]_{i=1}^{N}$ be a vector in $\mathbb{R}^N$ with i-th entry $v(i) \in \mathbb{R}$. Construct a set of vectors, $V = \{\mathbf{v}_j : \mathbf{v}_j \in \mathbb{R}^N\}_{j=1}^{D}$. Then we may construct a $D \times N$ matrix $\mathbf{V} = [\mathbf{v_j}]_{j=1}^{D}$ with $\mathbf{v}_j$ as its j-th column. We index this column by $\mathbf{V}_{(\cdot,j)} = \mathbf{v}_j$. Similarly we index the i-th row by $\mathbf{V}_{(i,\cdot)} = [v_j(i)]_{j=1}^{D}$, and let $\mathbf{V}_{(i,j)} = v_j(i)$. Let $\{x, y\} \subset \mathbb{R}^N$.

Define $\mathbf{x} \circ \mathbf{y}$ as the Hadamard (element-wise) product of two vectors. Choosing $\mathbf{A} \in \mathbb{R}^{N \times N}$, we define $[\mathbf{A} \odot \mathbf{y}]_{(i,\cdot)} = \mathbf{A}_{(i,\cdot)} \circ \mathbf{y}$, the row-wise Hadamard product. Note that $\mathbf{V} = \mathbf{I} \odot [\alpha_i]_{i=1}^{N}$ for $\alpha_i \in \mathbb{R}$ is a diagonal matrix with $\mathbf{V}_{(i,i)} = \alpha_i$.

2.2 Background

2.2.1 Graph Signal Processing

For a undirected, weighted graph with vertex set $V = \{v_i\}_{i=1}^{N}$ and edge weights $W : V^2 \mapsto \mathbb{R}$, the graph Laplacian $\mathcal{L}$ is a difference operator on the space of functions $g : V \mapsto \mathbb{R}$. The functions in this space can be treated as *graph signals* defines over each vertex. The graph Laplacian can be used for harmonic analysis on graphs in analogy to the Euclidean setting [56]. These tools form the field of graph signal processing, which uses the fact that the Laplacian eigenbasis is interpretable as a Fourier basis with frequencies $\Lambda = \{\lambda_i\}_{i=1}^{N}$

and corresponding harmonics $\Psi = \{\psi_i\}_{i=1}^N$[1].

We use the normalized Laplace operator $\mathcal{L}$. Let $d(i) = \sum_j W(v_i, v_j)$. Then $\mathcal{L}$ satisfies $(\mathcal{L}f_k)_i = \frac{1}{\sqrt{d(i)}} \sum_{x_j \in X} \left(\frac{f_k(i)}{\sqrt{d(i)}} - \frac{f_k(j)}{\sqrt{d(j)}} \right)$. The eigenvalues of this operator satisfy $0 \leq \lambda \leq 2$, and its eigenfunctions can be used to describe random-walk processes on graphs. More concretely, let $\mathbf{D}$ be the diagonal degree matrix with $\mathbf{D}_{(i,i)} = d(i)$ and let $\mathbf{W}$ be the weight matrix with $\mathbf{W}_{(i,j)} = W(v_i, v_j)$. Then one choice of random-walk operator is $\mathbf{M} = \mathbf{D}^{-1}\mathbf{W}$. Letting μ_i and ϕ_i be the i-th eigenvalues and eigenvectors of this matrix, one has the relations $\mu_i = 1 - \lambda_i$ and $\phi_i = D^{-1/2}\psi_i$.

2.2.2 Vertex-Frequency Analysis

Briefly, we review the construction of the generalized translation and modulation by [52]. These tools are used to build a vertex-frequency transform.

The derivation of the STFT relies on well-defined notions of translation and modulation. For general graph signals, these concepts cannot be applied due to vertex irregularity. However, for certain classes of functions, generalized translation and modulation can be performed. These functions, which must satisfy localization in both the vertex and frequency domain, can be translated and modulated via a generalization of convolution developed by [52]. This generalization allows us to define the WGFT by using low-frequency kernels centered at every vertex.

To form a WGFT frame, a window kernel is translated to each vertex i, $g_i(n) = (T_i g)(n) = \sqrt{N} \sum_{\ell=1}^N \hat{g}(\lambda_\ell)\psi_\ell(i)\psi_\ell(n)$, and modulated to every frequency k via $g_{i,k}(n) = (M_k g_i)(n) = \sqrt{N}\psi_k(n)g_i(n)$. [52] proposed the use of the heat kernel $\hat{g}(\lambda) = \exp(-t\lambda)$ for a window function. This construction forms a frame, which can be used for vertex-frequency analysis via the spectrogram matrix $S_t f(i,k) = [\mathbf{S}_t f]_{i,k} = \langle f, g_{i,k}^t \rangle$. We refer the reader to [52] for a detailed description of generalized convolution, translation, and

[1] We will treat the eigenvectors of $\mathcal{L}$ as a set and as a matrix with each eigenvector as a column.

modulation on graphs.

Finally, in [52], the authors demonstrate a toy example of "signal-biased spectral clustering" (SBSC) using the features generated by the WGFT. In brief, their algorithm proceeds as follows:

1. Generate a WGFT of the input signal, $S_t(f) = \mathbf{S}_t f$,

2. Apply an element-wise nonlinear activation function,
   $\mathbf{S}'_t f = \tanh(\alpha|\mathbf{S}_t f|)$,

3. Perform *k*-means clustering over the transformed points
   $Y = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N : \mathbf{y}_i = [\mathbf{S}'_t f]_{(i,\cdot)}\}$

Building on this work, we propose a method to analyze graph signal frequencies at many scales using a multi-resolution adaptation of the WGFT. We also use a PCA-based dimensionality reduction of this frequency information to produce a reduced feature space that can be used for efficient data partitioning.

2.3 Vertex-Frequency Clustering

2.3.1 Problem Statement

We seek to form k partitions of a dataset $X = \{x_i\}_{i=1}^N$ that capture the dynamics of one or many signals $F = \{f_i : X \mapsto \mathbb{R}\}_{i=1}^d$ relative to the latent geometry of X. This geometry is captured by an undirected, weighted graph with vertex set $V = X$ and edge weights $W : V^2 \mapsto \mathbb{R}$.[2] Our proposed approach leverages the joint localization of graph signals provided by the WGFT to cluster the data in a basis that describes both vertex and

[2]In this formulation we consider clustering the vertices of a graph. When no a priori graph exists for datasets of the form $X = \{x_i : x_i \in \mathbb{R}^D\}_{i=1}^N$, we construct a graph by defining a vertex for each data point x_i. The weights of this graph are then given by a Mercer kernel $\mathcal{K} : \mathbb{R}^{2\times D} \mapsto \mathbb{R}$ such that $W(x_i, x_j) = \mathcal{K}(x_i, x_j)$

frequency patterns. These patterns are obtained by aggregating Fourier features from the input signals over many vertex scales. Then, the aggregated features are reduced using principal components analysis to obtain salient directions of variance in the frequency space. The result of this is clustered using *k*-Means.

2.3.2 Frame Selection

We begin by noting that one may define the WGFT as the graph Fourier transform of a set of overlapping vertex slices of an input signal x. Letting $\mathbf{g}_i$ be a vector that contains the values of the window at each vertex around i, windowing may be written as $\mathbf{g}_i \circ \mathbf{f}$. Subsequently, the Fourier transform of the windowed signal $\widehat{\mathbf{g}_i \circ \mathbf{f}}$ is taken, yielding a set of frequency coefficients for each vertex slice. Using this second definition, the WGFT can be expressed using a filter matrix $\mathbf{P}^t$ that satisfies the vertex and frequency localization requirements of [52]. In the case of the heat kernel, $\mathbf{P}^t = \exp(-t\mathcal{L})$ such that $\mathbf{g}_i = \mathbf{P}^t_{(i,\cdot)}$. However, as $\exp(-t\mathcal{L})$ is computationally expensive to calculate, we note that the rows of a Markov matrix $\mathbf{P} = \mathbf{D}^{-1}\mathbf{W} = \mathbf{M}$ may be used analogously[3]. Then, the WGFT is

$$S_t(f) = \Psi^T \left[\mathbf{P}^t \odot \mathbf{f}\right]^T = \left[\mathbf{P}^t \odot \mathbf{f}\right] \Psi. \tag{2.1}$$

From this, we have the following lemma, which concerns the window scale of the WGFT:

Lemma 1 (Fourier scaling[4])**.** *Define $S_t(f) = \mathbf{S}_t$ as the windowed graph Fourier transform of a signal f with diagonalizable window matrix $\mathbf{P}$. Then*

$$\lim_{t \to 0} S_t(f) = \mathbf{f}^T \odot \Psi$$

Proof. For any diagonalizable matrix $\mathbf{A}^0 = \mathbf{U}\mathbf{V}^0\mathbf{U}^{-1} = \mathbf{I}$, so

[3]This construction relies on a graph with self-loops, as is the case with kernel matrices. This requirement may be relaxed by considering the related *lazy random walk* matrix, $\mathbf{P} = \frac{1}{2}\left(\mathbf{I} + \mathbf{D}^{1/2}\mathbf{M}\mathbf{D}^{-1/2}\right)$

[4]Note that this is an elaboration of a suggested result in [52]

$$\lim_{t\to 0} S_t(f) = \left[\mathbf{P}^0 \odot \mathbf{f}\right] \Psi = \left[\mathbf{I} \odot f\right] \Psi = \mathbf{f}^T \odot \Psi.$$

□

Thus, in the small scale limit the WGFT acts as a rescaling of the eigenvectors of the graph Laplacian. Consequently, if one has $\mathbf{f} = \mathbb{1}$, the all-ones vector, then $\lim_{t\to 0} S_t(f) = \Psi$, and SBSC amounts to traditional spectral clustering with an activation function.

2.3.3 VFC

Our proposed approach, VFC, is built by considering geometry over many scales. As clusters can manifest over many scales in a graph, we create a sequence of WGFT frames at Ω scales starting from $t = p$ by the collection $G = \{\mathbf{P}^{2^t}\}_{t=p}^{\Omega}$ to represent the graph over multiple geometric scales. The *diffusion time*, t, controls the radius of the geometry considered for each frame set.

We use this set of frames to analyze each signal in the ensemble $f \in F$. The application of each frame is given by via the inner product $\left[\mathbf{S}_t f\right]_{(i,k)} = \langle f, g_{i,k}^t \rangle = \left[\mathbf{P}_{(i,\cdot)}^t \circ f\right] \psi_k$. Each spectrogram thus contains the vertex-frequency analysis of an individual signal in the ensemble at a specific scale. As we are primarily interested in patterns and frequency localization, we use the common classical signal processing trick of introducing element-wise nonlinearity to each scale. We combine all the scales for a given signal f using

$$\mathbf{S}f = \sum_{t=p}^{\Omega-1} \tanh(|\mathbf{S}_{2^t} f|). \tag{2.2}$$

One interpretation for the nonlinearity in this step is that it removes the *phase* from the graph signal, and allows the algorithm to only consider raw frequency amplitude. If phase was of interest we would concatenate the signal, as shown in Section 2.5.

Finally, one constructs $\mathbf{S}f$ for each $f \in F$, combining these via the vertex-wise (hori-

zontal) concatenation $\mathbf{S}' = [\mathbf{S}f_i]_{f_i \in F}$. This $N \times (N \times |F|)$ matrix contains Fourier features aggregated across very local scales (which, according to Lemma 1 can converge to spectral clustering for small p) to very large scales (as $\Omega \to N$, features become close to the GFT). In practice, we have found that $\Omega = 6$ and $p = 0$ delivers stable clustering.

The aggregate Fourier features contained in $\mathbf{S}'$ are subsequently reduced to salient directions using principle component analysis, i.e.

$$Y = \{\mathbf{y}_i : \mathbf{y}_i = [v_j \mathbf{u}_j(i)]_{j=1}^{k+1}\}_{i=1}^{N}.$$

where $\mathbf{U} = [\mathbf{u}_i]_{i=1}^{N}$ and $\mathbf{V} = \mathbf{I} \odot [v_i]_{i=1}^{N}$ are the eigenvector and eigenvalue matrices of the covariance matrix $\mathbf{C} = \mathbf{S}'^T\mathbf{S}'$. This step allows the algorithm to learn *patterns* in the frequency domain, which will be summarized along principal components. In contrast, wavelet-based constructions will be restricted to contiguous regions of frequency space. Finally, we apply *k*-Means to the transformed data Y.

2.4 Synthetic Examples

2.4.1 Gaussian Mixtures

As SBSC depends on a fixed window size, the clusters that the algorithm captures come from a single scale. We found that this leads to instabilities when performing exploratory analysis, as the choice of window size t is inflexible to cluster size in both the vertex and frequency domains. In figure 2.1, we attempt to cluster a mixture of Gaussians sampled from two experimental conditions (± 1 in **Fig. 2.1a**). This mixture represents disparate responses of two populations to the condition. In the first population (top of **Fig. 2.1a**), the condition label is completely mixed, meant to exemplify a population that did not "respond" to the experimental treatment. In contrast, the second population (bottom of

Fig. 2.1a) transitions between two well-separated condition clusters. In this example, the goal is to partition the data into 4 profiles such that the unresponsive population (1) is separated from the transitioning population, which is partitioned into two pure regions (left: 2, right: 3) and the middle transitioning region (4). Traditional spectral clustering (**Fig. 2.1b**) without the input signal according to the Ng algorithm [61] captured population 1, 2, and 3, but created a spurious cluster for the 4th partition. On the other hand, SBSC with window size of $t = 1$ partitions the data across a single frequency scale, creating a single cluster for population 1 and population 4 (**Fig. 2.1c**). This can be attributed due to the high frequency content of both populations. We note that sufficient window size tuning could produce the desired clusters in this example, but in exploratory analysis ground true cluster granularity is rarely known.

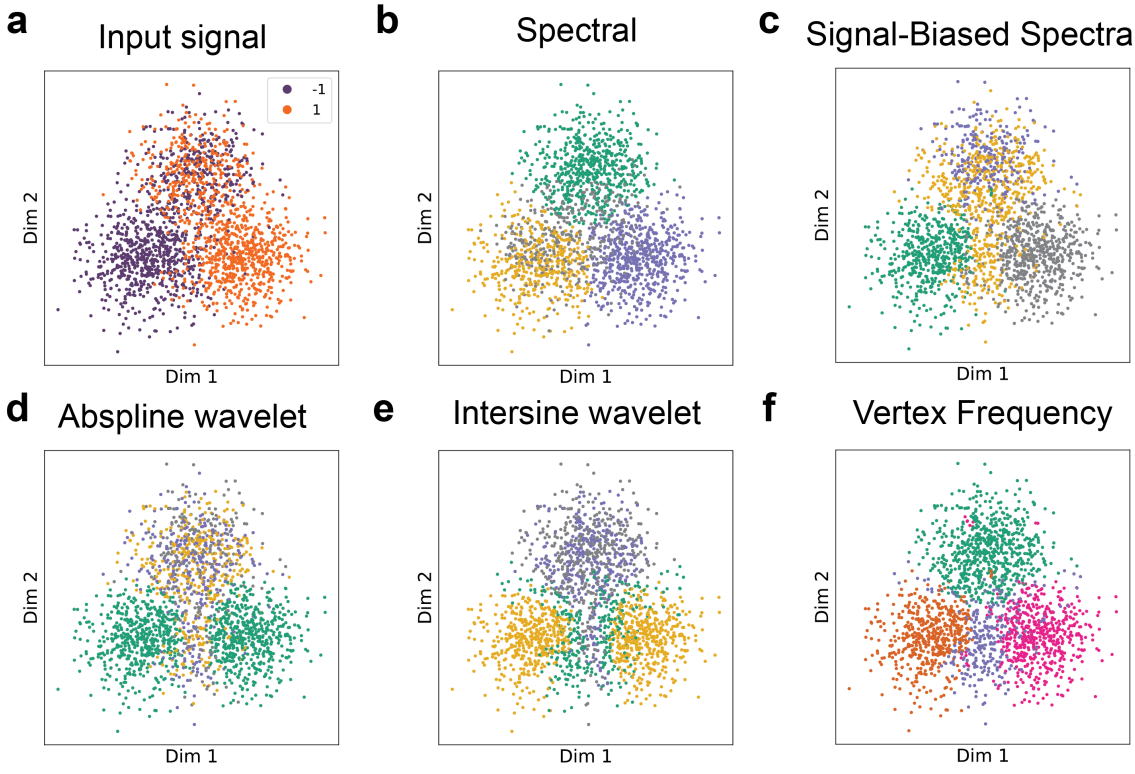

Figure 2.1: Comparison of VFC to other graph clustering methods.

A natural alternative to the WGFT for vertex-frequency analysis across multiple scales are spectral graph wavelets. We hypothesized that clustering the data using signal scalo-

grams would be robust to the single-scale limitations of SBSC. Figure 2.1d and 2.1e demonstrate the result of scalogram clustering using 20 filters using the Abspline wavelets of [57] and the tight frame itersine wavelets of [63]. We note that each wavelet construction groups both population 2 and 3 together due to the similarity in the low-order frequency content of both. However, abspline wavelets (**Fig. 2.1d**) fail to resolve meaningful differences amongst the remaining three higher-order clusters. We hypothesized that this could be due to the frame bounds of the abspline wavelets, which lead to non-uniform representation of the signal across the graph frequency domain. Itersine wavelets (**Fig. 2.1e**) are tight frame and do not suffer from this instability. Interestingly, this construction partitioned the transitioning population 4 based on medium frequency content on the periphery of populations 2 and 3. Despite this, each of the proposed methods were insufficient to recover the appropriate paritioning of the data. The observation that the aforementioned representations could not robustly capture a simple gaussian mixture scenario led us to design a robust and expressive representation of the data. In figure 2.1f we demonstrate clustering using our proposed representation, which we call Vertex-Frequency Clustering (VFC). The clusters obtained by VFC are mostly contiguous in the vertex domain, but represent the three disparate frequency regimes in the data.

2.4.2 Anomaly Detection

One application of VFC is in anomaly detection. In this setting, one wishes to capture a small region of the graph that whose behavior is an outlier with respect to some observation signal. Such behavior may manifest itself in the frequency spectrum, where the frequency signature of anomalous vertices will be disparate from their neighbors. In figure 2.2 we explore the ability of VFC to isolate an anomalous cluster. In brief, the goal in this scenario is to separate the anomaly from the background data points in the vertex domain based on its frequency profile alone. In figure 2.2a we show that VFC is robust to a wide range of

anomaly sizes in the vertex domain. In figure 2.2b we show that the method scales better than SBSC in terms of the bandwidth size of the anomaly. Taken together, these results indicate that VFC could be used to explore new methods for anomaly detection on graphs.

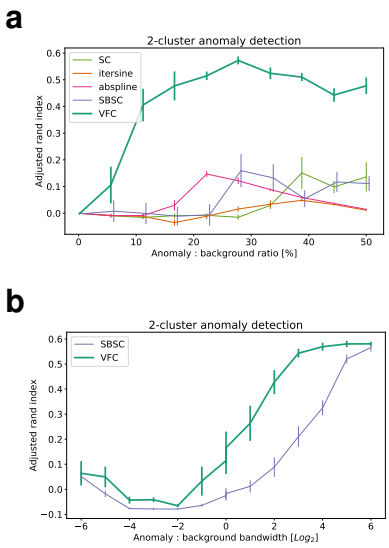

Figure 2.2: Anomaly detection with VFC. (**a**) $N = 750$ points were sampled from a Gaussian and a graph was built over the data. A sampling of points from the center of the Gaussian (x-axis) were selected to be an anomaly, and a high frequency signal (bandwidth = 60) was generated on them. This signal was combined with a low frequency signal (bandwidth = 20) which was located on the rest of the graph. Then, clustering was performed using the indicated method and the adjusted rand index relative to the anomaly indicator vector was recorded. (**b**) A similar experiment to (**a**) was performed, this time varying the ratio of the anomaly signal bandwidth to the low frequency background. Wavelets and spectral clustering were left out due to poor performance in (**a**).

2.5 Analysis of single-cell RNA sequencing

To demonstrate the ability for VFC to identify biological populations with various responses to perturbation, we analyzed two scRNA-seq samples of human pancreatic islets stimulated with interferon-gamma (IFNg). Human islets from a single donor were cultured for 24 hours with or without IFNg before collection for scRNA-seq. We visualized the data using PHATE data[11]. To obtain an input signal f, cells from the IFNg condition were assigned a value of $+1$ and cell from the control condition were assigned a value of -1 (**Fig. 2.3a**). In [64], we recently showed that this approach is useful for understanding

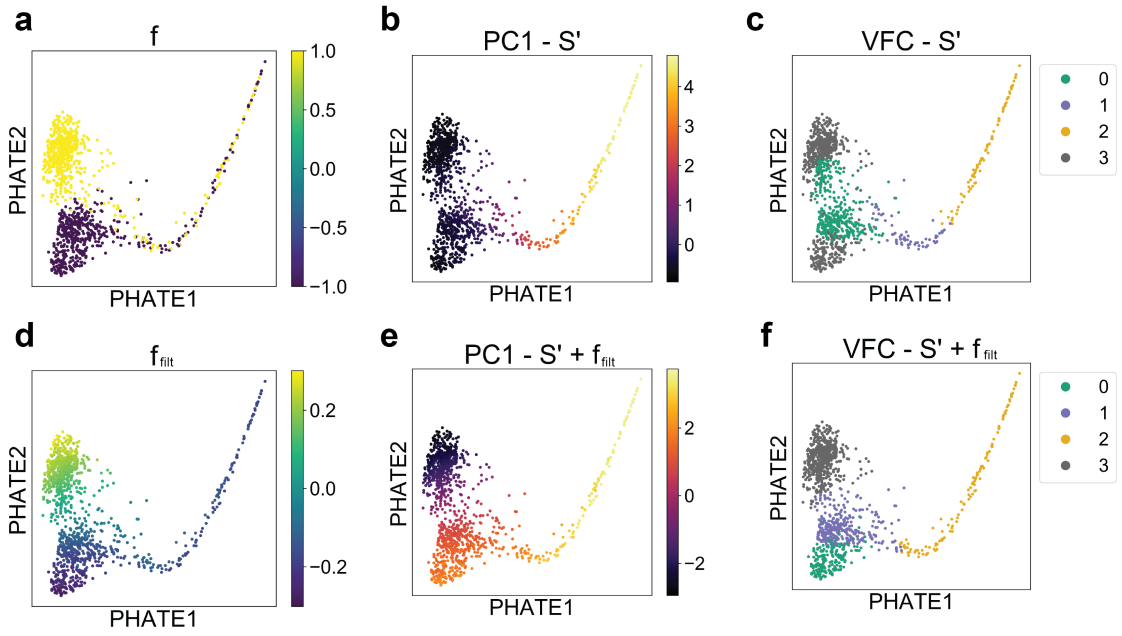

Figure 2.3: VFC captures biologically relevant populations of beta cells. (**a**) The input signal f denotes if a cell recieved IFNg ($f = 1$) or did not ($f = -1$). (**b**) We then calculate the spectrogram S' of f and plot the first principle component of S' on a PHATE embedding. (**c**) KMeans on S' produces partitions of the graph with uniform frequency. (**d**) The smoothed signal f_{filt}. (**e**) PC1 of the concatenation of f_{filt} to S', highlights signal frequency and sign. (**f**) Clustering on f_{filt} and S' produces clusters that distinguish our four conjectured populations.

the experimental conditions of a single-cell experiment.

Examining the distribution of f on a PHATE plot, we conjectured that there are four ideal partitions of the data: two clusters only found in the IFNg or control conditions, respectively, a third cluster transitioning between these two clusters, and a fourth cluster projecting off the main group of cells with uniform mixing of cells from either condition. We then calculated the combined spectrogram $\mathbf{S}'$ of f and perform clustering on this matrix setting *k=4* (**Fig. 2.3b,c**). We observed that VFC on $\mathbf{S}'$ alone partitioned each of the cells that are exclusively found in the IFNg or control conditions in a single cluster. Examining the first principle component (PC1) of $\mathbf{S}'$, we see that indeed these two populations of cells have similar frequencies despite being localized in different regions of the graph.

To obtain our conjectured clusters, we reasoned that we should consider not only the frequency of the input signal, but also the sign of the smoothed signal f_{filt}. We used the

graph filter proposed in [64] to filter f. Extreme values of f_{filt} denote areas of the graph enriched in either condition (**Fig. 2.3d**). To incorporate this information, we concatenated f_{filt} and $\mathbf{S}'$. PC1 of this concatenated matrix separates the four conjectured populations, and clustering on this concatenated matrix produces ideal cluster assignments (**Fig. 2.3e, f**).

To confirm these clusters are biologically relevant, we examined expression of STAT1 and IRF1, two genes known to be upregulated in response to IFNg stimulation [65]. We found that, in the clusters found from the concatenation of f_{filt} and $\mathbf{S}'$, clusters 0 and 2 have the lowest expression of the IFNg-induced genes, cluster 1 has intermediate expression of STAT1 and IRF1, and cluster 3 has the highest expression of these two genes. These results indicate that cluster 2, the group of cells with uniform mixture from either condition, are unaffected by the IFNg treatment. We also find that these cells are marked by extreme high insulin expression. Recent studies have described a subpopulation of beta cells marked by high insulin mRNA production that are hypothesized to have functional differences to typical beta cells [66].

2.6 Discussion

We present a novel clustering algorithm, Vertex-Frequency Clustering (VFC), that partitions the vertices of a graph in the context of the local Fourier content of a signal. This method has the potential to identify regions of a graph with divergent signal frequency composition. On synthetic and biological data, VFC outperforms existing graph clustering methods including existing and proposed frequency-biased clustering approaches. Applied to a single-cell RNA-sequencing dataset, VFC identifies a cluster corresponding to a recently-described subpopulation of pancreatic beta cells. In [64] we apply this method for deeper single-cell RNA sequencing analysis. Future works could enhance the algorithm using fast approximate transforms such as [67] and [60].

Chapter 3

Quantifying the effect of experimental perturbations at single-cell resolution

Authors

Daniel B. Burkhardt[1,†], Jay S. Stanley III[2,†], Alexander Tong[3], Ana Luisa Perdigoto[4], Scott A. Gigante[2], Kevan C. Herold[4], Guy Wolf[6,7,‡], Antonio J. Giraldez[1,‡], David van Dijk[5,‡,**], Smita Krishnaswamy[1,3,‡*]

[1]Department of Genetics; [2]Computational Biology & Bioinformatics Program; [3]Department of Computer Science; [4]Department of Immunobiology; [5]Department of Internal Medicine (Cardiology); Yale University, New Haven, CT, USA

[6]Department of Mathematics and Statistics, Université de Montréal, Montreal, QC, Canada

[7]Mila – Quebec AI Institute, Montreal, QC, Canada

[†] These authors contributed equally. [‡] These authors contributed equally.

Abstract

Current methods for comparing single-cell RNA sequencing datasets collected in multiple conditions focus on discrete regions of the transcriptional state space, such as clusters of cells. Here we quantify the effects of perturbations at the single-cell level using a contin-

uous measure of the effect of a perturbation across the transcriptomic space. We describe this space as a manifold and develop a relative likelihood estimate of observing each cell in each of the experimental conditions using graph signal processing. This likelihood estimate can be used to identify cell populations specifically affected by a perturbation. We also develop vertex frequency clustering to extract populations of affected cells at the level of granularity that matches the perturbation response. The accuracy of our algorithm at identifying clusters of cells that are enriched or depleted in each condition is, on average, 57% higher than the next-best-performing algorithm tested. Gene signatures derived from these clusters are more accurate than those of six alternative algorithms in ground truth comparisons.

Contribution

This project started during a laboratory hackathon in December 2017. During that event I identified the potential for metadata label smoothing over a graph, initially proposed by Dr. David van Dijk and Dr. Smita Krishnaswamy, as a measure of an experimental perturbation. From that point forward I took a leadership position driving the development of MELD. I led the algorithmic development and designed all experiments and figures with the exception of the signal separation experiment in Figure 3.20. I wrote the manuscript text and was helped on the methods section from Jay Stanley and Alexander Tong.

3.1 Introduction

As single-cell RNA-sequencing (scRNA-seq) has become more accessible, the design of single-cell experiments has become increasingly complex. Researchers regularly use scRNA-seq to quantify the effect of a drug, gene knockout, or other experimental perturbation on a biological system. However, quantifying the differences between single-

cell datasets collected from multiple experimental conditions remains an analytical challenge [68]. This task is hindered by biological heterogeneity, technical noise, and uneven exposure to a perturbation. Furthermore, each single-cell dataset comprises several intrinsic structures of heterogeneous cells, and the effect of the treatment condition could be diffuse across all cells or isolated to particular populations. To address this, we develop a method that quantifies the probability that each cell state would be observed in a given sample condition.

Our goal is to quantify the effect of an experimental perturbation on every cell observed in matched treatment and control scRNA-seq samples of the same biological system. We begin by modelling the cellular transcriptomic state space as a smooth low-dimensional manifold or set of manifolds. This approach has been previously applied to characterize cellular heterogeneity and dynamic biological processes in single-cell data [14, 11, 17, 69, 26, 29, 9]. We then define and calculate a *sample-associated density estimate*, which quantifies the density of each sample over the manifold of cell states. We then consider differences in the sample-associated density estimates for each cell to calculate a *sample-associated relative likelihood*, which quantifies the effect of an experimental perturbation as the likelihood of observing each cell in each experimental condition (**Figure 3.1**).

Almost all previous work quantifying differences between single-cell datasets relies on discrete partitioning of the data prior to downstream analysis [70–73, 8, 74–76]. First, datasets are merged applying either batch normalization [75, 76] or a simple concatenation of data matrices [70–73, 8, 74]. Next, clusters are identified by grouping either sets of cells or modules of genes. Finally, within each cluster, the cells from each condition are used to calculate statistical measures, such as fold-change between samples. However, reducing experimental analysis to the level of clusters sacrifices the power of single-cell data. We demonstrate cases where subsets of a cluster exhibit divergent responses to a perturbation that were missed in published analysis that was limited to clusters derived using data geometry alone. Instead of quantifying the effect of a perturbation within clusters, we focus

on the level of single-cells.

In the sections that follow, we show that the sample-associated relative likelihood has useful information for the analysis of experimental conditions in scRNA-seq. First, the relative likelihoods of each condition can be used to identify the cell states most and least affected by an experimental treatment. Second, we show that the frequency composition of the sample label and the relative likelihood scores can be used as the basis for a clustering algorithm we call *vertex frequency clustering* (VFC). VFC identifies populations of cells that are similarly affected (either enriched, depleted, or unchanged) between conditions at the level of granularity of the perturbation response. Third, we obtain gene signatures of a perturbation by performing differential expression between vertex frequency clusters.

We call the algorithm to calculate the sample-associated density estimate and relative likelihood the MELD algorithm, so named for its utility in joint analysis of single-cell datasets. The MELD and VFC algorithms are provided in an open-source Python package available on GitHub at `https://github.com/KrishnaswamyLab/MELD`.

3.2 Results

3.2.1 Overview of the MELD algorithm

We propose a framework for quantifying differences in cell states observed across single-cell samples. The power of scRNA-seq as a measure of an experimental treatment is that it provides samples of cell state at thousands to millions of points across the transcriptomic space in varying experimental conditions. Our approach is inspired by recent successes in applying manifold learning to scRNA-seq analysis [37]. The manifold model is a useful approximation for the transcriptomic space because biologically valid cellular states are intrinsically low-dimensional with smooth transitions between similar states. In this context, our goal is to quantify the change in enrichment of cell states along the underlying

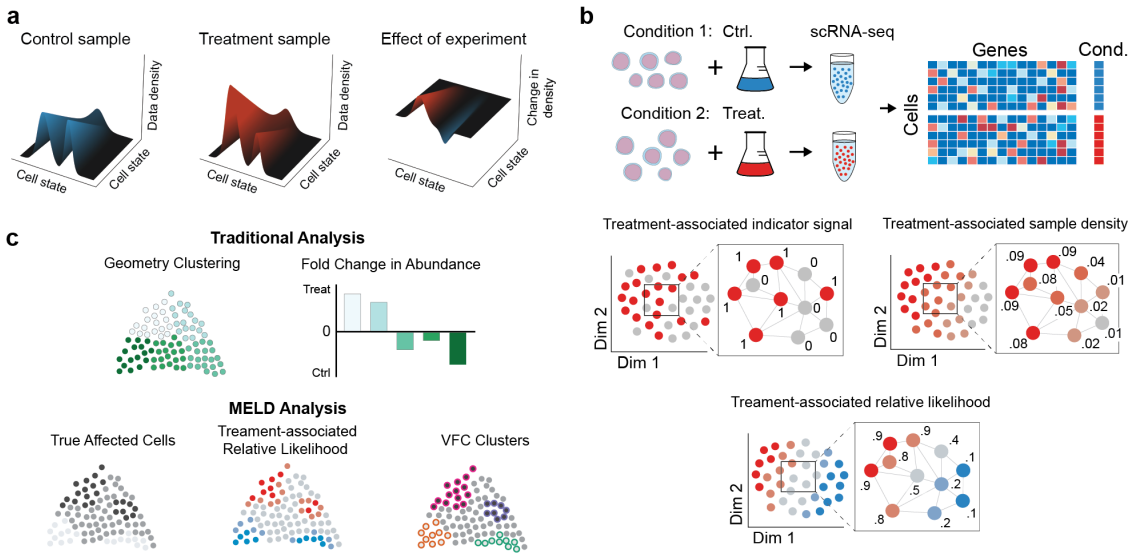

Figure 3.1: (**a**) To quantify the effect of an experiment, we model single-cell experiments as samples from a probability density function (pdf) over the underlying transcriptomic cell state space manifold. The pdf for the control sample is the frequency with which cell states are observed in the control sample compared to the overall frequency of the cell state in both samples combined. In this context, the effect of an experimental perturbation is to alter this probability density and thus the data density in the treatment sample relative to the control. Therefore, the effect of an experimental perturbation can be quantified as the change in the probability density in the experiment condition relative to the control. (**b**) The sample-associated relative likelihood quantifies this effect by computing a kernel density estimate over the cell similarity graph using graph signals representing indicator vectors for each sample. The sample-associated relative likelihood indicates the likelihood that a particular cell is from the treatment or control conditions. (**c**) In traditional analysis of scRNA-seq datasets, the clusters are based solely on the data geometry and changes in abundance between conditions may not align with the true affected populations. Using the sample-associated relative likelihood and VFC, we can identify the correct cluster resolution for downstream analysis.

cellular manifold as a result of the experimental treatment (**Figure 3.1**).

For an intuitive understanding, we first consider a simple experiment with one sample from a treatment condition and one sample from a control condition. Here, sample refers to a library of scRNA-seq profiles, and condition refers to a particular configuration of experimental variables. In this simple experiment, our goal is to calculate the relative likelihood that each cell would be observed in either the treatment or control condition over a manifold approximated from all cells from both conditions. This relative likelihood can be used as a measure of the effect of the experimental perturbation because it indicates for each cell how much more likely we are to observe that cell state in the treatment condition relative to the control condition (**Figure 3.1**). We refer to this ratio as the

sample-associated relative likelihood. The steps to calculate the sample-associated relative likelihood are given in **Algorithm 1** and a visual depiction can be found in **Figure S1**.

As has been done previously, we first approximate the cellular manifold by constructing an affinity graph between cells from all samples [14, 11, 17, 69, 26, 29, 9]. In this graph, each node corresponds to a cell, and the edges between nodes describe the transcriptional similarity between the cells. We then estimate the density of each sample over the graph using graph signal processing [56]. A graph signal is any function that has a defined value for each node in a graph. Here we use labels indicating the sample origin of each cell to develop a collection of one-hot indicator signals over the graph with one signal per sample. Each indicator signal has value 1 associated with each cell from the corresponding sample and value 0 elsewhere. In a simple two-sample experiment, the sample indicator signals would comprise two one-hot signals, one for the control sample and one for the treatment sample. These one-hot signals are column-wise L1 normalized to account for different numbers of cells sequenced in each sample. After normalization, each indicator signal represents an empirical probability density over the graph for the corresponding sample. We next use these normalized indicator signals to calculate a kernel density estimate of each sample over the graph.

Algorithm 1: The MELD algorithm

Input: Dataset $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}, \mathbf{x}_i \in \mathbb{R}^m$; Condition labels $\mathbf{y}$ s.t. $\mathbf{y}_i$ indicates the condition in which observation $\mathbf{x}_i$ was sampled.

Output: Sample-associated relative likelihood $\tilde{\mathbf{Y}}_{norm} \in \mathbb{R}^{n \times d}$ where d is the number of unique conditions in $\mathbf{y}$

1. Build graph $\mathbf{G} = \{V, E\}$ by applying anisotropic or other kernel function on $\mathbf{X}$;
2. Instantiate One-Hot Indicator $\mathbf{Y}$, with one column for each unique condition in $\mathbf{y}$;
3. Column-wise L1-normalize $\mathbf{Y}$ to yield $\mathbf{Y}_{norm}$;
4. Apply manifold heat filter over $(\mathbf{G}, \mathbf{Y}_{norm})$ to calculate $\tilde{\mathbf{Y}}$, the kernel density estimate of the data in each condition, also referred to as the **sample-associated density estimates**;
5. Row-wise L1 normalize $\tilde{\mathbf{Y}}$ to yield $\tilde{\mathbf{Y}}_{norm}$ also referred to as the **sample-associated relative likelihoods**;

3.2.2 Calculating sample-associated density estimates

A popular non-parametric approach to estimating data density is using a kernel density estimate (KDE), which relies on an affinity kernel function. To estimate the density of single-cell samples over a graph, we turn to the heat kernel. This kernel uses diffusion to provide local adaptivity in regions of varying data density [77] such as is observed in single-cell data. Here, we extend this kernel as a low pass filter over a graph to estimate the density of a sample represented by the sample indicator signals defined above. To begin, we take the Gaussian KDE, which is a well known tool for density estimation in $\mathbb{R}^d$. We then generalize this form to smooth manifolds. The full construction of this generalization is described in detail in the **Methods**, and a high level overview is provided here.

A kernel density estimator $\hat{f}(x,t)$ with bandwidth $t > 0$ and kernel function $K(x,y,t)$ is defined as

$$\hat{f}(x,t) = \frac{1}{N}\sum_{i=1}^{N} K(x, X_i, t), \quad x \in \mathcal{X} \tag{3.1}$$

where X is the observed data, x is some point in $\mathcal{X} := \mathbb{R}^d$ (i.e., $\mathcal{X}$ is defined as $\mathbb{R}^d$), and $\mathcal{X}$ is endowed with the Gaussian kernel defined as

$$K(x,y,t) = \frac{1}{(4\pi t)^{d/2}} e^{-\|x-y\|_2^2/4t} \tag{3.2}$$

Thus, **Equation 3.2** defines the Gaussian KDE in $\mathbb{R}^d$. However, this function relies on the Euclidean distance $\|x-y\|_2^2$, which is derived from the kernel space in $\mathbb{R}^d$. Since manifolds are only locally Euclidean, we cannot apply this KDE directly to a general manifold.

To generalize the Gaussian KDE to a manifold we need to define a kernel space (i.e., the range of a kernel operator) over a manifold. In $\mathbb{R}^d$ the kernel space is often defined via infinite weighted sums of sines and cosines, also known as the Fourier series. However, this basis is not well defined for a Riemannian manifold, so we instead use the eigenbasis of the Laplace operator as our kernel basis. The derivation and implication of this extension is formally explored in the **Methods**. The key insight is that using this kernel space, the Gaussian KDE can be defined as a filter constructed from the eigenvectors and eigenvalues of the Laplace operator on a manifold. When this manifold is approximated using a graph, we define this KDE as a graph filter over the graph Laplacian given by the following equation:

$$\hat{f}(x,t) = e^{-t\mathcal{L}}x = \Psi h(\Lambda)\Psi^{-1}x \tag{3.3}$$

where t is the kernel bandwidth, $\mathcal{L}$ is the graph Laplacian, x is the empirical density, Ψ and Λ are the eigenvectors and corresponding eigenvalues of $\mathcal{L}$, and $e^{-t\mathcal{L}}$ is the matrix

exponential. This signal processing formulation can alternatively be formulated as an optimization with Tikhonov Regularization, which seeks to reconstruct the original signal while penalizing differences along edges of the graph. This connection is further explored in the **Methods**.

To achieve an efficient implementation of the filter in **Equation 3.3**, the MELD algorithm considers the spectral representation of the sample indicator signals and uses a Chebyshev polynomial approximation [78] to efficiently compute the sample-associated density estimate (see the **Methods**). The result is a highly scalable implementation. The sample-associated density estimate for two conditions can be calculated on a dataset of 50,000 cells in less than 8 minutes in a free Google Colaboratory notebook[1], with more than 7 minutes of that time spent constructing a graph that can be reused for visualization [11] or imputation [17]. With the sample-associated density estimates, it is now possible to identify the cells that are most and least affected by an experimental perturbation.

3.2.3 Using sample-associated relative likelihood to quantify differences between experimental conditions

Each sample-associated density estimate over the graph indicates the probability of observing each cell within a given experimental sample. For example, in a healthy peripheral blood sample, we would expect high density estimates associated with abundant blood cells such as neutrophils and T cells and low density estimates associated with less abundant cells types, such as basophils and eosinophils. When considering the effect of an experimental perturbation, we are not only interested in these density estimates directly, but we want also to quantify the change in density associated with a change in an experimental variable. For example, one might want to know if a drug treatment causes a change in probability of observing some kinds of blood cells in peripheral blood.

[1]Freely available at `colab.research.google.com`, most instances provide a 4-core 2GHz CPU and 20GB of RAM.

When examining the rows of the sample-associated density estimates for a single-cell, the values represent the likelihood of observing that cell in each experimental condition. To quantify the change in likelihood across conditions, we apply a normalization across the likelihoods for each cell to calculate sample-associated relative likelihoods. These relative likelihoods sum to 1 for each cell and provide a basis for quantifying the change in likelihood of observing a cell in each condition. We then use these relative likelihoods as a basis for identifying cell states that are enriched, depleted, or unaffected by the perturbation.

The sample-associated relative likelihoods can be used to analyze scRNA-seq perturbation studies of varying experimental designs. For cases with only one experimental and one control condition, we typically only refer to the sample-associated relative likelihood of the treatment condition for downstream analysis. For more complicated experiments comprising replicates, we normalize matched experimental and control conditions individually, then average the relative likelihood of the each condition across replicates, as in **Section 3.2.7** and **Section 3.2.8**. With datasets comprising three or more experimental conditions, each sample-associated relative likelihood may be used individually to analyze cells that are enriched, depleted, or unaffected in the corresponding condition, as in **Section 3.2.9**. We expect this flexibility will enable the use of sample-associated density estimates and relative likelihoods across a wide range of single-cell studies.

3.2.4 Vertex-frequency clustering identifies cell populations affected by a perturbation

A common goal for analysis of experimental scRNA-seq data is to identify subpopulations of cells that are responsive to the experimental treatment. Existing methods cluster cells by transcriptome alone and then attempt to quantify the degree to which these clusters are differentially represented in the two conditions. However, this is problematic because the granularity, or sizes, of these clusters may not correspond to the sizes of the cell popu-

lations that respond similarly to experimental treatment. Additionally, when partitioning data along a continuum, cluster boundaries are somewhat arbitrary and may not correspond to populations with distinct differences between conditions. Our goal is to identify clusters that are not only transcriptionally similar but also respond similarly to an experimental perturbation (**Figure 3.2**).

A naïve approach to identify such clusters would be to simply concatenate the sample-associated relative likelihood to the gene expression data as an additional feature and cluster on these combined features. However, the magnitude of the relative likelihood does not give a complete picture of differences in response to a perturbation. For example, even in a two-sample experiment, there are multiple ways for a cell to have a sample-associated relative likelihood of 0.5. In one case, it might be that there is a continuum of cells one end of which is enriched in the treatment condition and the other end is enriched in the control condition. In this case transitional cells halfway through this continuum will have a sample-associated relative likelihood of 0.5 (we show an example of this in **Section 3.2.6**). Another scenario that would result in a relative likelihood of 0.5 is even mixing of a population of cells between control and treatment conditions with no transition, i.e., cells that are part of a non-responsive cell subtype that is unchanged between conditions (we show an example of this in **Section 3.2.8** and **Figure S2**). To differentiate between such scenarios we must consider not only the magnitude of the sample-associated relative likelihood but also the frequency of the input sample indicator signals over the manifold. Indeed in the transitional case the input sample labels change gradually or has *low frequency* over the manifold, and in the even-mixture case it changes frequently between closely connected cells or has *high frequency* over the manifold.

As no contemporary method is suitable for resolving these cases, we developed an algorithm that integrates gene expression, the magnitude of sample-associated relative likelihoods, and the frequency response of the input sample labels over the cellular manifold (**Figure S2**). In particular, we cluster using local frequency profiles of the sample indicator

signal around each cell. This method, which we call *vertex-frequency clustering* (VFC), is an adaptation of the signal-biased spectral clustering proposed by Shuman et al. [52]. The VFC algorithm provides a feature basis for clustering based on the spectrogram [52] of the sample indicator signals, which can be thought of as a histogram of frequency components of graph signals. We observe that we can distinguish between non-responsive populations of cells with high frequency sample indicator signal components and transitional populations with lower frequency indicator signal components. The VFC feature basis combines this frequency information with the magnitude of the sample-associated relative likelihood and the cell similarity graph to identify phenotypically similar populations of cells with uniform response to a perturbation. The algorithm is discussed in further detail in the **Methods**.

With VFC, it is possible to define a new paradigm for recovering the gene signature of a perturbation. In traditional analysis, where clusters are calculating data geometry alone, gene signatures are often calculated using differential expression analysis between experimental conditions within each cluster (**Figure S3a**). The theory of the traditional framework is that these expression differences reflect the change in cell states observed as a result of the perturbation. However, if the cluster contains multiple subpopulations that each contain different responses to the perturbation, we can first separate these populations using VFC and then compare each subpopulation individually (**Figure S3b**). Not only does this allow for more finely resolved comparisons, we show in the following section that this approach is capable of recovering gene signatures more accurately than directly comparing two samples.

We describe a full pipeline for analysis of scRNA-seq datasets with MELD and VFC in **Supplementary Note 1** and **Figure S4**.

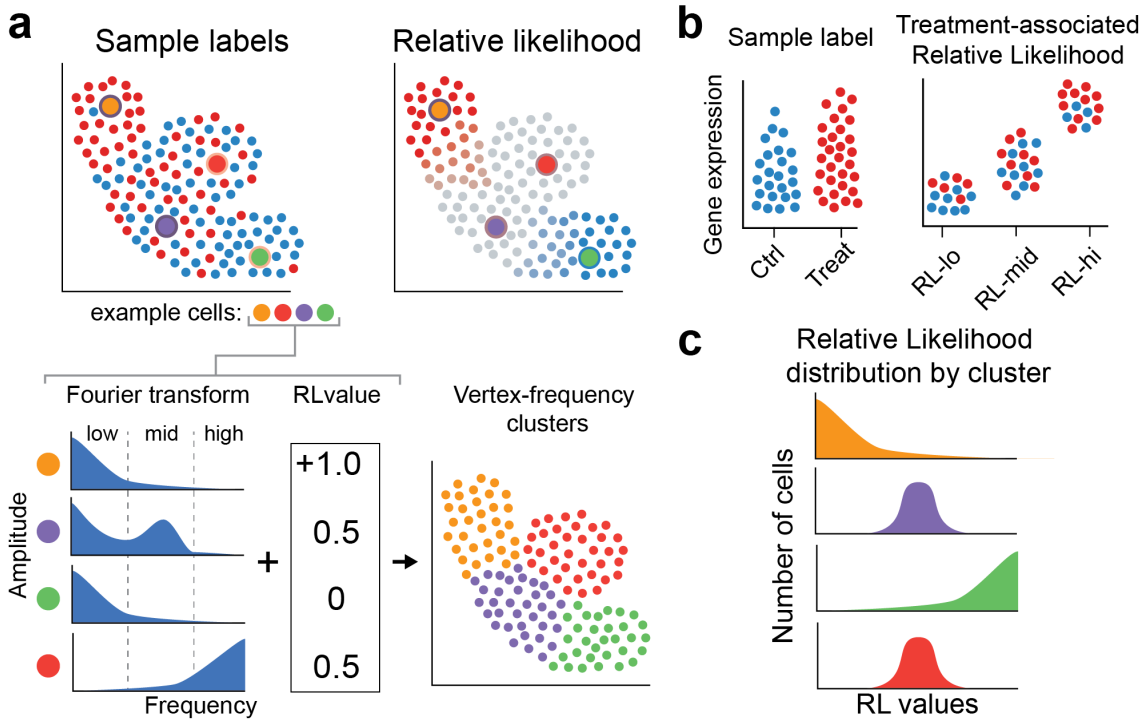

Figure 3.2: Vertex Frequency Analysis using the sample-associated indicator signals and relative likelihood (**a**) The Windowed Graph Fourier Transform of the sample-associated indicator signals and values of sample-associated relative likelihood values at four example points shows distinct patterns between a transitional (blue) and unaffected (red) cell. This information is used in spectral clustering, resulting in Vertex Frequency Clustering. (**b**) Characterizing Vertex Frequency Clusters with the highest and lowest sample-associated relative likelihood values elucidates gene expression changes associated with experimental perturbations. (**c**) Examining the distribution of sample-associated relative likelihood scores in vertex-frequency clusters identifies cell populations most affected by a perturbation.

3.2.5 Quantitative validation of the MELD and VFC algorithms

No previous benchmarks exist to quantify the ability of an algorithm to capture changes in density between scRNA-seq samples. To validate the sample-associated relative likelihood and VFC algorithms, we used a combination of simulated scRNA-seq data and synthetic experiments using previously published datasets. To create simulated scRNA-seq data, we used Splatter [79]. To ensure the algorithms worked on real scRNA-seq datasets, we also used two previously published datasets comprising Jurkat T cells [8] and cells from whole zebrafish embryos [75]. In each dataset, we created a ground truth relative likelihood distribution over all cells that determines the relative likelihood each cell would be observed in one of two simulated conditions. In each simulation, different populations of cells of varying sizes were depleted or enriched. Cells were then randomly split into two samples according to this ground truth relative likelihood and used as input to each algorithm. More detail on the comparison experiments is provided in the **Methods**.

We performed three sets of quantitative comparisons. First, we calculated the degree to which the MELD algorithm captured the ground truth relative likelihood distribution in each simulation. We found that MELD outperformed other graph smoothing algorithms by 10-52% on simulated data and 36-51% on real datasets (**Figure 3.3**, **Table S1**). We also determined that the MELD algorithm is robust to the number of cells captured in the experiment with only a 10% decrease in performance when 65% of the cells in the T cell dataset were removed (**Figure S5**). We used results from these simulations to determine the optimal parameters for the MELD algorithm (**Summplementary Note 3**). Next, we quantified the accuracy of the VFC algorithm to identify clusters of cells that were enriched or depleted in each condition. When compared to six common clustering algorithms including Leiden [27] and CellHarmony [80], VFC was the top performing algorithm on every simulation on the T cell data and best performing on average on the zebrafish dataset with a 57% increase in average performance over Louvain, the next best algorithm (**Figures**

S6a-c & S7, **Table S2**). Finally, we calculated how well VFC clusters could be used to calculate the gene signature of a perturbation. Gene signatures obtained using VFC were compared to signatures obtained using direct comparison of two conditions–the current standard–and those obtained using other clustering algorithms (**Figure S6d**). These results confirm that MELD and VFC outperform existing methods for analyzing multiple scRNA-seq datasets from different experimental conditions.

3.2.6 The sample-associated relative likelihood identifies a biologically relevant signature of T cell activation

To demonstrate the biological relevance of the MELD algorithm, we analyze Jurkat T cells cultured for 10 days with and without anti-CD3/anti-CD28 antibodies as part of a Cas9 knock-out screen published by Datlinger et al. [8] (**Figure 3.4a**). The goal of this experiment was to characterize the transcriptional signature of T cell Receptor (TCR) activation and determine the impact of gene knockouts in the TCR pathway. First, we visualized cells using PHATE, a visualization and dimensionality reduction tool for single-cell RNA-seq data (**Figure 3.4b**) [11]. We observed a large degree of overlap in cell states between the stimulated and control conditions, as noted in the original study [8].

To determine a gene signature of the TCR activation, we considered cells with no CRISPR perturbation. First, we computed sample-associated relative likelihood and VFC clusters on these samples. Then we derived a gene signature by performing differential expression analysis between VFC clusters with the highest and lowest relative likelihood values. We identified 2335 genes with a q-value < 0.05 as measured by a rank sum test with a Benjamini & Hochberg False Discovery Rate correction [81]. We then compared this signature to those obtained using the same methods from our simulation experiments. To determine the biological relevance of these signature genes, we performed gene set enrichment analysis on both gene sets using EnrichR [82]. Considering the GO terms

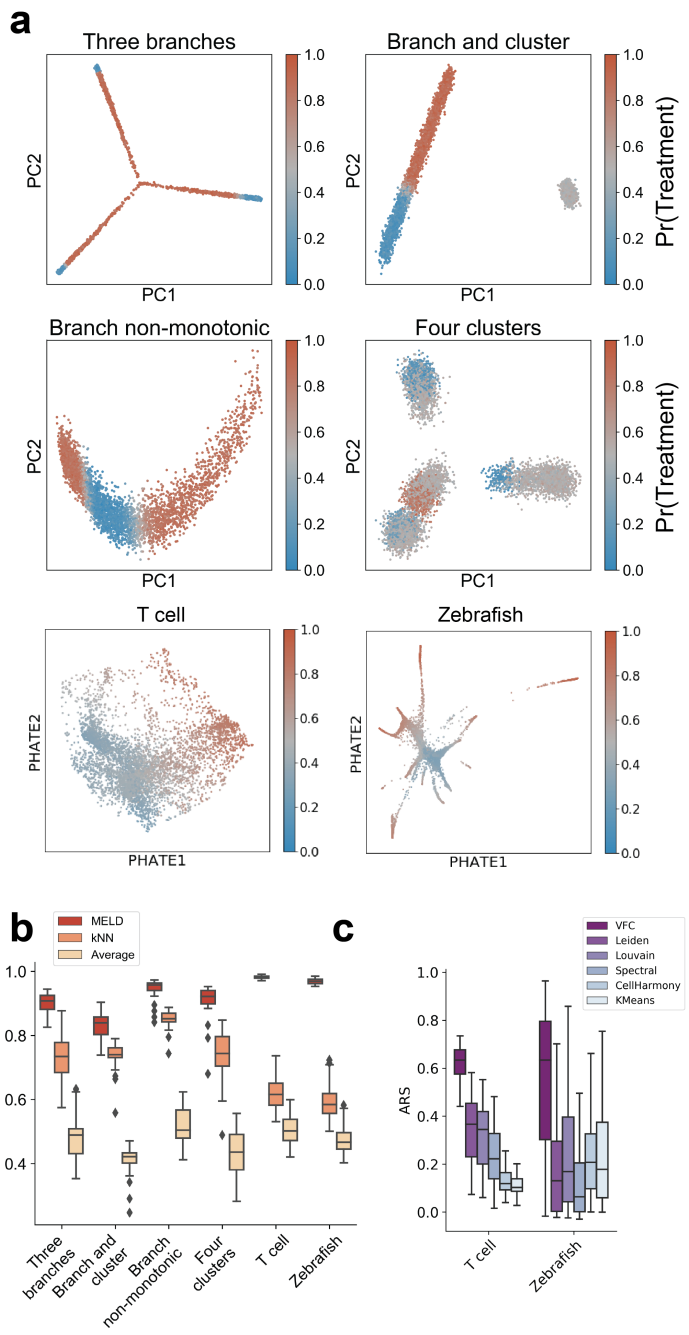

Figure 3.3: Quantitative comparison of the sample-associated relative likelihood and VFC. (**a**) Single-cell datasets were generated using Splatter [79] or taken from previously published experiments [8, 75]. Ground truth sample assignment probabilities with each of two conditions were randomly generated 20 times with varying noise and regions of enrichment for the simulated data and 100 random sample assignments were generated for the real-world datasets. Each cell is colored by the probability of being assigned to the treatment sample. (**b**) Pearson correlation comparison of the sample-associated relative likelihood algorithm to kNN averaging of the sample labels and graph averaging of the sample labels. Higher values are better. (**c**) Comparison of VFC to popular clustering algorithms. Adjusted Rand Score (ARS) quantifies how accurately each method detects regions that were enriched, depleted, or unchanged in the experimental condition relative to the control. Higher values are better.

highlighted by Datlinger et al. [8], we found that the MELD gene list has the highest combined score in all of the gene terms we examined (**Figure 3.4d**). These results show that the sample-associated relative likelihood and VFC are capable of identifying a biologically relevant dimension of T cell activation at the resolution of single-cells. Furthermore, the gene signature identified using the MELD and VFC outperformed standard differential expression analyses to identify the signature of a real-world experimental perturbation.

Finally, to quantitatively rank the impact of each Cas9 gene knockout on TCR activation we examined the distribution of sample-associated relative likelihood values for all stimulated cells transfected with gRNAs targeting a given gene (**Figure S8**). We observed a large variation in the impact of each gene knockout consistent with the published results from Datlinger et al. [8]. Encouragingly, our results agree with the bulk RNA-seq validation experiment of Datlinger et al. [8] showing strongest depletion of TCR response with knockout of kinases LCK and ZAP70 and adaptor protein LAT. We also find a slight increase in relative likelihood of the stimulation condition in cells in which negative regulators of TCR activation are knocked out, including PTPN6, PTPN11, and EGR3. Together, these results show that the MELD and VFC algorithms are suitable for characterizing a biological process such as TCR activation in the context of a complex Cas9 knockout screen.

3.2.7 VFC improves characterization of subpopulation response to *chd* loss-of-function

To demonstrate the utility of sample-associated relative likelihood analysis applied to datasets composed of multiple cell types, we analyzed a chordin loss-of-function experiment in zebrafish using CRISPR/Cas9 (**Figure S9**) [75]. In the experiment published by Wagner et al. [75], zebrafish embryos were injected at the 1-cell stage with Cas9 and gRNAs targeting either chordin (*chd*), a BMP-antagonist required for developmental pat-

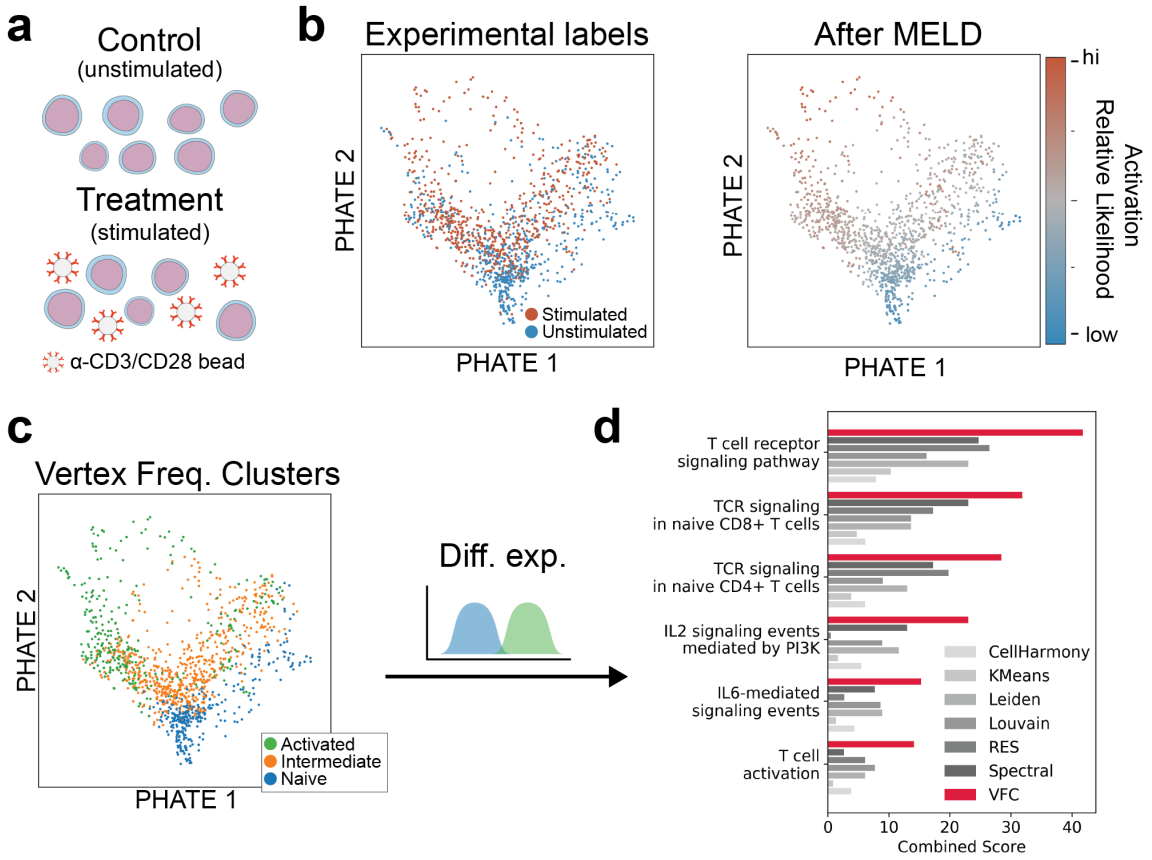

Figure 3.4: MELD recovers signature of TCR activation. (**a**) Jurkat T-cells were stimulated with α-CD3/CD28 coated beads for 10 days before collection for scRNA-seq. (**b**) Examining a PHATE plot, there is a large degree of overlap in cell state between experimental conditions. However, after MELD it is clear which cells states are prototypical of each experimental condition. (**c**) Vertex Frequency Clustering identifies an activated, a naive, and an intermediate population of cells. (**d**) Signature genes identified by comparing the activated to naive cells are enriched for annotations related to TCR activation using EnrichR analysis. Combined scores for the MELD gene signature are shown in red and scores for a gene signature obtained using the sample labels only are shown in grey.

terning, or tyrosinase (*tyr*), a control gene. Embryos were collected for scRNA-seq at 14-16 hours post-fertilization (hpf). We expect incomplete penetrance of the perturbation in this dataset because of the mosaic nature of Cas9 mutagenesis [83].

First, we calculate the sample-associated relative likelihood between the chordin and tyrosinase conditions. Because the experiment was performed in triplicate with three paired *chd* and *tyr* samples, we first calculated the sample-associated density estimates for each of the six samples. We then normalized the density estimated across the paired *chd* and *tyr* conditions. Finally, we averaged the replicate-specific relative likelihoods of the *chd* condition for downstream analysis. We refer to this averaged likelihood simply as the chordin relative likelihood (**Figure S9**).

To characterize the effect of mutagenesis on various cell populations, we first examined the distribution of chordin relative likelihood values across the 28 cell state clusters generated by Wagner et al. [75] (**Figure 3.5b**). We find that overall the most enriched clusters contain mesodermal cells and the most depleted clusters contain dorsally-derived neural cells matching the ventralization phenotype previously reported with *chd* loss-of-function [84–86]. However, we observe that several clusters have a wide range of chordin relative likelihood values suggesting that there are cells in these clusters with different perturbation responses. Using VFC analysis we find that several of these clusters contain biologically distinct subpopulations of cells with divergent responses to *chd* knock out.

An advantage of using MELD and VFC is the ability to characterize the response to the perturbation at the resolution corresponding to the perturbation response (**Figure 3.2c**). We infer that the resolution of the published clusters is too coarse because the distribution of chordin relative likelihood values is very large for several of the clusters. For example the chordin relative likelihoods within the Tailbud – Presomitic Mesoderm (TPM) range from 0.29-0.94 indicating some cells are strongly enriched while others are depleted. To disentangle these effects, we performed VFC subclustering for all clusters using the strategy proposed in **Supplementary Note 1**. We found 12 of the 28 published

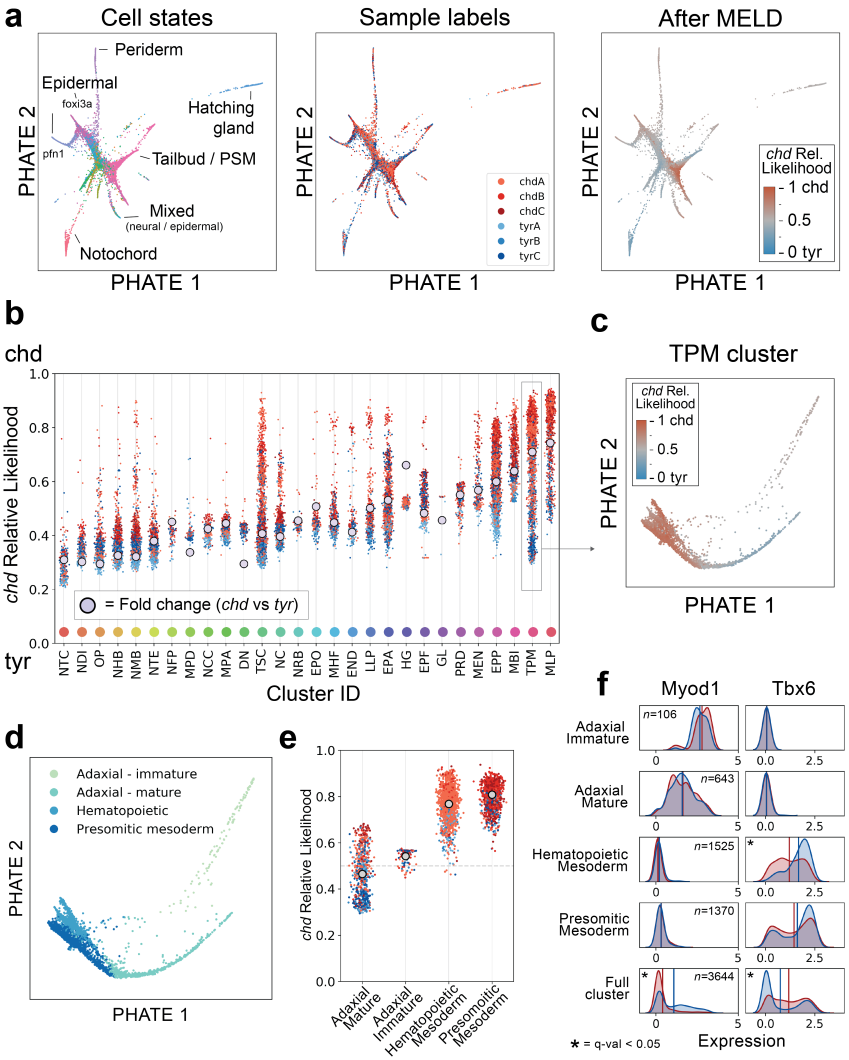

Figure 3.5: Characterizing chordin Cas9 mutagenesis with MELD. (**a**) PHATE shows a high degree of overlap of sample labels across cell types. Applying MELD to the mutagenesis vector reveals regions of cell states enriched in the *chd* or *tyr* conditions. (**b**) Using published cluster assignments[2], we show that the *chd*-associated relative likelihood quantifies the effect of the experimental perturbation on each cell, providing more information than calculating fold-change in the number of cells between conditions in each cluster (grey dot), as was done in the published analysis. Color of each point corresponds to the sample labels in panel (a). Generally, average relative likelihood within each cluster aligns with the fold-change metric. However, we can identify clusters, such as the TPM or TSC, with large ranges of relative likelihoods indicating non-uniform response to the perturbation. (**c**) Visualizing the TPM cluster using PHATE, we observe several cell states with mostly non-overlapping relative likelihood values. (**d**) Vertex Frequency Clustering identifies four cell types in the TPM. (**e**) We see the range of relative likelihood values in the TPM cluster is due to subpopulations with divergent responses to the *chd* perturbation. (**f**) We observe that changes in gene expression between the *tyr* (blue) and *chd* (red) conditions is driven mostly by changes in abundance of subpopulations with the TPM cluster.

clusters warranted further subclustering with VFC resulting in a total of 50 final cluster labels (**Figure S10j**). To determine the biological relevance of the VFC clusters, we manually annotated each of the three largest clusters subdivided by VFC revealing previously unreported effects of *chd* loss-of-function within this dataset. A full exploration can be found in **Supplementary Note 2** with the results of TPM cluster shown in **Figure 3.5c-f**.

3.2.8 Identifying the effect of IFNγ stimulation on pancreatic islet cells

To determine the ability of the MELD and VFC to uncover biological insights, we generated and characterized a dataset of human pancreatic islet cells cultured for 24 hours with and without interferon-gamma (IFNγ), a system with significant clinical relevance to auto-immune diseases of the pancreas such as Type I Diabetes mellitus (T1D) and islet allograft rejection [87]. Previous studies have characterized the effect of these cytokines on pancreatic beta cells using bulk RNA-sequencing[88], but no studies have addressed this system at single-cell resolution.

To better understand the effect of immune cytokines on islet cells, we cultured islet cells from three donors for 24 hours with and without IFNγ and collected cells for scRNA-seq. After filtering, we obtained 5,708 cells for further analysis. Examining the expression of marker genes for major cell types of the pancreas, we observed a noticeable batch effect associated with the donor ID, driven by the maximum expression of glucagon, insulin, and somatostatin in alpha, beta, and delta cells respectively (**Figure S11a**). To correct for this difference while preserving the relevant differences between donors, we applied the MNN kernel correction described in the **Methods**. Note, here we are applying the MNN

[2]Abbreviations: MLP: Lateral plate, TPM: Tailbud - Presomitic mesoderm, HG: Hatching gland, MBI: Blood island, EPP: Epidermal - pfn1, MEN: Endothelial, PRD: Periderm, EPA: Epidermal anterior, EPO: Otic placode, LLP: Lateral line, EPF: Epidermal - foxi3a, GL: Germline, NRB: Rohon beard, NFP: Floorplate, MHF: Heart field, MPA: Pharyngeal arch, NCC: Neural crest - crestin, END: Endoderm, TSC: Tailbud - spinal cord, NC: Neural crest, NTE: Telencephalon, MPD: Pronephric duct, NHB: Hindbrain, NMB: Midbrain, NTC: Notocord, NDI: Diencephalon, DN: Neurons, OP: Optic

correction is only applied across donors, not across the IFNg treatment. We developed guidelines for applying batch correction prior to running MELD in **Supplementary Note 3**.

To quantify the effect of IFNγ treatment across these cell types, we calculated the sample-associated relative likelihood of IFNγ stimulation using the same strategy to handle matched replicates as was done for the zebrafish data (**Figure 3.6a**). We then used established marker genes of islet cells [89] to identify three major populations of cells corresponding to alpha, beta, and delta cells (**Figures 3.6a-b & S11b**). We next applied VFC to each of the three endocrine cell types and identified a total of nine clusters. Notably, we found two clusters of beta cells with intermediate IFNg relative likelihood values. These clusters are cleanly separated on the PHATE plot of all islet cells (**Figure 3.6a**) and together the beta cells represent the largest range of IFNg relative likelihood scores in the dataset.

To further inspect these beta cell clusters, we consider a separate PHATE plot of the cells in the four beta cell clusters (**Figure 3.6e**). Examining the distribution of input sample signals values in these intermediate cell types, we find that one cluster, which we label as *Non-responsive*, exhibits high frequency input sample signals indicative of a population of cells that does not respond to an experimental treatment. The *Responsive - Mid* cluster matches our characterization of a transitional population with a structured distribution of input sample signals. Supporting this characterization, we find a lack of upregulation in IFNγ-regulated genes such as STAT1 in the non-responsive cluster, similar to the cluster of beta cells with the lowest IFNg relative likelihood values (**Figure 3.6f**).

In order to understand the difference between the non-responsive beta cells and the responsive populations, we calculated differential expression of genes in the non-responsive clusters and all others. The gene with the greatest difference in expression was insulin, the major hormone produced by beta cells, which is approximately 2.5-fold increased in the non-responsive cells (**Figure 3.6f**). This cluster of cells bears resemblance to a recently

described “extreme” population of beta cells that exhibit elevated insulin mRNA levels and are found to be more abundant in diabetic mice[90, 66]. That these cells appear non-responsive to IFNγ stimulation and exhibit extreme expression of insulin suggests that the presence of extreme high insulin in a beta cell prior to IFNγ exposure may inhibit the IFNγ response pathway through an unknown mechanism.

We next characterized the gene expression signature of IFNγ treatment across all three endocrine cell types (**Figure 3.6c-d**). Using a rank sum test to identify genes that change the most between the clusters with highest and lowest IFNg relative likelihood values within each endocrine population, we identify 911 genes differentially expressed in all three cell types. This consensus signature includes activation of genes in the JAK-STAT pathway including STAT1 and IRF1 [91] and in the IFN-mediated antiviral response including MX1, OAS3, ISG20, and RSAD2 [92–94]. The activation of both of these pathways has been previously reported in beta cells in response to IFNγ [95, 96]. To confirm the validity of our gene signatures, we use EnrichR [82] to perform gene set enrichment analysis on the signature genes and find strong enrichment for terms associated with interferon signalling pathways (**Figure S11d**). From these results we conclude that although IFNγ leads to upregulation of the canonical signalling pathways in all three cell types, the response to stimulation in delta cells is subtly different to that of alpha or beta cells.

Here, we applied MELD analysis to identify the signature of IFNγ stimulation across alpha, beta, and delta cells, and we identified a population of beta cells with high insulin expression that appears unaffected by IFNγ stimulation. Together, these results demonstrate the utility of MELD analysis to reveal biological insights in a clinically-relevant biological experiment.

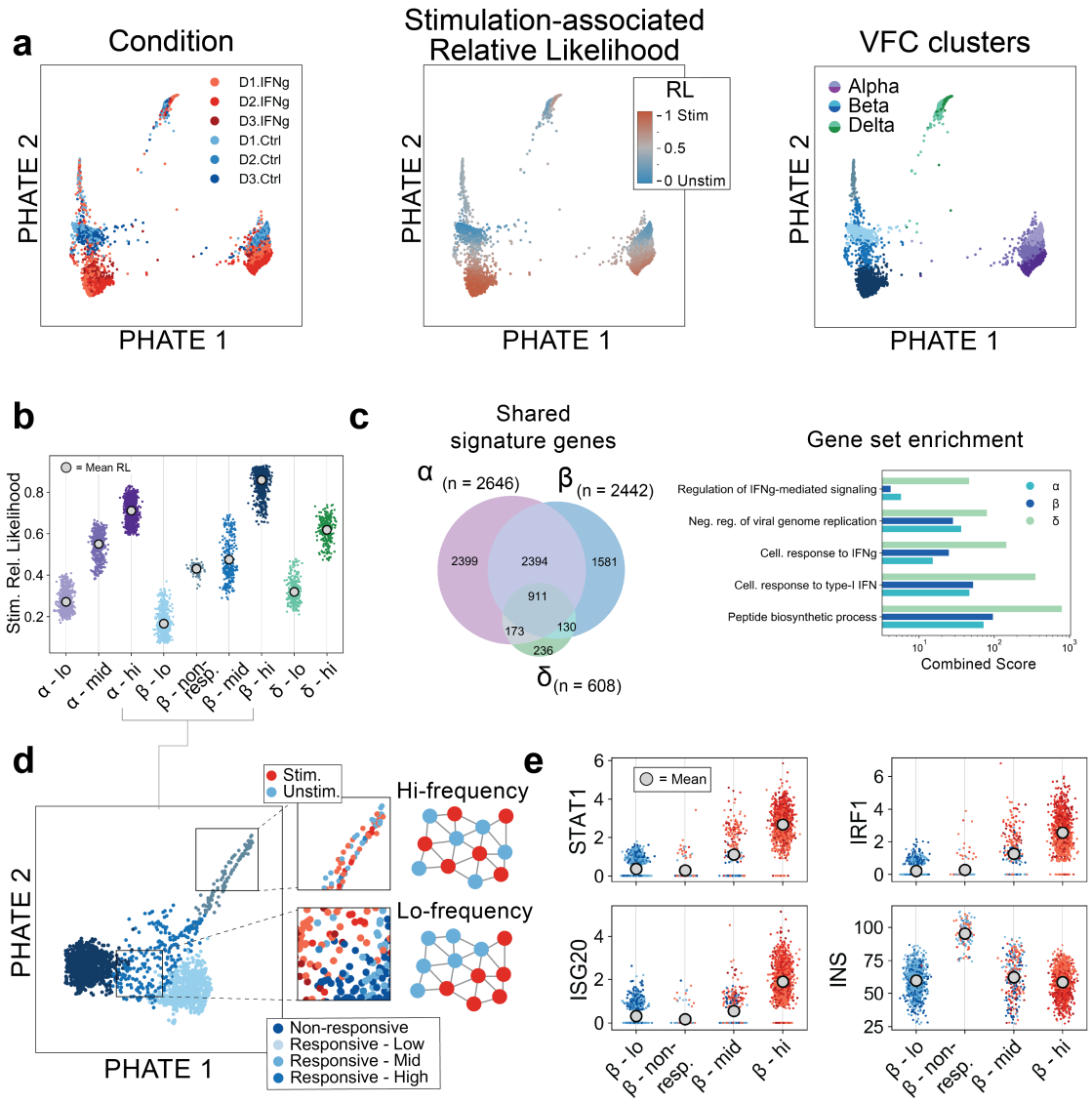

Figure 3.6: MELD characterizes the response to IFNγ in pancreatic islet cells. (**a**) PHATE visualization of pancreatic islet cells cultured for 24 hours with or without IFNγ. Vertex-frequency clustering identifies nine clusters corresponding to alpha, beta, and delta cells. (**b**) Examining the stimulation-associated relative likelihood (RL) in each cluster, we observe that beta cells have a wider range of responses than alpha or delta cells. (**c**) We identify the signature of IFNγ stimulation by calculating differential expression between the VFC clusters with the highest and lowest stimulation likelihood values for each cell type. We find a high degree of overlap of the significantly differentially expressed genes between alpha and beta cells. (**d**) Results of gene set enrichment analysis for signature genes in each cell type. Beta cells have the strongest enrichment for IFN response pathway genes. (**e**) Examining the four beta cell clusters more closely, we observe two populations with intermediate relative likelihood values. These populations are differentiated by the structure of the sample label in each cluster (outset). In the non-responsive cluster, the sample label has very high frequency unlike the low frequency pattern in the transitional Responsive - mid cluster. (**f**) We find that the non-responsive cluster has low expression of IFNγ-regulated genes such as STAT1 despite containing roughly equal numbers of unstimulated and stimulated cells. This cluster is marked by approximately 40 percent higher expression of insulin.

3.2.9 Analysis of donor-specific composition

Although most of the analysis here focuses on two condition experiments, we show that it is possible to use the sample-associated relative likelihood to quantify the differences between more than two conditions. In the islet dataset, we have samples of treatment and control scRNA-seq data from three different donors. To quantify the differences in cell profiles between donors, we first create a one-hot vector for each donor label and normalize across all three smoothed vectors. This produces a measure of how likely each transcriptional profile is to be observed in donor 1, 2, or 3. We then analyze each of these signals for each cluster examined in **Section 3.2.8** (**Figure S12**). We find that all of the alpha cell and delta cell clusters are depleted in donor 3 and the non-responsive beta cell cluster is enriched primarily in donor 1. Furthermore, the most highly activated alpha cell cluster is enriched in donor 2. As with the sample-associated relative likelihood derived for the IFNγ response, it is also possible to identify donor-specific changes in gene expression, or clusters of cells differentially abundant between each donor. We propose that this strategy could be used to extend MELD analysis to experiments with multiple categorical experimental conditions, such as data collected from different tissues or stimulus conditions.

3.3 Discussion

When performing multiple scRNA-seq experiments in various experimental and control conditions, researchers often seek to characterize the cell types or sets of genes that change from one condition to another. However, quantifying these differences is challenging due to the subtlety of most biological effects relative to the biological and technical noise inherent to single-cell data. To overcome this hurdle, we designed the MELD and VFC algorithms to quantify compositional differences between samples. The key innovation in

the sample-associated relative likelihood algorithm is quantifying the effect of a perturbation at the resolution of single-cells using theory from manifold learning.

We have shown that our analysis framework improves over the current best-practice of clustering cells based on gene expression and calculating differential abundance and differential expression within clusters. Clustering prior to quantifying compositional differences can fail to identify the divergent responses of subpopulations of cells within a cluster. Using the sample labels and sample-associated relative likelihood, we apply VFC to derive clusters of cells to identify cells that are most enriched in either condition and cells that are unaffected by an experimental perturbation. We show that gene signatures extracted using these clusters outperform those derived from direct comparison of two samples.

We demonstrated the application of MELD analysis on single-cell datasets from three different biological systems and experimental designs. We provided a framework for handling paired experimental and control replicates and guidance on analysis of complex experimental designs with more than two conditions and in the context of a single-cell Cas9 knockout screen. In our analysis of the zebrafish dataset, we showed the published clusters contained biologically relevant subpopulations of cells with divergent responses to the experimental perturbation. We also described a previously unpublished dataset of pancreatic islet cells stimulated with IFN-γ and characterize a previously unreported subpopulation of β cells that appeared unresponsive to stimulation. We related this to emerging research describing aβcells subtype marked by high insulin mRNA expression and unique biological responses.

We anticipate MELD to have widespread use in many contexts since experimental labels can arise in many contexts. As we showed, if we have sets of single-cell data from healthy individuals vs sick individuals, the sample-associated relative likelihood could indicate cell types specific to disease. This framework could potentially be extended to patient level measurements where patients' phenotypes as measured with clinical vari-

ables and laboratory values can be associated with enriched states in disease or treatment conditions. Indeed MELD has already seen use in several contexts [97–101].

Acknowledgements

The authors would like to thank C. Vejnar, R. Coifman, J. Noonan, V. Tornini, and C. Kontur for fruitful discussions. We would like to thank Guilin Wang of the Yale Center for Genome Analysis for help in preparing the pancreatic islet data. This research was supported in part by: the Eunice Kennedy Shriver National Institute of Child Health & Human Development of the NIH (Award Number: F31HD097958) *[D.B.]*; the Gruber Foundation *[S.G.]*; IVADO Professor startup & operational funds, IVADO Fundamental Research Project grant PRF-2019-3583139727 [*G.W.*]; NIH grants R01GM135929 & R01GM130847 [*G.W., S.K.*]; and Chan-Zuckerberg Initiative grants 182702 & CZF2019-002440 [*S.K.*]. The content provided here is solely the responsibility of the authors and does not necessarily represent the official views of the funding agencies.

3.4 Author Contributions

D.B.B., S.K., G.W., D.v.D., and A.J.G. envisioned the project. D.B.B., J.S. A.T. S.K., and G.W. developed the mathematical formulation of the problem and related numerical analysis. D.B.B, J.S., and S.G. implemented the code. D.B.B. and S.K. performed the analysis of biological and simulated data. A.L.P. and K.C.H. generated and assisted with the analysis of the pancreatic islet dataset. A.J.G. assisted with the analysis of the zebrafish data and related writing. D.B.B., J.S., A.T., S.K., and G.W. wrote the paper. S.G. assisted with the writing.

3.5 Competing Interests

The authors declare the following competing interests: S.K. is a paid scientific advisor to AI Therapeutics (Guilford, CT).

3.6 Methods

In this section, we will provide details about our computational methods for computing the sample-associated density estimate and relative likelihood, as well as extracting information from the sample label and sample-associated relative likelihood by way of a method we call *vertex frequency clustering*. We will outline the mathematical foundations for each algorithm, explain how they relate to previous works in manifold learning and graph signal processing, and provide details of the implementations of each algorithm.

3.6.1 Computation of the sample-associated density estimate

Computing the sample-associated density estimate and relative likelihood involves the following steps each of which we will describe in detail.

1. A cell similarity graph is built over the combined data from all samples where each node or vertex in the graph is a cell and edges in the graph connect cells with similar gene expression values.

2. The sample label for each cell is used to create the sample-associated indicator signal.

3. Each indicator signal is then smoothed over the graph to estimate the density of each sample using the manifold heat filter.

4. Sample-associated density estimates for paired treatment and control samples are normalized to calculate the sample-associated relative likelihood.

Graph construction

The first step in the MELD algorithm is to create a cell similarity graph. In single-cell RNA sequencing, each cell is measured as a vector of gene expression counts measured as unique molecules of mRNA. Following best practices for scRNA-seq analysis [68], we normalize these counts by the total number of Unique Molecular Indicators (UMIs) per cell to give relative abundance of each gene and apply a square-root transform. Next we compute the similarity all pairs of cells, by using their Euclidean distances as an input to a kernel function. More formally, we compute a similarity matrix W such that each entry W_{ij} encodes the similarity between cell gene expression vectors $\mathbf{x_i}$ and $\mathbf{x_j}$ from the dataset X.

In our implementation we use α-decaying kernel proposed by Moon et al. [11] because in practice it provides an effective graph construction for scRNA-seq analysis. However, in cases where batch, density, and technical artifacts confound graph construction, we also use a mutual nearest neighbor kernel as proposed by Haghverdi et al. [102].

The α-decaying kernel [11] is defined as

$$K_{k,\alpha}(x,y)=\tfrac{1}{2}\exp\Big(-\Big(\tfrac{\|x-y\|_2}{\varepsilon_k(x)}\Big)^{\alpha}\Big)+\tfrac{1}{2}\exp\Big(-\Big(\tfrac{\|x-y\|_2}{\varepsilon_k(y)}\Big)^{\alpha}\Big), \tag{3.4}$$

where x, y are data points, $\varepsilon_k(x), \varepsilon_k(y)$ are the distance from x, y to their k-th nearest neighbors, respectively, and α is a parameter that controls the decay rate (i.e., heaviness of the tails) of the kernel. This construction generalizes the popular Gaussian kernel, which is typically used in manifold learning, but also has some disadvantages alleviated by the α-decaying kernel, as explained in Moon et al. [11].

The similarity matrix effectively defines a weighted and fully connected graph between

cells such that every two cells are connected and that the connection between cells x and y is given by $K(x, y)$. To allow for computational efficiency, we sparsify the graph by setting very small edge weights to 0.

While the kernel in **Equation 3.4** provides an effective way of capturing neighborhood structure in data, it is susceptible to batch effects. For example, when data is collected from multiple patients, subjects, or environments (generally referred to as "batches"), such batch effects can cause affinities within each batch are often much higher than between batches, thus artificially creating separation between them rather than follow the underlying biological state. To alleviate such effects, we adjust the kernel construction using an approach inspired by recent work from by Haghverdi et al. [102] on the Mutual Nearest Neighbors (MNN) kernel. We extend the standard MNN approach, which has previous been applied to the k-Nearest Neighbors kernel, to the α-decay kernel as follows. First, within each batch, the affinities are computed using **Equation 3.4**. Then, across batches, we compute slightly modified affinities as

$$K'_{k,\alpha}(x,y)=\min\left\{\exp\left(-\left(\frac{\|x-y\|_2}{\varepsilon'_k(x)}\right)^{\alpha}\right),\exp\left(-\left(\frac{\|x-y\|_2}{\varepsilon'_k(y)}\right)^{\alpha}\right)\right\},$$

where $\varepsilon'_k(x)$ are now computed via the k-th nearest neighbor of x in the batch containing y (and vice versa for $\varepsilon'_k(y)$). Next, a rescaling factor γ_{xy} is computed such that

$$\sum_{z\in\text{batch}(y)} \gamma_{xy} K'_{k,\alpha}(x,z) \leq \beta \sum_{z\in\text{batch}(x)} K_{k,\alpha}(x,z)$$

for every x and y, where $\beta > 0$ is a user configurable parameter. This factor gives rise to the rescaled kernel

$$K'_{k,\alpha,\beta}(x,y) = \begin{cases} K'_{k,\alpha}(x,y) & \text{if batch}(x) = \text{batch}(y) \\ \gamma_{xy} K'_{k,\alpha}(x,y) & \text{otherwise.} \end{cases}$$

Finally, the full symmetric kernel is then computed as

$$K'_{k,\alpha}(x,y) = K'_{k,\alpha}(y,x) = \min\left\{K'_{k,\alpha,\beta}(x,y), K'_{k,\alpha,\beta}(y,x)\right\},$$

and used to set the weight matrix for the constructed graph over the data. Note that this construction is a well-defined extension of (**Equation 3.4**), as it reduces back to that kernel when only a single batch exists in the data.

We also perform an anisotropic density normalization transformation so that the kernel reflects the underlying geometry normalized by density as in Coifman and Lafon [41]. The density normalized kernel $K^q_{k,\alpha}$ divides out by density, estimated by the sum of outgoing edge weights for each node is as follows,

$$K^q_{k,\alpha} = \frac{K'_{k,\alpha}(x,y)}{q(x)q(y)},$$

where

$$q(x) = \int_X K'_{k,\alpha} q(y) dy.$$

We use this density normalized kernel in all experiments. When the data is uniformly sampled from the manifold then the density around each point is constant then this normalization has no effect. When the density is non-uniformly sampled from the manifold this allows an estimation of the underlying geometry unbiased by density. This is especially important when performing density estimation from empirical distributions with different underlying densities. By normalizing by density, we allow for construction of the manifold geometry from multiple differently distributed samples and individual density estimation for each of these densities on the same support. This normalization is further discussed in **Section 3.6.1**.

Estimating sample-associated density and relative likelihood on a graph

Density estimation is difficult in high dimensions because the number of samples needed to accurately reconstruct density with bounded error is exponential in the number of dimensions. Since general high dimensional density estimation is an intrinsically difficult problem, additional assumptions must be made. A common assumption is that the data exists on a manifold of low intrinsic dimensionality in ambient space. Under this assumption a number of works on graphs have addressed density estimation limited to the support of the graph nodes [103–107]. Instead of estimating kernel density or histograms in D dimensions where D could be large, these methods rendered the data as a graph, and density is estimated each point on the graph (each data point) as some variant counting the number of points which lie within a radius of each point on the graph.

The MELD algorithm also estimates density of a signal on a graph. We use a generalization of the standard heat kernel on the graph to estimate density (See **Section 3.6.1**). We draw analogs between the resulting sample-associated density estimate and Gaussian kernel density estimation on the manifold showing our density estimate with a specific parameter set is equivalent to the Gaussian density estimate on the graph (See **Section 3.6.1**).

Graph Signal Processing

The MELD algorithm leverages recent advances in graph signal processing (GSP) [56], which aim to extend traditional signal processing tools from the spatiotemporal domain to the graph domain. Such extensions include, for example, wavelet transforms [57], windowed Fourier transforms [52], and uncertainty principles [108]. All of these extensions rely heavily on the fundamental analogy between classical Fourier transform and graph Fourier transform (described in the next section) derived from eigenfunctions of the graph Laplacian, which is defined as

$$\mathcal{L} := D - W, \tag{3.5}$$

where D is the *degree* matrix, which is a diagonal matrix with $D_{ii} = d(i) = \sum_j W_{ij}$ containing the degrees of the vertices of the graph defined by W.

The Graph Fourier Transform

One of the fundamental tools in traditional signal processing is the Fourier transform, which extracts the frequency content of spatiotemporal signals [109]. Frequency information enables various insights into important characteristics of analyzed signals, such as pitch in audio signals or edges and textures in images. Common to all of these is the relation between frequency and notions of *smoothness*. Intuitively, a function is *smooth* if one is unlikely to encounter a dramatic change in value across neighboring points. A simple way to imagine this is to look at the *zero-crossings* of a function. Consider, for example, sine waves $\sin ax$ of various frequencies $a = 2^k, k \in \mathbb{N}$. For $k = 0$, the wave crosses the x-axis (a zero-crossing) when $x = \pi$. When we double the frequency at $k = 1$, our wave is now twice as likely to cross the zero and is thus less smooth than $k = 0$. This simple zero-crossing intuition for smoothness is relatively powerful, as we will see shortly.

Next, we show that our notions of smoothness and frequency are readily applicable to data that is not regularly structured, such as single-cell data. The graph Laplacian $\mathcal{L}$ can be considered as a graph analog of the Laplace (second derivative) operator ∇^2 from multivariate calculus. This relation can be verified by deriving the graph Laplacian from first principles.

For a graph $\mathcal{G}$ on N vertices, its graph Laplacian $\mathcal{L}$ and an arbitrary graph signal

$\mathbf{f} \in \mathbb{R}^N$, we use **Equation 3.5** to write

$$\begin{aligned}(\mathcal{L}\,\mathbf{f})\,(i) &= ([D - W]\,\mathbf{f})\,(i) \\ &= d(i)\mathbf{f}(i) - \sum_j W_{ij}\mathbf{f}(j) \\ &= \sum_j W_{ij}\,(\mathbf{f}(i) - \mathbf{f}(j))\,. \end{aligned} \tag{3.6}$$

As the graph Laplacian is a weighted sum of differences of a function around a vertex, we may interpret it analogously to its continuous counterpart as the curvature of a graph signal. Another common interpretation made explicit by the derivation in **Equation 3.6** is that $(\mathcal{L}\mathbf{f})(i)$ measures the *local variation* of a function at vertex i.

Local variation naturally leads to the notion of *total variation*,

$$\mathbf{TV}(\mathbf{f}) = \sum_{i,j} W_{ij}(\mathbf{f}(i) - \mathbf{f}(j))^2,$$

which is effectively a sum of all local variations. $\mathbf{TV}(\mathbf{f})$ describes the global smoothness of the graph signal $\mathbf{f}$. In this setting, the more smooth a function is, the lower the value of the variation. This quantity is more fundamentally known as the *Laplacian quadratic form*,

$$\mathbf{f}^T\mathcal{L}\,\mathbf{f} = \sum_{i,j} W_{ij}(\mathbf{f}(i) - \mathbf{f}(j))^2. \tag{3.7}$$

Thus, the graph Laplacian can be used as an operator and in a quadratic form to measure the smoothness of a function defined over a graph. One effective tool for analyzing such operators is to examine their eigensystems. In our case, we consider the eigendecomposition $\mathcal{L} = \Psi\Lambda\Psi^{-1}$, with eigenvalues[3] $\Lambda := \{0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_N\}$ and

[3]Note that in this discussion we abuse notation by treating Λ as an ordered set of Laplacian eigenvalues and as the diagonal matrix with entries from the elements of this set. Similarly, Ψ is both the set of column eigenvectors $\{\psi_i\}_{i=1}^N$ as well as the $N \times N$ matrix $[\psi_1\psi_2\cdots\psi_N]$ with eigenvector as a column.

corresponding eigenvectors $\Psi := \{\psi_i\}_{i=1}^N$. As the Laplacian is a square, symmetric matrix, the spectral theorem tells us that its eigenvectors in Ψ form an orthonormal basis for $\mathbb{R}^N$. Furthermore, the Courant-Fischer theorem establishes that the eigenvalues in Λ are local minima of $\mathbf{f}^T\mathcal{L}\mathbf{f}$ when $\mathbf{f}^T\mathbf{f} = 1$ and $\mathbf{f} \in U$ as $\dim(U) = i = 1, 2, \ldots, N$. At each eigenvalue λ_i this function has $\mathbf{f} = \psi_i$. In summary, the eigenvectors of the graph Laplacian (1) are an orthonormal basis and (2) minimize the Laplacian quadratic form for a given dimension.

Henceforth, we use the term *graph Fourier basis* interchangeably with graph Laplacian eigenvectors, as this basis can be thought of as an extension of the classical Fourier modes to irregular domains [56]. In particular, the ring graph eigenbasis is composed of sinusoidal eigenvectors, as they converge to discrete Fourier modes in one dimension. The graph Fourier basis thus allows one to define the *graph Fourier transform* (GFT) by direct analogy to the classical Fourier transform.

The GFT of a signal f is given by $\hat{f}(\lambda_\ell) = \sum_i f(i)\psi_\ell^T(i) = \langle \mathbf{f}, \psi_\ell \rangle$, which can also be written as the matrix-vector product

$$\hat{\mathbf{f}} = \Psi^T\mathbf{f}. \tag{3.8}$$

As this transformation is unitary, the inverse graph Fourier transform (IGFT) is $\mathbf{f} = \Psi\hat{\mathbf{f}}$. Although the graph setting presents a new set of challenges for signal processing, many classical signal processing notions such as filterbanks and wavelets have been extended to graphs using the GFT. We use the GFT to process, analyze, and cluster experimental signals from single-cell data using a novel graph filter construction and a new harmonic clustering method.

The manifold heat filter

In the MELD algorithm, we seek to estimate the change in sample density between experimental labels along a manifold represented by a cell similarity graph. To estimate sample density along the graph, we employ a novel graph filter construction, which we explain in the following sections. To begin, we review the notion of filtering with focus on graphs and demonstrate manifold heat filter in a low-pass setting. Next, we demonstrate the expanded version of the manifold heat filter and provide an analysis of its parameters. Finally, we provide a simple solution to the manifold heat filter that allows fast computation.

Filters on graphs

Filters can be thought of as devices that alter the spectrum of their input. Filters can be used as bases, as is the case with wavelets, and they can be used to directly manipulate signals by changing the frequency response of the filter. For example, many audio devices contain an equalizer that allows one to change the amplitude of bass and treble frequencies. Simple equalizers can be built simply by using a set of filters called a filterbank. In the MELD algorithm, we use a tunable filter to estimate density of a sample indicator signal on a single-cell graph.

Mathematically, graph filters work analogously to classical filters. Specifically, a filter takes in a signal and attenuates it according to a frequency response function. This function accepts frequencies and returns a response coefficient. This is then multiplied by the input Fourier coefficient at the corresponding frequency. The entire filter operation is thus a reweighting of the input Fourier coefficients. In low-pass filters, the function only preserves frequency components below a threshold. Conversely, high-pass filters work by removing frequencies below a threshold. Bandpass filters transfer frequency components that are within a certain range of a central frequency. The tunable filter in the MELD algorithm is capable of producing any of these responses.

As graph harmonics are defined on the set Λ, it is common to define them as functions of the form $h : [0, \max(\Lambda)] \mapsto [0, 1]$. For example, a low pass filter with cutoff at λ_k would have $h(x) > 0$ for $x < \lambda_k$ and $h(x) = 0$ otherwise. By abuse of notation, we will refer to the diagonal matrix with the filter h applied to each Laplacian eigenvalue as $h(\Lambda)$, though h is not a set-valued or matrix-valued function. Filtering a signal $\mathbf{f}$ is clearest in the spectral domain, where one simply takes the multiplication $\hat{\mathbf{f}}_{\text{filt}} = h(\Lambda)\hat{\mathbf{f}} = h(\Lambda)\Psi^T\mathbf{f}$.

Finally, it is worth using the above definitions to define a vertex-valued operator to perform filtering. As a graph filter is merely a reweighting of the graph Fourier basis, one can construct the *filter matrix*,

$$H = \Psi h(\Lambda)\Psi^T. \tag{3.9}$$

A manipulation using **Equation 3.8** will verify that $H\mathbf{f}$ is the WGFT of $\hat{\mathbf{f}}_{\text{filt}}$. This filter matrix will be used to solve the manifold heat filter in approximate form for computational efficiency.

Laplacian Regularization

A simple assumption for density estimation is *smoothness*. In this model the density estimate is assumed to have a low amount of neighbor to neighbor variation. *Laplacian regularization* [110–118] is a simple technique that targets signal smoothness via the optimization

$$\mathbf{y} = \underset{\mathbf{z}}{\operatorname{argmin}} \underbrace{\|\mathbf{x} - \mathbf{z}\|_2^2}_{\text{a}} + \underbrace{\beta \mathbf{z}^T \mathcal{L} \mathbf{z}}_{\text{b}}. \tag{3.10}$$

Note that this optimization has two terms. The first term (a), called a *reconstruction penalty*, aims to keep the density estimate similar to the input sample information. The second term (b) ensures smoothness of the signal. Balancing these terms adjusts the amount

of smoothness performed by the filter.

Laplacian regularization is a sub-problem of the manifold heat filter that we will discuss for low-pass filtering. In the above, a reconstruction penalty (a) is considered alongside the Laplacian quadratic form (b), which is weighted by the parameter β. The Laplacian quadratic form may also be considered as the norm of the *graph gradient*, i.e.

$$\beta \mathbf{z}^T \mathcal{L} \mathbf{z} = \beta \|\nabla_G \mathbf{z}\|_2^2.$$

Thus one may view Laplacian regularization as a minimization of the edge-derivatives of a function while preserving a reconstruction. Because of this form, this technique has been cast as *Tikhonov regularization* [112, 119], which is a common regularization to enforce a low-pass filter to solve inverse problems in regression. In our results we demonstrate a manifold heat filter that may be reduced to Laplacian regularization using a squared Laplacian.

In **Section 3.6.1** we introduced filters as functions defined over the Laplacian eigenvalues ($h(\Lambda)$) or as vertex operators in **Equation 3.9**. Minimizing optimization **Equation 3.10** reveals a similar form for Laplacian regularization. Although Laplacian regularization filter is presented as an optimization, it also has a closed form solution. We derive this solution here as it is a useful building block for understanding the sample-associate density estimate. To begin,

$$\begin{aligned}
\mathbf{y} &= \underset{\mathbf{z}}{\operatorname{argmin}} \ \|\mathbf{x} - \mathbf{z}\|_2^2 + \beta \mathbf{z}^T \mathcal{L} \mathbf{z} \\
&= \underset{\mathbf{z}}{\operatorname{argmin}} \ (\mathbf{x} - \mathbf{z})^T (\mathbf{x} - \mathbf{z}) + \beta \mathbf{z}^T \mathcal{L} \mathbf{z} \\
&= \underset{\mathbf{z}}{\operatorname{argmin}} \ \mathbf{x}^T \mathbf{x} + \mathbf{z}^T \mathbf{z} - 2\mathbf{x}^T \mathbf{z} + \beta \mathbf{z}^T \mathcal{L} \mathbf{z}
\end{aligned}$$

Substituting $\mathbf{y} = \mathbf{z}$, we next differentiate with respect to $\mathbf{y}$ and set this to 0,

$$\begin{aligned} 0 &= \nabla_{\mathbf{y}}(\mathbf{x}^T\mathbf{x} + \mathbf{y}^T\mathbf{y} - 2\mathbf{y}^T\mathbf{x} + \beta\mathbf{y}^T\mathcal{L}\mathbf{y}) \\ &= 2\mathbf{y} - 2\mathbf{x} + 2\beta\mathcal{L}\mathbf{y} \\ \mathbf{x} &= (\mathbf{I} + \beta\mathcal{L})\mathbf{y}, \end{aligned}$$

so the global minima of (3.10) can be expressed in closed form as

$$\mathbf{y} = (\mathbf{I} + \beta\mathcal{L})^{-1}\mathbf{x}. \tag{3.11}$$

As the input $\mathbf{x}$ is a graph signal in the vertex domain, the least squares solution (3.11) is a filter matrix $H_{\text{reg}} = (I + \beta\mathcal{L})^{-1}$ as discussed in **Section 3.6.1**. The spectral properties of Laplacian regularization immediately follow as

$$\begin{aligned} H_{\text{reg}} &= (\mathbf{I} + \beta\mathcal{L})^{-1} \\ &= \Psi\frac{1}{1 + \beta\Lambda}\Psi^T. \end{aligned} \tag{3.12}$$

Thus Laplacian regularization is a graph filter with frequency response $h_{\text{reg}}(\lambda) = (1 + \beta\lambda)^{-1}$. **Figure S13** shows that this function is a low-pass filter on the Laplacian eigenvalues with cutoff parameterized by β.

Tunable Filtering

Though simple low-pass filtering with Laplacian regularization is a powerful tool for many machine learning tasks, we sought to develop a filter that is flexible and capable of filtering the signal at any frequency. To accomplish these goals, we introduce the manifold heat

filter:

$$\mathbf{y} = \underset{z}{\operatorname{argmin}} \|\mathbf{x} - \mathbf{z}\|_2^2 + \mathbf{z}^T \mathcal{L}_* \mathbf{z} \tag{3.13}$$

$$\text{where } \mathcal{L}_* = \exp\big(\beta(\mathcal{L}/\lambda_{max} - \alpha\mathbf{I})^\rho\big) - \mathbf{I}$$

This filter expands upon Laplacian regularization by the addition of a new smoothness structure. Early and related work proposed the use of a power Laplacian smoothness matrix S in a similar manner as we apply here [112], but little work has since proven its utility. In our construction, α is referred to as modulation, β acts as a reconstruction penalty, and ρ is filter order. These parameters add a great deal of versatility to the manifold heat filter, and we demonstrate their spectral and vertex effects in **Figure S13**, as well as provide mathematical analysis of the MELD algorithm parameters in the following section.

A similar derivation as **Section 3.6.1** reveals the filter matrix

$$H_{\text{MELD}}(\mathcal{L}) = e^{-\beta(\mathcal{L}/\lambda_{max} - \alpha\mathbf{I})^\rho}, \tag{3.14}$$

which has the frequency response

$$h_{\text{MELD}}(\lambda) = e^{-\beta(\lambda/\lambda_{max} - \alpha)^\rho}. \tag{3.15}$$

Thus, the value of the MELD algorithm parameters in the vertex optimization (**Equation 3.13**) has a direct effect on the graph Fourier domain.

Parameter Analysis

β steepens the cutoff of the filter and shifts it more towards its central frequency (**Figure S13**). In the case of $\alpha = 0$, this frequency is $\lambda_1 = 0$. This is done by scaling all frequencies by a factor of β. For stability reasons, we choose $\beta > 0$, as a negative choice of β yields a

high frequency amplifier.

The parameters α and ρ change the filter from low pass to band pass or high pass. **Figure S13** highlights the effect on frequency response of the filters and showcases their vertex effects in simple examples. We begin our mathematical analysis with the effects of ρ.

ρ powers the Laplacian harmonics. This steepens the frequency response around the central frequency of the manifold heat filter. Higher values of ρ lead to sharper tails (**Figure S13d,e**), limiting the frequency response outside of the target band, but with increased response within the band. Finally, ρ can be used to make a high pass filter by setting it to negative values (**Figure S13f**).

For the integer powers, a basic vertex interpretation of ρ is available. Each column of $\mathcal{L}^k$ is *k-hop localized*, meaning that $\mathcal{L}^k_{ij}$ is non-zero if and only if the there exists a path length k between vertex i and vertex j (for a detailed discussion of this property, see Hammond et al. [57, section 5.2].) Thus, for $\rho \in \mathbb{N}$, the operator $\mathcal{L}^\rho$ considers variation over a hop distance of ρ. This naturally leads to the spectral behavior we demonstrate in **Figure S13d**, as signals are required to be smooth over longer hop distances when $\alpha = 0$ and $\rho > 1$.

The parameter α removes values from the diagonal of $\mathcal{L}$. This results in a *modulation* of frequency response by translating the Laplacian harmonic that yields the minimal value for the problem (**Equation 3.13**). This allows one to change the central frequency, as α effectively modulates a band-pass filter. As graph frequencies are positive, we do not consider $\alpha < 0$. In the vertex domain, the effect of α is more nuanced. We study this parameter for $\alpha > 0$ by considering a modified Laplacian $\mathcal{L}_*$ with $\rho = 1$.

To conclude, we propose a filter parameterized by reconstruction β (**Figure S13**), order ρ, and modulation α . The parameters α and β are limited to be strictly greater than or equal to 0. When $\alpha = 0$, ρ may be any integer, and it adds more low frequencies to the frequency response as it becomes more positive. On the other hand, if ρ is negative and

$\alpha = 0$, ρ controls a high pass filter. When $\alpha > 0$, the manifold heat filter becomes a band-pass filter. In standard use cases we propose to use the parameters $\alpha = 0, \beta = 60,$ and $\rho = 1$. Other parameter values are explored further in (**Figure S13**). We note that the results are relatively robust to parameter values around this default setting. All of our biological results were obtained using this parameter set, which gives a square-integrable low-pass filter. As these parameters have direct spectral effects, their implementation in an efficient graph filter is straightforward and presented below.

In contrast to previous works using Laplacian filters, our parameters allow analysis of signals that are combinations of several underlying changes occurring at various frequencies. For an intuitive example, consider that the frequency of various Google searches will vary from winter to summer (low-frequency variation), Saturday to Monday (medium-frequency variation), or morning to night (high-frequency variation). In the biological context such changes could manifest as differences in cell type abundance (low-frequency variation) and cell-cycle (medium-frequency variation) [120]. We illustrate such an example in **Figure S13** by blindly separating a medium frequency signal from a low frequency contaminating signal over simulated data. Such a technique could be used to separate low- and medium-frequency components so that they can be analyzed independently. Each of the filter parameters is explained in more detail below in the Paramter Analysis section.

Relation between MELD and the Gaussian KDE through the Heat Kernel

Kernel density estimators (KDEs) are widely used as estimating density is one of the fundamental tasks in many data applications. The density estimate is normally done in ambient space, and there are many methods to do so with a variety of advantages and disadvantages depending on the application. We instead assume that the data is sampled from some low dimensional subspace of the ambient space, e.g. that the data lies along a manifold. The MELD algorithm can be thought of as a Gaussian KDE over the discrete manifold formed by the data. This gives a density estimate at every sampled point for a

number of distributions. This density estimate, as the number of samples goes to infinity, should converge to the density estimate along a continuous manifold formed by the data. The case of data uniformly sampled on the manifold was explored in [121] proving convergence of the eigenvectors and eigenvalues of the discrete Laplacian to the eigenfunctions of the continuous manifold. Coifman and Maggioni [122] explored when the data is non-uniformly sampled from the manifold and provided a kernel which can normalize out this density which results in a Laplacian modeling the underlying manifold geometry, irrespective of data density. Building on these two works, MELD allows us to estimate the manifold geometry using multiple samples with unknown distribution along it and estimate density and conditional density for each distribution on this shared manifold.

A general kernel density estimator (KDE) $f(x,t)$ with bandwidth $t > 0$ and kernel function $K(x,y,t)$ is defined as

$$\hat{f}(x,t) = \frac{1}{N}\sum_{i=1}^{N} K(x, X_i, t), \quad x \in \mathcal{X} \tag{3.16}$$

With $\mathcal{X} := \mathbb{R}^d$, and endowed with the Gaussian kernel,

$$K(x,y,t) = \frac{1}{(4\pi t)^{d/2}} e^{-\|x-y\|_2^2/4t}, \tag{3.17}$$

we have the Gaussian KDE in $\mathbb{R}^d$.

This kernel is of particular interest for its thermodynamic interpretation. Namely the Gaussian KDE is a space discretization of the unique solution to the heat diffusion partial differential equation (PDE) [123, 77]:

$$\frac{\partial}{\partial t}\hat{f}(x,t) = \frac{1}{2}\frac{\partial^2}{\partial x^2}\hat{f}(x,t), x \in \mathcal{X}, t > 0, \tag{3.18}$$

with $\hat{f}(x,0) = \frac{1}{N}\sum_{i=1}^{n} \delta_{X_i}$ where δ_x is the Dirac measure at x. This is sometimes called

Green's function for the diffusion equation. Intuitively, $\hat{f}(x,t)$ can be thought of as measuring the heat after time t after placing units of heat on the data points at $t = 0$.

In fact the Gaussian kernel can be represented instead in terms of the eigenfunctions of the ambient space. With eigenfunctions ϕ and eigenvalues λ, the Gaussian kernel can be alternative expressed as:

$$K(x,y,t) = \sum_{n=0}^{\infty} e^{-t\lambda_n}\phi_n(x)\phi_n(y) \tag{3.19}$$

Of course for computational reasons we often prefer the closed form solution in (3.17). We now consider the case when $\mathcal{X}$ instead consists of uniform samples from a Riemannian manifold $\mathcal{M}$ embedded in $\mathbb{R}^d$ such that $\mathcal{X} \subset \mathcal{M} \subset \mathbb{R}^d$. An analog to the Gaussian KDE in $\mathbb{R}^d$ on a manifold is then the solution to the heat PDE restricted to the manifold, and again we can use the eigenfunction interpretation of the Green's function in (3.19), except replacing the eigenfunctions of the manifold.

The eigenfunctions of the manifold can be approximated through the eigenvectors of the discrete Laplacian. The solution of the heat equation on a graph is defined in terms of the discrete Laplacian $\mathcal{L} = \Psi\Lambda\Psi^{-1}$ as

$$\hat{K}_{\mathcal{L}}(x,y,t) = \delta_x e^{-t\mathcal{L}}\delta_y = \delta_x \Psi e^{-t\Lambda}\Psi^{-1}\delta_y \tag{3.20}$$

Where δ_x, δ_y are dirac functions at x and y respectively. This is equivalent to MELD when $\beta = t\lambda_{max}$, $\alpha = 0$, and $\phi = 1$.

When data $\mathcal{X}$ is sampled uniformly from the manifold $\mathcal{M}$ and the standard gaussian kernel is used to construct the graph, then Theorem 2.1 of Belkin and Niyogi [121] which proves the convergence of the eigenvalues of the discrete graph laplacian to the continuous laplacian implies (3.20) converges to the Gaussian KDE on the manifold.

However, real data is rarely uniformly sampled from a manifold. When the data is

instead sampled from a smooth density $\mathcal{X} \sim q(x)$ over the manifold then the density must be normalized out to recover the geometry of the manifold. This problem was first tackled in Coifman and Lafon [41], by constructing an anisotropic kernel which divides out the density at every point. This correction allows us to estimate density over the underlying *geometry of the manifold* even in the case where data is not uniformly sampled. This allows us to use samples from multiple distributions, in our case distributions over cellular states, which allows a better estimate of underlying manifold utilizing all available data.

In practice, we combine two methods to construct a discrete Laplacian that reflects the underlying data geometry over which we estimate heat propagation and perform density estimation, as explained in **Section 3.6.1**.

Implementation

A naïve implementation of the MELD algorithm would apply the matrix inversion presented in **Equation 3.14**. This approach is untenable for the large single-cell graphs that the MELD algorithm is designed for, as H_{MELD}^{-1} will have many elements, and, for high powers of ρ or non-sparse graphs, extremely dense. A second approach to solving **Equation 3.13** would diagonalize $\mathcal{L}$ such that the filter function in **Equation 3.15** could be applied directly to the Fourier transform of input raw experimental signals. This approach has similar shortcomings as eigendecomposition is substantively similar to inversion. Finally, a speedier approach might be to use conjugate gradient or proximal methods. In practice, we found that these methods are not well-suited for estimating sample-associated density.

Instead of gradient methods, we use Chebyshev polynomial approximations of $h_{\text{MELD}}(\lambda)$ to rapidly approximate and apply the manifold heat filter. These approximations, proposed by Hammond et al. [57] and Shuman et al. [78], have gained traction in the graph signal processing community for their efficiency and simplicity. Briefly, a truncated and shifted Chebyshev polynomial approximation is fit to the frequency response of a

graph filter. For analysis, the approximating polynomials are applied as polynomials of the Laplacian multiplied by the signal to be filtered. As Chebyshev polynomials are given by a recurrence relation, the approximation procedure reduces to a computationally efficient series of matrix-vector multiplications. For a more detailed treatment one may refer to Hammond et al. [57] where the polynomials are proposed for graph filters. For application of the manifold heat filter to a small set of input sample indicator signals, Chebyshev approximations offer the simplest and most efficient implementation of our proposed algorithm. For sufficiently large sets of samples, such as when considering hundreds of conditions, the computational cost of obtaining the Fourier basis directly may be less than repeated application of the approximation operator; in these cases, we diagonalize the Laplacian either approximately through randomized SVD or exactly using eigendecomposition, depending on user preference. Then, one simply constructs $H_{\text{MELD}} = \Psi h_{\text{MELD}}(\Lambda)\Psi^T$ to calculate the sample-associated density estimate from the input sample indicator signals.

Summary of the MELD algorithm

In summary, we have proposed a family of graph filters based on a generalization of Laplacian regularization framework to implement the computation of sample-associated density estimates on a graph. This optimization, which can be solved analytically, allows us to derive the relative likelihood of each sample in a dataset, as a smooth and denoised signal, while also respecting multi-resolution changes in the likelihood landscape. As we show in **Section 3.6.7**, this formulation performs better at deriving the true conditional likelihood in quantitative comparisons than simpler label smoothing algorithms. Further, the MELD algorithm it is efficient to compute.

The MELD algorithm is implemented in Python 3 as part of the MELD package and is built atop the `scprep`, `graphtools`, and `pygsp` packages. We developed `scprep` efficiently process single-cell data, and `graphtools` was developed for construction and manipulation of graphs built on data. Fourier analysis and Chebyshev approximations are

implemented using functions from the `pygsp` toolbox [124].

3.6.2 Vertex-frequency clustering

Next, we will describe the vertex frequency clustering algorithm for partitioning the cellular manifold into regions of similar response to experimental perturbation. For this purpose, we use a technique proposed in Shuman et al. [52] based on a graph generalization of the classical Short Time Fourier Transform (STFT). This generalization will allow us to simultaneously localize signals in both frequency and vertex domains. The output of this transform will be a spectrogram Q, where the value in each entry $Q_{i,j}$ indicates the degree to which each sample indicator signal in the neighborhood around vertex i is composed of frequency j. We then concatenate the sample-associated relative likelihood and perform k-means clustering. The resultant clusters will have similar transcriptomic profiles, similar likelihood estimates, and similar *frequency trends* of the sample indicator signals. The frequency trends of the sample indicator signals are important because they allow us to infer movements in the cellular state space that occur during experimental perturbation.

We derive vertex frequency clusters in the following steps:

1. We create the cell graph in the same way as is done in **Section 3.6.1**.

2. For each vertex in the graph (corresponding to a cell in the data), we create a series of localized windowed signals by masking the sample indicator signal using a series of heat kernels centered at the vertex. Graph Fourier decomposition of these localized windows capture frequency of the sample indicator signal at different scales around each vertex.

3. The graph Fourier representation of the localized windowed signals is thresholded using a $tanh$ activation function to produce pseudo-binary signals.

4. These pseudo-binarized signals are summed across windows of various scales to

produce a single $N \times N$ spectrogram Q. PCA is performed on the spectrogram for dimensionality reduction.

5. The sample-associated relative likelihood is concatenated to the reduced spectrogram weighted by the $L2$-norm of PC1 to produce $\hat{Q}$ which captures both local sample indicator frequency trends and changes in conditional density around each cell in both datasets.

6. k-Means is performed on the concatenated matrix to produce vertex-frequency clusters.

Analyzing frequency content of the sample indicator signal

Before we go into further detail about the algorithm, it may be useful to provide some intuitive explanations for why the frequency content of the sample indicator signal provides a useful basis for identifying clusters of cells affected by an experimental perturbation. Because the low frequency eigenvectors of the graph Laplacian identify smoothly varying axes of variance through a graph, we associate trends in the sample indicator signal associated these low-frequency eigenvectors as biological transitions between cell states. This may correspond to the shift in T cells from naive to activated, for example. We note that at intermediate cell transcriptomic states between the extreme states that are most enriched in either condition, we observe both low and middle frequency sample indicator signal components, see the blue cell in the cartoon in **Figure 3.2a**. This is because locally, the sample indicator signal varies from cell to cell, but on a large scale is varying from enriched in one condition to being enriched in the other. This is distinct from what we observe in our model when a group of cells are completely unaffected by an experimental perturbation. Here, we expect to find only high frequency variations in the sample indicator signal and no underlying transition or low-frequency component. The goal of vertex frequency clustering is to distinguish between these four cases: enriched in the experiment, enriched in

the control, intermediate transitional states, and unaffected populations of cells. We also want these clusters to have variable size so that even small groups of cells that may be differentially abundant are captured in our clusters.

Using the Windowed Graph Fourier Transform (WGFT) to identify local changes in sample indicator signal frequency

While the graph Fourier transform is useful for exploring the frequency content of a signal, it is unable to identify how the frequency content of graph signals change locally over different regions of the graph. In vertex frequency clustering, we are interested in understanding how the frequency content of the sample indicator signal changes in neighborhoods around each cell. In the time domain, the windowed Fourier transform identifies changing frequency composition of a signal over time by taking slices of the signal (e.g. a sliding window of 10 seconds) and applying a Fourier decomposition to each window independently (WFT) [109]. The result is a spectrogram Q, where the value in each cell $Q_{i,j}$ indicates the degree to which time-slice i is composed of frequency j. Recent works in GSP have generalized the constructions windowed Fourier transform to graph signals[52]. To extend the notion of a sliding window to the graph domain, Shuman et al. [52] write the operation of translation in terms of convolution as follows.

The *generalized translation operator* $T_i : \mathbb{R}^N \to \mathbb{R}^N$ of signal f to vertex $i \in \{1, 2, ..., N\}$ is given by

$$(T_i f)(n) := \sqrt{N}(f * \delta_i)(n), \quad \delta_i(j) = \begin{cases} 1 & j = i \\ 0 & j \neq i \end{cases} \tag{3.21}$$

which convolves the signal f, in our case the sample indicator signal, with a dirac at vertex i. Shuman et al. [52] demonstrate that this operator inherits various properties of its classical counterpart; however, the operator is not isometric and is affected by the graph

that it is built on. Furthermore, for signals that are not tightly localized in the vertex domain and on graphs that are not directly related to Fourier harmonics (e.g., the circle graph), it is not clear what graph translation implies.

In addition to translation, a *generalized modulation operator* is defined by Shuman et al. [52] as $M_k : \mathbb{R}^N \to \mathbb{R}^N$ for frequencies $k \in \{0, 1, ..., N-1\}$ as

$$(M_k f)(n) := \sqrt{N} f(n) U_k(n) \tag{3.22}$$

This formulation is analogous in construction to classical modulation, defined by pointwise multiplication with a pure harmonic – a Laplacian eigenvector in our case. Classical modulation translates signals in the Fourier domain; because of the discrete nature of the graph Fourier domain, this property is only weakly shared between the two operators. Instead, the generalized modulation M_k translates the *DC component* of f, $\hat{f}(0)$, to λ_k, i.e. $\widehat{(M_k f)}(\lambda_k) = \hat{f}(0)$. Furthermore, for any function f whose frequency content is localized around λ_0, $(M_k f)$ is localized in frequency around λ_k. Shuman et al. [52] details this construction and provides bounds on spectral localization and other properties.

With these two operators, a graph windowed Fourier atom is constructed[52] for any window function $g \in \mathbb{R}^N$

$$g_{i,k}(n) := (M_k T_i g)(n) = N U_k(n) \sum_{\ell=0}^{N-1} \hat{g}(\lambda_\ell) U_\ell^*(i) U_\ell(n). \tag{3.23}$$

We can then build a spectrogram $Q = (q_{ik}) \in \mathbb{R}^{N \times N}$ by taking the inner product of each $g_{i,k} \forall i \in \{1, 2, ..., N\} \wedge \forall k \in \{0, 1, ..., N-1\}$ with the target signal f

$$q_{ik} = Sf(i,k) := \langle f, g_{i,k} \rangle. \tag{3.24}$$

As with the classical windowed Fourier transform, one could interpret this as segmenting

the signal by windows and then taking the Fourier transform of each segment

$$q_i = \langle (T_i g \odot f), U \rangle \tag{3.25}$$

where $\odot$ is the element-wise product.

Using heat kernels of increasing scales to produce the WGFT of the sample indicator signal

To generate the spectrogram for clustering, we first need a suitable window function. We use the normalized heat kernel as proposed by Shuman et al. [52]

$$\hat{g}(\lambda) = Ce^{-t\lambda}, \tag{3.26}$$

$$C = ||g||_2^{-1}. \tag{3.27}$$

By translating this kernel, element-wise multiplying it with our target signal f and taking the Fourier transform of the result, we obtain a windowed graph Fourier transform of f that is localized based on the *diffusion distance* [108, 52] from each vertex to every other vertex in the graph.

For an input sample indicator signal $\mathbf{f}$, signal-biased spectral clustering as proposed by Shuman et al. [52] proceeds as follows:

1. Generate the window matrix P_t, which contains as its columns translated and normalized heat kernels at the scale t

2. Column-wise multiply $F_t = P \odot \mathbf{f}$; the i-th column of F_t is an entry-wise product of the i-th window and $\mathbf{f}$.

3. Take the Fourier Transform of each column of F_t. This matrix, $\hat{C}_t$ is the normalized WGFT matrix.

This produces a single WGFT for the scale t. At this stage, Shuman et al. [52] proposed to saturate the elements of $\hat{C}_t$ using the activation function $\tanh(|\hat{C}_t|)$ (where $|\,.\,|$ is an element-wise absolute value). Then, k-means is performed on this saturated output to yield clusters. This operation has connections to spectral clustering as the features that k-means is run on are coefficients of graph harmonics.

We build upon this approach to add robustness, sensitivity to sign changes, and scalability. Particularly, vertex-frequency clustering builds a set of activated spectrograms at different window scales. These scales are given by simulated heat diffusion over the graph by adjusting the time-scale t in **Equation 3.26**. Then, the entire set is combined through summation.

Combining the sample-associated relative likelihood and WGFT of the sample indicator signal

As discussed in **Section 3.2.4**, it is useful to consider the value of the sample likelihood in addition to the frequency content of the sample indicator. This is because if we consider two populations of cells, one of which is highly enriched in the experimental condition and another that is enriched in the control, we expect to find similar frequency content of the sample indicator signal. Namely, both should have very low-frequency content, as indicated in the cartoon in **Figure 3.2a**. However, we expect these two populations to have very different sample likelihood values. To allow us to distinguish between these populations, we also include the sample-associated relative likelihood in the matrix used for clustering.

We concatenate the sample-associated relative likelihood as an additional column to the multi-resolution spectrogram Q. However, we want to be able to tune the clustering with respect to how much the likelihood affects the result compared to the frequency information in Q. Therefore, inspired by spectral clustering as proposed by [61], we first perform PCA on Q to get $k+1$ principle components and then normalize the likelihood by

the $L2$-norm of the first principle component. We then add the likelihood as an additional column to the PCA-reduced Q to produce the matrix $\hat{Q}$. The weight of the likelihood can be modulated by a user-adjustable parameter w, but for all experiments in this chapter, we leave $w = 1$. Finally, $\hat{Q}$ is used as input for k-means clustering.

The multiscale approach we have proposed has a number of benefits. Foremost, it removes the complexity of picking a window-size. Second, using the actual input signal as a feature allows the clustering to consider both frequency and sign information in the raw experimental signal. For scalability, we leverage the fact that P_t is effectively a diffusion operator and thus can be built efficiently by treating it as a Markov matrix and normalizing the graph adjacency by the degree.

Summary of the vertex frequency clustering algorithm

To identify clusters of cells that are transcriptionally similar and also affected by an experimental perturbation in the same way, we introduced an algorithm called vertex frequency clustering. Our approach builds on previous work by Shuman et al. [52] analyzing the local frequency content of the sample indicator vector as defined over the vertices of a graph. Here, we introduce two novel adaptations of the algorithm. First, we take a multiresolution approach to quantifying frequency trends in the neighborhoods around each node. By considering windowed signals that are large (i.e. contain many neighboring points) and small (i.e. very proximal on the graph), we can identify clusters both large and small that are similarly affected by an experimental perturbation. Our second contribution is the inclusion of the relative likelihood of each sample in our basis for clustering. This allows VFC to take into account the degree of enrichment of each group of cells between condition.

3.6.3 Parameter search for the MELD algorithm

To determine the optimal set of parameters for the MELD algorithm, we performed a parameter search using splatter-generated datasets. For each of the four dataset structures, we generated 10 datasets with different random seeds and 10 random ground-truth probability densities per dataset for a total of 400 datasets per combination of parameters. A coarse-grained grid search revealed that setting $\alpha = 0$ and $\rho = 1$ performed best regardless of the β parameter. This is expected because with these settings, the MELD filter is the standard heat kernel. A fine-grained search over parameters for β showed that optimal values were between 50-75 (**Figure S14**). We chose a value of 60 as the default in the MELD toolkit and this was used for all experiments. We would like to note that the optimal β parameter will vary with dataset structure and the number of cells. **Figure S14b** shows how the optimal β values varies as a function of the number of cells generated using splatter while keeping the underlying geometry the same.

3.6.4 Processing and analysis of the T-cell datasets

Gene expression counts matrices prepared by Datlinger et al. [8] were accessed from the NCBI GEO database accession GSE92872. 3,143 stimulated and 2,597 unstimulated T-cells were processed in a pipeline derived from the published supplementary software. First, artificial genes corresponding to gRNAs were removed from the counts matrix. Genes observed in fewer than five cells were removed. Cells with a library size higher than 35,000 UMI / cell were removed. To filter dead or dying cells, expression of all mitochondrial genes was z-scored and cells with average z-score expression greater than 1 were removed. As in the published analysis, all mitochondrial and ribosomal genes were excluded. Filtered cells and genes were library size normalized and square-root transformed. To build a cell-state graph, 100 PCA dimensions were calculated and edge weights between cells were calculated using an alpha-decay kernel as implemented in

the Graphtools library (www.github.com/KrishnaswamyLab/graphtools) using default parameters. MELD was run on the cell state graph using the stimulated / unstimulated labels as input with the smoothing parameter β = 60. To identify a signature, the top and bottom VFC clusters by sample-associated relative likelihood were used for differential expression using a rank test as implemented in diffxpy [81] and a q-value cutoff of 0.05. GO term enrichment was performed using EnrichR using the gseapy Python package (`https://pypi.org/project/gseapy/`).

3.6.5 Processing and analysis of the zebrafish dataset

Gene expression counts matrices prepared by Wagner et al. [75] (the chordin dataset) were downloaded from NCBI GEO (GSE112294). 16079 cells from *chd* embryos injected with gRNAs targeting chordin and 10782 cells from *tyr* embryos injected with gRNAs targeting tyrosinase were accessed. Lowly expressed genes detected in fewer than 5 cells were removed. Cells with library sizes larger than 15,000 UMI / cell were removed. Counts were library-size normalized and square root transformed. Cluster labels included with the counts matrices were used for cell type identification.

During preliminary analysis, a group of 24 cells were identified originating exclusively from the *chd* embryos. Despite an average library size in the bottom 12% of cells, these cells exhibited 546-fold, 246-fold, and 1210-fold increased expression of Sh3Tc1, LOC101882117, and LOC101885394 respectively relative to other cells. To the best of our knowledge, the function of these genes in development is not described. These cells were annotated by Wagner et al. [75] as belonging to 7 cell types including the Tailbud – Spinal Cord and Neural – Midbrain. These cells were excluded from further analysis.

To generate a cell state graph, 100 PCA dimensions were calculated from the square root transformed filtered gene expression matrix of both datasets. Edge weights between cells on the graph were calculated using an alpha-decay kernel with parameters knn=20,

decay=40. MAGIC was used to impute gene expression values using default parameters. MELD was run using the *tyr* or *chd* labels as input. The sample-associated density estimate was calculated for each of the 6 samples independently and normalized per replicate to generate 3 chordin relative likelihood estimates. The average likelihood for the chordin condition was calculated and used for downstream analysis. To identify subpopulations within the published clusters, we manually examined a PHATE embedding of each subcluster, the distribution of chordin likelihood values in each cluster, and the results of VFC subclustering with varying numbers of clusters. The decision to apply VFC was done one a per-cluster basis with the goal of identifying cell subpopulations with transcriptional similarity (as assessed by visualization) and uniform response to perturbation (as assessed by likelihood values). Cell types were annotated using sets of marker genes curated by Farrell et al. [76]. Changes in gene expression between VFC clusters was assess using a rank sum test as implemented by `diffxpy`.

3.6.6 Generation, processing and analysis of the pancreatic islet data

Single-cell RNA-sequencing was performed on human islet cells from three different islet donors in the presence and absence of IFNγ. The islets were received on three different days. Cells were cultured for 24 hours with 25ng/mL IFNγ (R&D Systems) in CMRL 1066 medium (Gibco) and subsequently dissociated into single-cells with 0.05% Trypsin EDTA (Gibco). Cells were then stained with FluoZin-3 (Invitrogen) and TMRE (Life Technologies) and sorted using a FACS Aria II (BD). The three samples were pooled for the sequencing. Cells were immediately processed using the 10X Genomics Chromium 3' Single-Cell RNA-sequencing kit at the Yale Center for Genome Analysis. The raw sequencing data was processed using the 10X Genomics Cell Ranger Pipeline. Raw data will be made available prior to publication.

Data from all three donors was concatenated into a single matrix for analysis. First,

cells not expressing insulin, somatostatin, or glucagon were excluded from analysis using donor-specific thresholds. The data was square root transformed and reduced to 100 PCA dimensions. Next, we applied an MNN kernel to create a graph across all three donors with parameters knn=5, decay=30. This graph was then used for PHATE. MELD was run on the sample labels using default parameters. To identify coarse-grained cell types, we used previously published markers of islet cells [89]. We then used VFC to identify subpopulations of stimulated and unstimulated islet cells. To identify signature genes of IFNγ stimulation, we calculated differential expression between the clusters with the highest and lowest treatment likelihood values within each cell type using a rank sum test as implemented in diffxpy. A consensus signature was then obtained by taking the intersection genes with q-values < 0.05. Gene set enrichment was then calculated using gseapy.

3.6.7 Quantitative comparisons

To generate single-cell data for the quantitative comparisons, we used Splatter. Datasets were all generated using the "Paths" mode so that a latent dimension in the data could be used to create the ground truth likelihood that each cell would be observed in the "experimental" condition relative to the "control". We focused on four data geometries: a tree with three branches, a branch and cluster with either end of the branch enriched or depleted and the cluster unaffected, a single branch with a middle section either enriched or depleted, and four clusters with random segments enriched or depleted. To create clusters, a multi-branched tree was created, and all but the tips of the branches were removed. The ground truth experimental signal was created using custom Python scripts taking the "Steps" latent variable from Splatter and randomly selecting a proportion of each branch or cluster between 10% and 80% of the data was enriched or depleted by 25%. These regions were divided into thirds to create a smooth transition between the unaffected regions

and the differentially abundant regions. This likelihood ratio was then centered so that, on average, half the cells would be assigned to each condition. The centered ground truth signal was used to parameterize a Bernoulli random variable and assign each cell to the experimental or control conditions. The data and sample labels were used as input to the respective algorithms.

To quantify the accuracy of MELD to approximate the ground truth likelihood ratio, we compared MELD, a kNN-smoothed signal, or a graph averaged signal to the ground truth likelihood of observing each cell in either of the two conditions. We used the Pearson's R statistic to calculate the degree to which these estimates approximate the likelihood ratio. Each of the four data geometries was tested 30 times with different random seeds.

We also performed MELD comparisons using the T cell and zebrafish datasets described above. The preprocessed data was used to generate a three-dimensional PHATE embedding that was z-score normalized. We then used a combination of PHATE dimensions to create a ground truth probability each cell would be observed in the experimental or control condition. Cells were then assigned to either condition based on this probability as described above. We ran the same comparisons as on the simulated data with 100 random seeds per dataset.

To quantify the accuracy of VFC to detect the regions of the dataset that were enriched, depleted, or unaffected between conditions, we calculated the Adjusted Rand Score between the ground truth regions with enriched, depleted, or unchanged likelihood ratios between conditions. VFC was compared to k-Means, Spectral Clustering, Louvain, Leiden, and CellHarmony. As Leiden and Louvain do not provide a method to control the number of clusters, we implemented a binary search to identify a resolution parameter that provides the target number of clusters. Although Cell Harmony relies on an initial Louvain clustering, the tool does not implement Louvain with a tuneable resolution. It is also not possible to provide an initial clustering to CellHarmony, so we resorted to cutting Louvain at the level closest to our target number of clusters. Finally, because CellHarmony

does not reconcile the disparate cluster assignments in the reference and query datasets, and because not all cells in the query dataset may be aligned to the reference we needed to generate manually new cluster labels for cells in the query dataset so that the method could be compared to other clustering tools.

To characterize the ability of MELD to characterize gene signatures of a perturbation dataset, we returned to the T cell dataset. We again used the same setup to create synthetically 3 regions with different sampling probabilities in the dataset using PHATE clusters as above. Because one of these clusters has no differential abundance between conditions, we calculated the ground truth gene expression signature between the enriched and depleted clusters only using diffxpy [81]. To calculate the gene signature for each clustering method, we performed differential expression between the most enriched cluster in the experimental condition and the most depleted cluster in the experimental condition (or highest and lowest treatment likelihood for MELD). We also considered directly performing two-sample comparison using the sample labels. To quantify the performance of each method, we used the area under the receiving operator characteristic (AUCROC) to compare the q-values produced using each method to the ground truth q-values. This process was repeated over 100 random seeds. The AUCROC curves and performance of each method relative to VFC is displayed in **Figure S6d,e**.

3.7 Data availability

Gene expression counts matrices prepared by Datlinger et al. [8] were accessed from the NCBI GEO database accession GSE92872. Gene expression counts matrices prepared by Wagner et al. [75] were downloaded from NCBI GEO accession GSE112294. The pancreatic islets datasets are available on NCBI GEO at accession GSE161465.

3.8 Code availability

Code for the MELD and VFC algorithms implemented in Python is available as part of the MELD package on GitHub `https://github.com/KrishnaswamyLab/MELD` and on the Python Package Index (PyPI). The GitHub repository also contains tutorials, code to reproduce the analysis of the zebrafish dataset, and code associated with several of the quantitative comparisons.

3.9 Supplementary Notes

Supplementary Note 1: A pipeline for analyzing single-cell data using MELD

Using the MELD algorithm and VFC, it is now possible to propose a novel framework for analyzing single-cell perturbation experiments. The goal of this framework is to identify populations of cells that are the most affected by an experimental perturbation and to characterize a gene signature of that perturbation. A schematic of the proposed pipeline is shown in **Figure 3.10**.

Prior to using the algorithms in MELD, we recommended first following established best practices for analysis of single-cell data including exploratory analysis using visualization, preliminary clustering, and cluster annotation via differential expression analysis [68]. These steps ensure that the dataset is of high quality and comprises the cell types expected from the experimental setup. Following exploratory characterization, we propose the following analysis:

1. Estimate the sample-associate relative likelihood for each condition

2. Determine which exploratory clusters require subclustering with VFC by examining the likelihood distribution within each cluster, a visualization of the cluster, and the results of VFC with varying numbers of clusters

3. Create new cluster assignments using VFC

4. Annotate each cluster following best practices [68]

5. Characterize enrichment of cell populations using sample likelihood and gene signatures

The basic steps to calculate the sample-associated relative likelihood are provided in the Results. In the case of multiple replicates, we recommend calculating the sample density for each sample over a graph of all cells from all samples so long as there is sufficient overlap between samples. This overlap can be assessed using the k-nearest neighbor batch effect test described in Büttner et al. [125]. We then normalize the sample density for matched experimental and control samples of the same replicate and average across replicates to obtain an average measure of the perturbation. Variation in this likelihood across replicates can be used as a measure of consistency for the measured perturbation across cell types. The result of this step is an estimate of the probability that each cell would be observed in the treatment condition relative to the control.

Having calculated the sample-associated relative likelihood, we next recommend determining which cell populations identified during exploratory analysis require further subclustering with VFC to identify cell types enriched or depleted in the experimental condition. Determining optimal cluster resolution for single-cell analysis will vary across experiments depending on the biological system being studied and the goals of each individual researcher. Instead of providing a single measure to determine the number of clusters, we outline a general strategy as a guide for users of MELD.

To determine the number of VFC clusters, we suggest taking into consideration transcriptional variation within each coarse-grained cluster and the effect of the perturbation. First, using a dimensionality reduction tool such as PHATE, examine a two or three dimensional scatter plot of the cluster colored by the sample likelihood for each cell. Here, the goal is to identify either regions that have very different likelihood values or regions of

data density separated by low-density regions suggesting the present of multiple subclusters to target with VFC. We also suggest examining the distribution the likelihood values within each cluster to determine if the cells in the cluster exhibit a restricted range of responses to the perturbation or large variation that would require subclustering. Finally, we recommend running VFC with various numbers of clusters (2-5 is often sufficient) and inspecting the output on a PHATE plot and/or with a swarm plot. In ambiguous cases, it may be helpful to perform differential expression analysis and gene set enrichment to determine whether or not each cluster is biologically relevant to the experimental question under consideration [68, 126]. Importantly, not all clusters need subclustering, and we emphasize the ideal cluster resolution will vary based on the goals of each analyst.

To determine the gene signature of the perturbation, we recommend quantifying the differences in expression between VFC clusters. For experiments with only a single-cell type and 3-4 VFC clusters, it is often sufficient to perform differential expression analysis between the cluster most enriched in the experimental condition and the cluster most depleted in the experimental condition. And example of this analysis is provided in the T cell analysis section of the **Results**. For experiments with several cell types, we recommend calculating the gene signature between the enriched and depleted VFC clusters within each exploratory cluster. To obtain a consensus gene signature, a research may take the intersection of the gene signatures within exploratory cluster. An example of this analysis is provided in the pancreatic islets section of the **Results**.

We note that the strategy for identifying gene signatures outlined in the previous paragraph differs from the current framework employed in recent papers (**Figure 3.9**). Instead of comparing expression between cells from the experimental condition and the control, we compare clusters of cells identified with VFC. The rationale for the framework presented here is that if VFC clusters are transcriptionally homogeneous and exhibit a uniform response to the perturbation, we expect differences in gene expression between conditions *within* each cluster to represent biological and technical noise. However, characterizing

transcriptional differences *between* cells of different clusters regardless of condition of origin will yield a description of the cell states that vary between experimental conditions. We confirm that the gene signatures obtained in this manner are more accurate than between-sample comparisons in our quantitative comparisons.

Supplementary Note 2: VFC improves analysis of *chd* Cas9 knockout in zebrafish embryos Here we provide details of our analysis of three clusters in the zebrafish datasets [75] that required further subclustering using VFC. In each example, we show biologically relevant insights that were missed in the published analysis.

The Tailbud – Presomitic Mesoderm (TPM) cluster exhibits the largest range of chordin relative likelihood values of all the clusters annotated by Wagner et al. [75]. In a PHATE visualization of the cluster, we observe many different branches of cell states, each with varying ranges of chordin relative likelihood values (**Figure 5c**). Within the TPM cluster, we find four subclusters using VFC (**Figure 5d**). Using established markers [76], we identify these clusters as immature adaxial cells, mature adaxial cells, presomitic mesoderm cells, and hematopoietic cells (**Figures 5c & 3.16**). Examining the distribution of chordin relative likelihood scores within each cell type, we conclude that the large range of chordin relative likelihood values within the TPM cluster is due to largely non-overlapping distributions of scores within each of these subpopulations (**Figure 5e**). The immature and mature adaxial cells, which are embryonic muscle precursors, have low chordin relative likelihood values indicating depletion of these cells in the *chd* condition which matches observed depletion of myotomal cells in chordin mutants [84]. Conversely, the presomitic mesoderm and hematopoietic mesoderm have high chordin relative likelihood values, indicating that these cells are enriched in a chordin mutant. Indeed, expansion of the hematopoietic mesoderm has been observed in chordin morphants [127] and expansion of the presomitic mesoderm was observed in siblings of the *chd* embryos by Wagner et al. [75]. This heterogeneous effect was entirely missed by the fold-change analysis, since the averaging of all cells assigned to the TPM cluster caused the depletion of adaxial

cells to be masked by the expansion of the presomitic and hematopoietic mesoderm.

Another advantage of vertex-frequency clustering is that we can now differentiate between a change in gene expression levels across conditions and a change in abundance of cells expressing a given gene between conditions. When we examined marker gene expression within each of the VFC subclusters, we find different trends in expression in each cluster (**Figure 5f**). For example, Myod1, a marker of adaxial cells, is lowly expressed in the presomitic and hematopoietic mesoderm, but highly expressed in adaxial cells. Using a rank sum test, we find that Myod1 is not differentially expressed between conditions within any of the VFC clusters despite there being differential expression using all cells in the TPM cluster (**Figure 5f**). We find a similar trend with Tbx6, a mesoderm marker that is not expressed in adaxial cells. We find Tbx6 is differentially expressed between *chd* and *tyr* embryos within the whole cluster but not within the adaxial or presomitic mesoderm clusters. These results show that the observed change in expression of these genes in the published analysis was in fact due to changes in abundance of cell subpopulations that led to misleading differences in statistics calculated across multiple populations as a whole. Using the chordin relative likelihood and VFC, we can identify more appropriate clusters.

We similarly analyzed the "Epidermal - pfn1 (EPP)" and "Tailbud - Spinal Cord (TSC)" clusters which had the 6th and 3rd largest standard deviation in chordin relative likelihood values of all published clusters, respectively (**Figure 3.16**). We used VFC to break up the Epidermal - pfn1 cluster into two subclusters. Among the top differentially expressed genes between the resulting clusters we find tbx2b, crabp2a, and pfn1. Crabp2a, a marker of the neural plate border [76], is more lowly expressed in the cluster with higher chordin relative likelihood values, suggesting that *chd* loss-of-function inhibits expression of crabp2a. This is consistent with previous studies showing a requirement of chordin for proper gene expression patterning within the neural plate [128, 129].

Within the Tailbud - Spinal Cord cluster we further identified three subpopulations of cells using VFC. Examining gene expression within the subclusters, we can see that the

published cluster contains different populations of cells. One group expresses markers of the spinal cord (neurog, elavl3) and dorsal tissues (olig3, pax6a/b) with an average chordin relative likelihood of 0.38, which is consistent with prior evidence that *chd* loss-of-function disrupts specification of the neuroectoderm and dorsal tissues such as the spinal cord [84]. Examining the two remaining subclusters, we see that these cells resemble cells found in both the TPM and Epidermal - Pfn1 clusters. One cluster exhibits high levels of crabp2a and chordin relative likelihood values <0.5 similar to the neural plate border cells subpopulation within the Epidermal - Pfn1 cluster. Similarly, we find the remaining cluster expressed markers of the tailbud and presomitic mesoderm including tbx6, sox2, and fgf8a. Together, these results demonstrate the advantage of using the sample-associated relative likelihood and vertex frequency clustering to quantify the effect of genetic loss-of-function perturbations in a complex system with many cell types.

Supplementary Note 3: Applying MELD analysis to single-cell datasets with a batch effect

When jointly analyzing single-cell datasets collected in different samples, difficulty may arise due to systematic changes in gene expression profiles between biologically equivalent cells [125]. These changes may be technical in nature (e.g. differences in the reverse transcription efficiency during library preparation) or biological (e.g. changes in sample preparation cause unexpected changes in biological state of otherwise equivalent cells). Regardless of the cause, the unifying feature of batch effects is that they confound the analysis a given research wants to perform. As such, it is unsurprising that dozens of batch normalization tools have been developed for single-cell data [130]. However, it is important to emphasize that what constitutes a batch effect is dependent on the biological question in which a researcher is interested. Some analysts might be uninterested in variation caused by a change in cell media composition between samples, but other researchers might want to study these differences. Batch normalization tools have no way to know what variation is biologically relevant to the specific hypotheses of a given experiment and

thus risk removing meaningful experimental signal when ”correcting” measured values. This is problematic for analysis using MELD, because the goal of the toolkit is to quantify the differences that exist between samples without regard for the specific interests of given hypothesis. As such, we do not recommend using batch correction along the experimental axis (i.e. between experimental and control conditions) before running MELD. However, recognizing that in some cases batch correction is essential, we describe several considerations for performing MELD analysis on batch-corrected data.

For the MELD algorithm to accurately estimate relative likelihood for each sample, we assume that the graph learned from single-cell data approximates the underlying cell state manifold. In the **Methods** we describe the use of an anisotropic kernel that normalizes for varying sampling density across cell states. However, some batch correction methods, such as mutual nearest neighbors [102], rely on the construction of a graph with artificially inflated weights between nodes from different samples. This graph no longer models the cell states an experiment measured, but rather enforces similarities between cells based on the heuristic of the chosen normalization model. We provide no theoretical guarantees that a graph learned from batch corrected data will accurately model the underlying probability densities of each condition.

In practice when analyzing islet cells collected from multiple donors, that applying batch correction methods across the donor label improves our ability to capture a signal of IFNg stimulation. It is important to note that in this case, batch correction applied to a label that is orthogonal to the experimental axis. We have no examined the accuracy of the MELD algorithm when batch correction is applied between experimental and control samples, although it is our expectation that this will likely remove biological signal. We recommend any user considering applying batch correction methods prior to running MELD analysis follow these steps:

1. To determine if a batch effect exists, confirm that cells from one sample are not

finding appropriate neighbors in another following the strategy outlined by Büttner et al. [125].

2. To characterize the effect, identify which genes change the most between the samples

3. Confirm that the genes that are different are not relevant to the biological question under investigation

4. Apply batch correction

5. Confirm that relevant biological differences are still present using MELD analysis

6. If the biological differences are not present, repeat from step 1 with less batch correction. If you hit your personal recursion limit, consider that you don't actually want to do batch correction

7. If biological differences are present, then confirm that previous batch effect has been corrected and proceed to downstream analysis

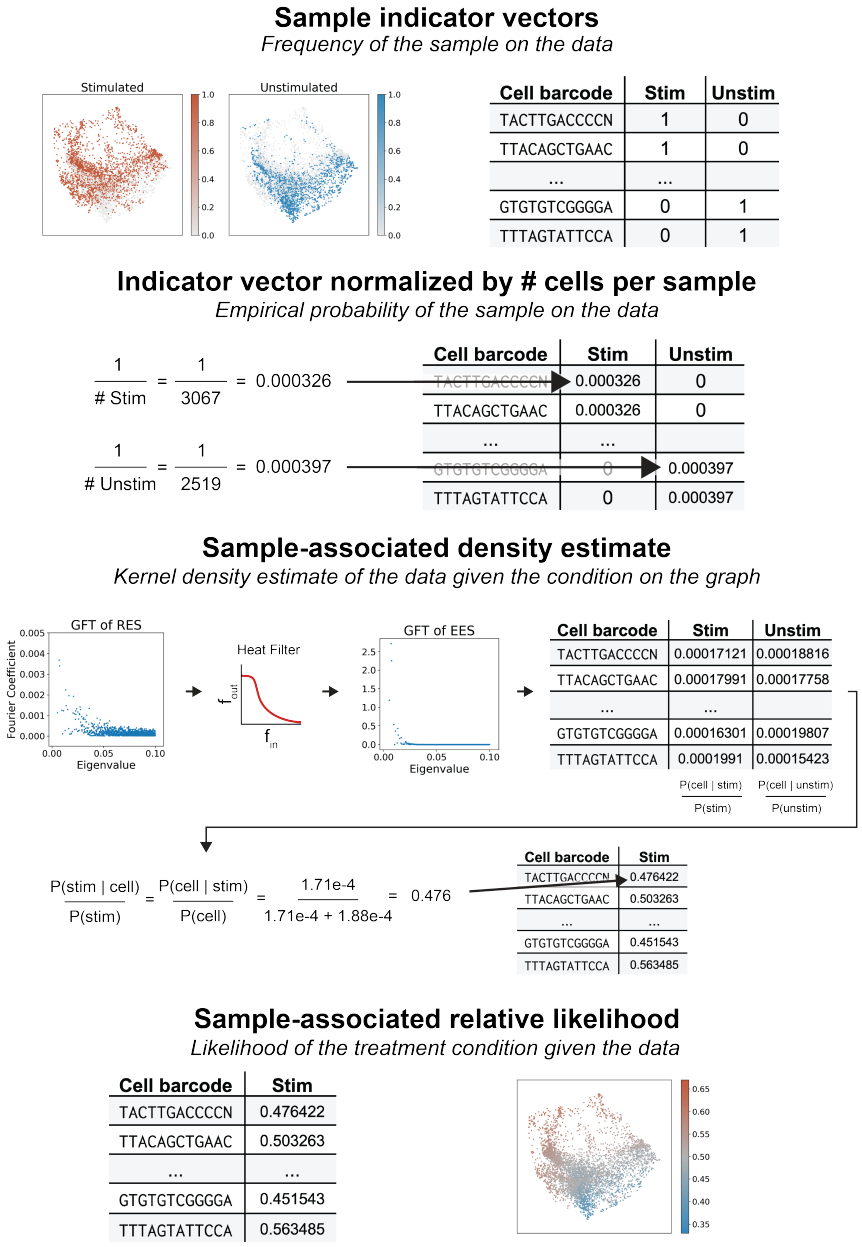

Figure 3.7: A step-by-step visual representation of the MELD algorithm using data from Datlinger et al. [8]. The sample labels are used to create a one-hot indicator signal for each condition. These one-hot signals are then column-wise L1-normalized such that the sum of each vector is 1. This gives each sample equal weight over the manifold despite a potential uneven number of cells in each condition. Next, the manifold heat filter is used to calculate a kernel density estimate for each condition. These sample-associated density estimates are then row-wise L1-normalized to yield the relative likelihood that each cell would be observed in each condition. The relative likelihood of the treatment condition relative to the control is used for two-condition experiments.

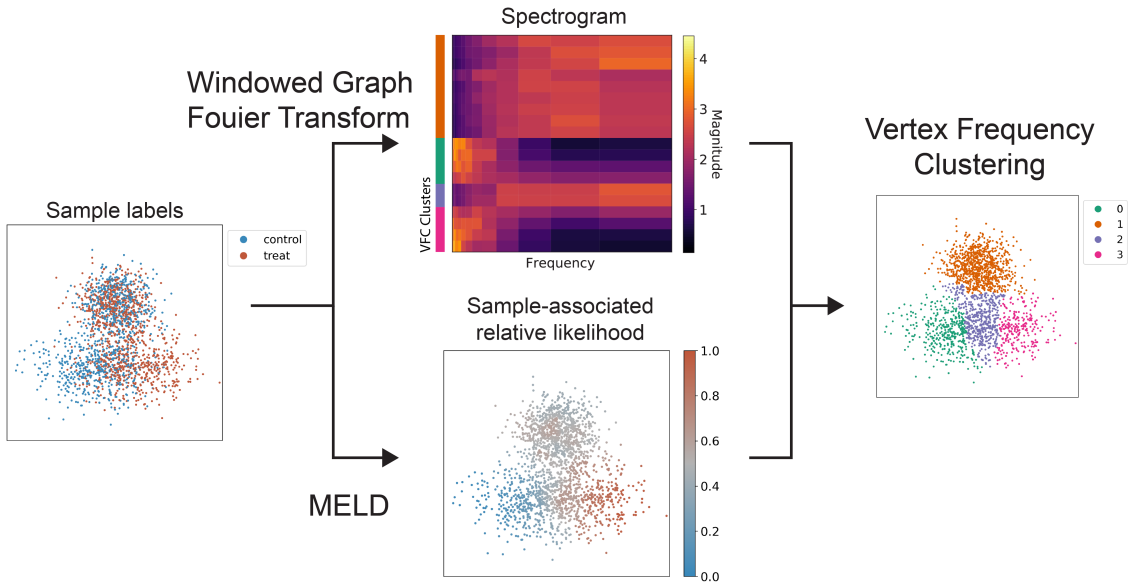

Figure 3.8: Vertex-Frequency clustering with MELD. A Gaussian mixture model was used to generate N = 2000 points in a mixture of three Gaussian distributions. This experiment is representative of a two-cell type experiment (split by Dim 2) in which one sample changes (bottom clusters) along Dim 1 due to the experiment while the other remains mixed (top clusters). Briefly, the sample labels (left) are used for (1) a windowed graph Fourier Transform to obtain vertex-frequency information (above, logarithmically downsampled for clarity) and (2) to calculate the sample-associated relative likelihood. These measures are concatenated together and clustered with *k*-Means. The clusters (right) separate the two groups of data (orange and green/purple/pink), and finds a separate grouping of points that are in transition from green to pink, shown in purple. One may see along the left side of the spectrogram that points in the green and pink clusters are found on relatively low frequency patterns with high activations in lower frequencies, whereas the transition group in purple has a well-separated medium frequency pattern. The well-mixed, nonresponsive population is entirely high frequency.

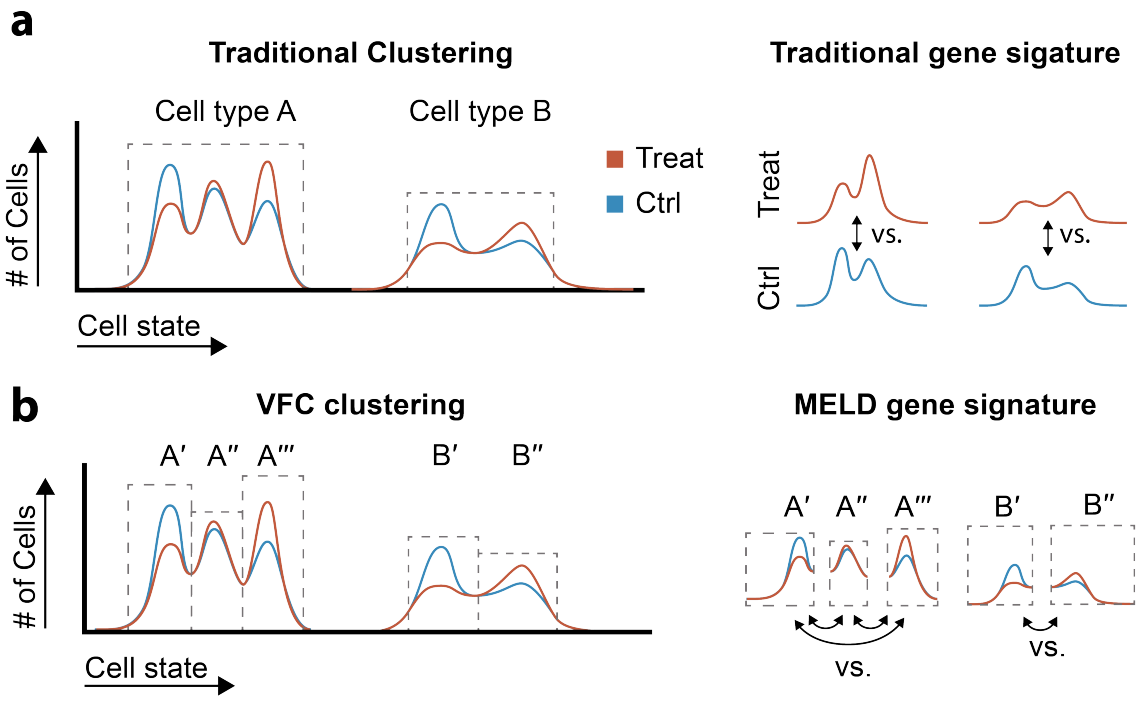

Figure 3.9: Identifying gene signatures using MELD. (**a**) In traditional gene signature analysis, clusters are identified based on data geometry and may not capture subpopulations of cells with varying response to a perturbation. In this framework, gene signatures are calculated by comparing cells from the experimental and control condition within each cluster. (**b**) To identify gene signatures of a perturbation with the MELD toolkit, we propose first partitioning cell populations with divergent responses to an experimental perturbation prior to differential expression analysis. We then assume that the differences within each VFC cluster is noise. Differential expression can either be calculated between subclusters identified by VFC (as shown) or by comparing each VFC cluster to the rest of the dataset independently.

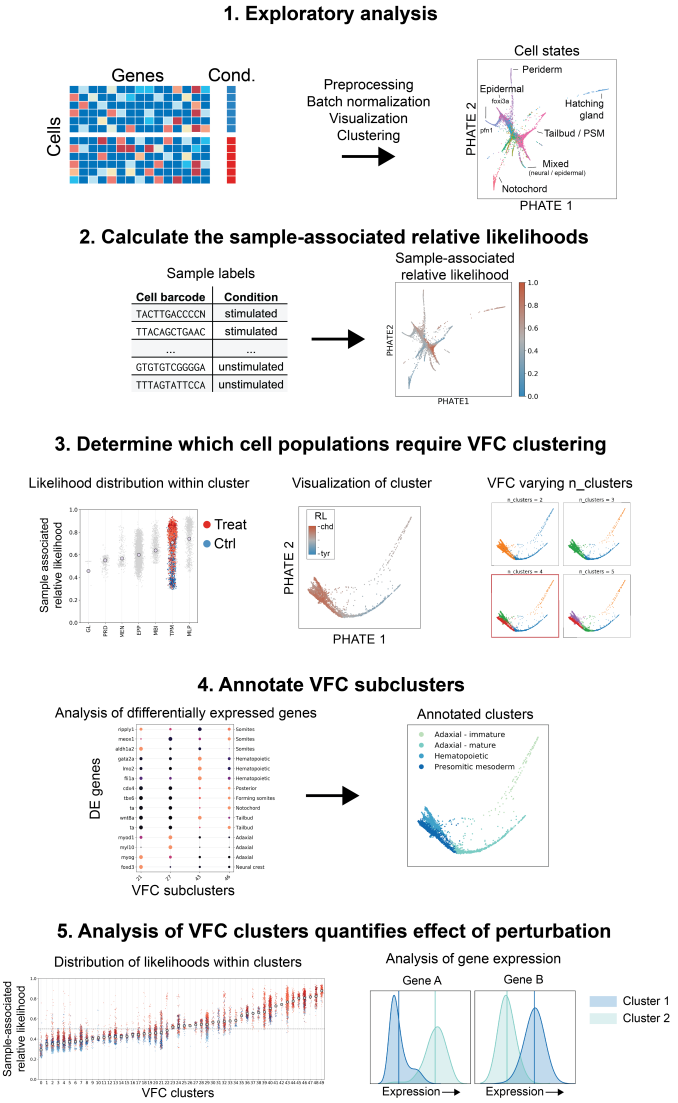

Figure 3.10: Overview of a pipeline for single-cell analysis using MELD. (**1.**) Initial exploratory analysis of the dataset should follow established best practices to identify coarse-grained cell populations [68, 126]. (**2.**) Calculating the sample-associated relative likelihood provides a measure for each cell describing the probability that cell would be observed in the experimental condition relative to the control. (**3.**) To identify populations most affected by a perturbation, we consider several sources of information regarding biological heterogeneity and the effect of the perturbation within each exploratory cluster. We then apply VFC at the determined cluster resolution. (**4.**) To assess the biological relevance of each VFC cluster, standard methods for cluster annotation can be applied. (**5.**) To characterize the gene signature of the perturbation, we compare expression differences between VFC clusters with varying relative likelihood distributions.

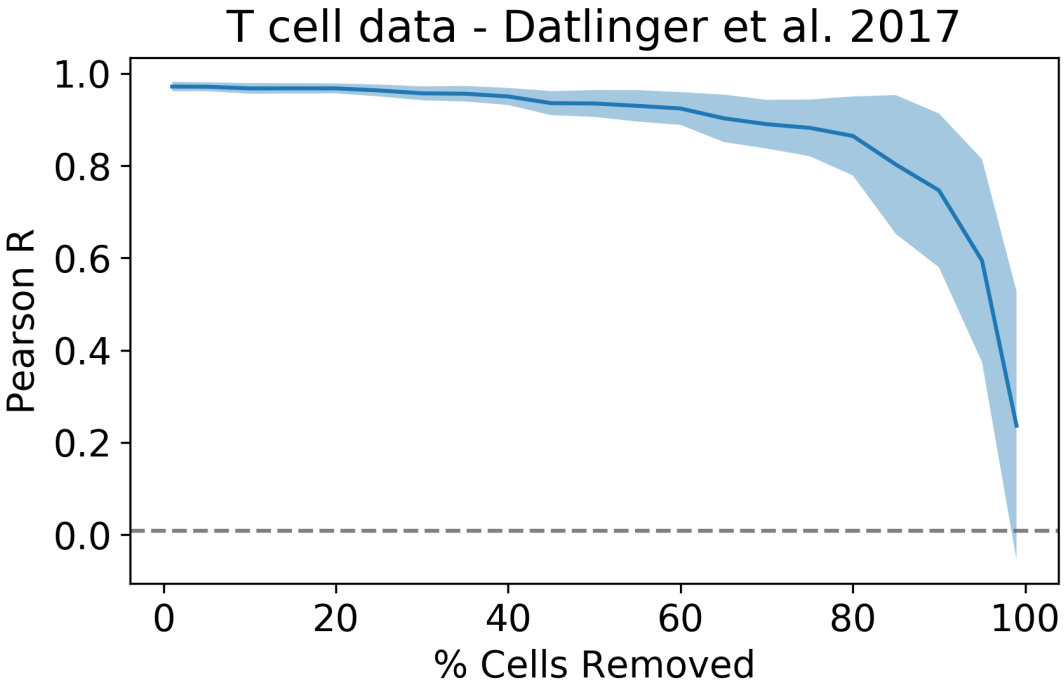

Figure 3.11: Result of down-sampling on accurately recovering simulated relative likelihood values. We generated 100 random ground truth relative likelihoods and then removed between 1-99% of the cells in the dataset before running the MELD with default parameters. The average Pearson's R is shown as a function of the number of cells removed prior to estimating the sample-associated relative likelihood. The shaded area demarks ± 1 standard deviation. We observe an average correlation >0.9 for all experiments with at least 35% of the data present, or 1956 out of 5591 cells.

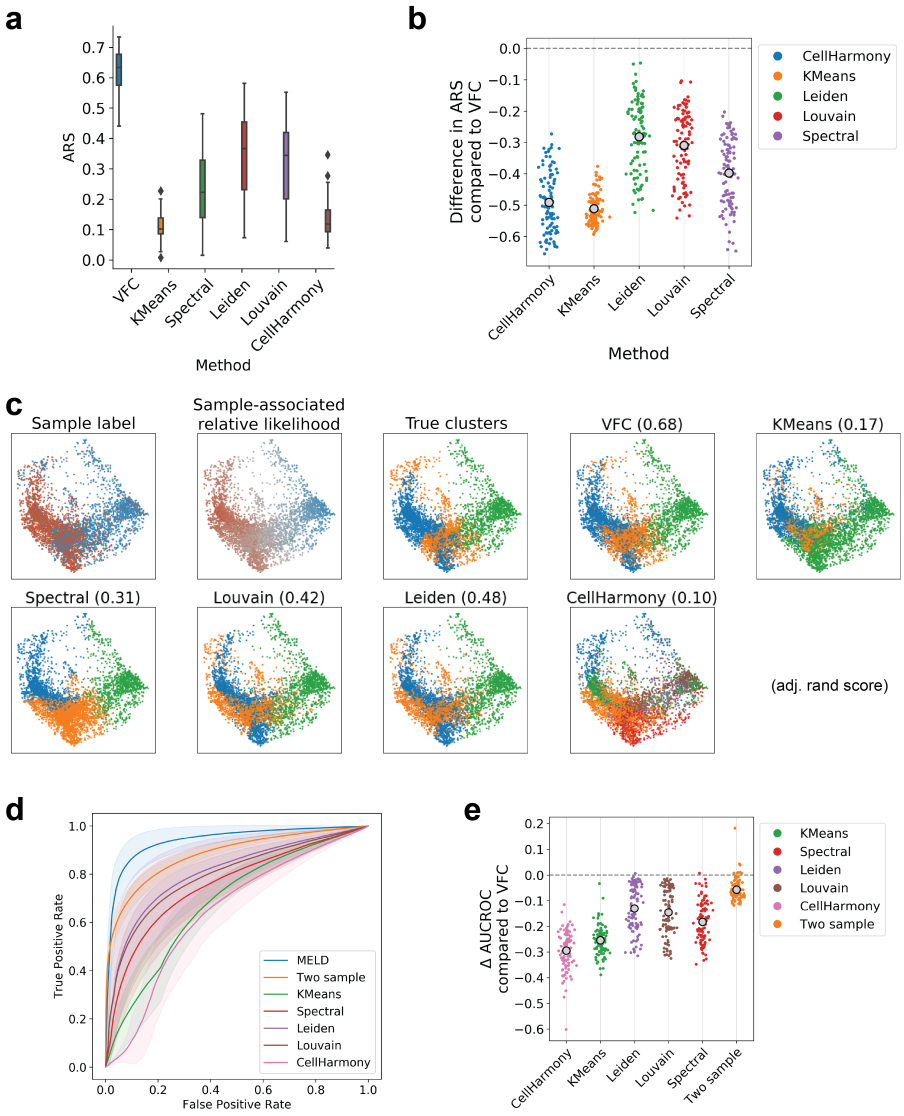

Figure 3.12: VFC accurately identifies cell populations affected by a perturbation in T cell data from Datlinger et al. [8]. (**a**) To create ground truth clusters, we artificially enriched and depleted various cell populations in either the experimental or control condition. Here we show the Adjusted Rand Score (ARS) over 100 simulations for 6 methods. For ARS, values close to 1 indicate perfect correspondence with ground truth, and values close to 0 indicate random labelling. VFC is the top performing method. (**b**) Because each simulation produced varying ARS scores for each method due to random seeds, we also consider the difference on performance between each method and VFC on each simulation. In none of 100 random seeds did any method outperform VFC. (**c**) The sample labels, sample-associated relative likelihoods, and clustering results for one randomly selected simulation. (**d**) Receiver operating characteristic (ROC) curves for the gene expression signatures described in the quantitative comparison section. The Area Under the Curve of the ROC (AUCROC) indicates the overall performance of each strategy for identifying a gene signature. MELD is the top performing approach followed by direct comparison of the two samples. (**e**) As above, we consider the difference in AUCROC over each of 100 simulations between MELD and each method. In only 4 simulations does another method outperform MELD by more than 0.01.

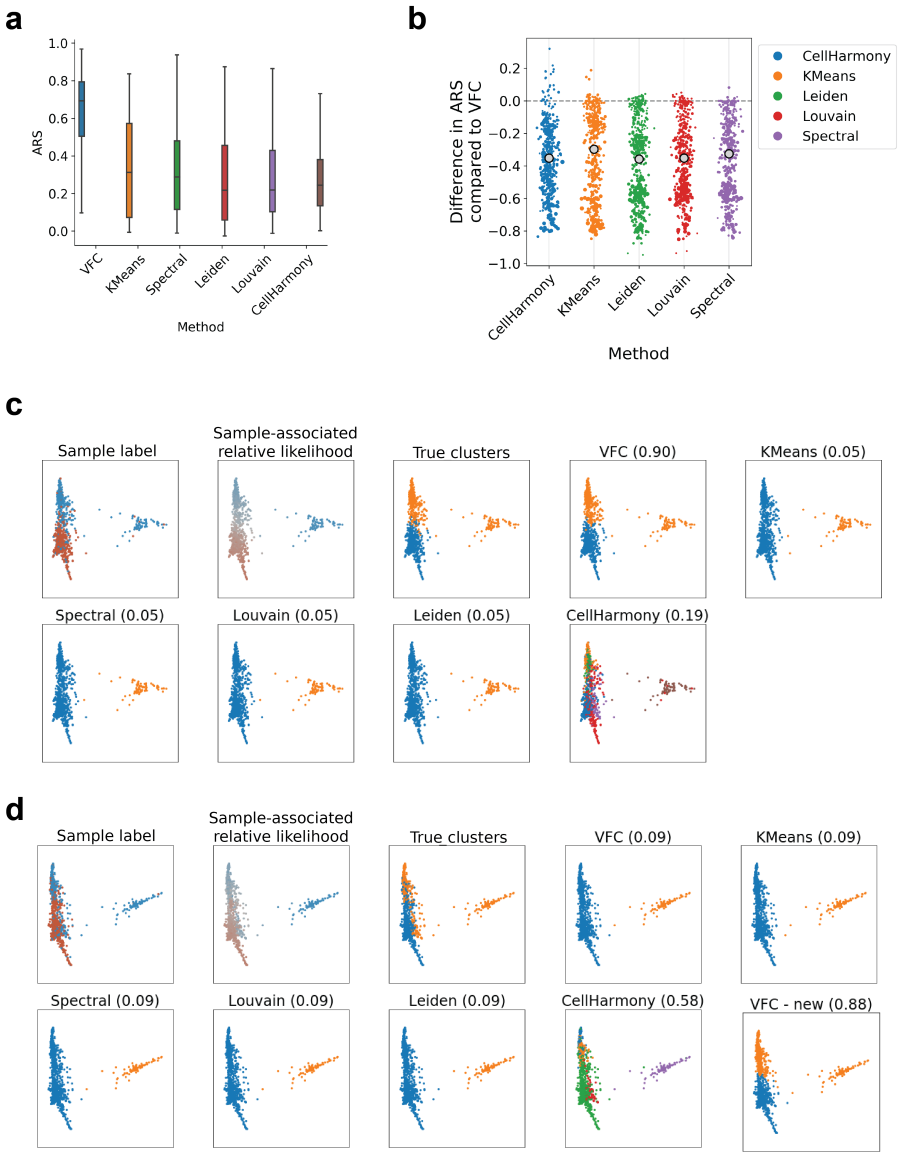

Figure 3.13: Quantitative comparison of clustering algorithms using zebrafish data from Wagner et al. [75]. (**a**) To create ground truth clusters, we artificially enriched and depleted various cell populations in either the experimental or control condition. Here we show the Adjusted Rand Score (ARS) over 100 simulations for 6 methods. VFC is the top performing method on average. (**b**) Difference on performance between each method and VFC on each simulation. (**c**) The sample labels, sample-associated relative likelihoods, and clustering results for the simulation in which VFC performed best relative to other methods and (**d**) for the simulation in which VFC performed worst relative to other methods. We found that by adjusting the weighting of the sample-associated relative likelihood from 1 (default) to 2, VFC becomes the top performing algorithm on this case ('VFC - new').

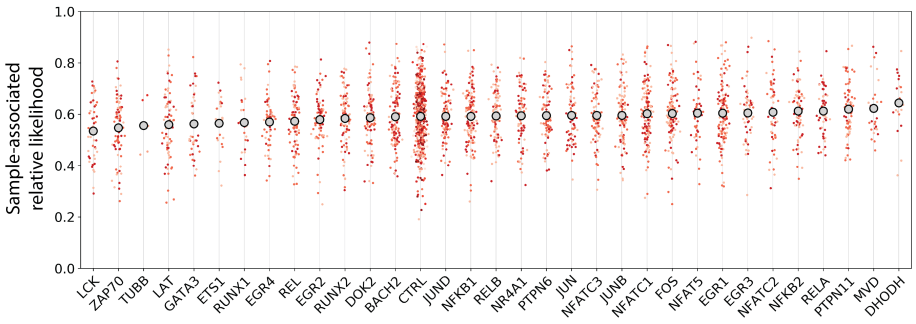

Figure 3.14: Quantitative analysis of Cas9 perturbations in T cells [8] using the MELD. Each plot shows the distribution of sample-associate relative likelihood values for all stimulated cells transfected with gRNAs targeting a specific gene. The shade of each cell indicates the different gRNAs targeting the same gene. To determine the impact of the gRNA on the TCR activation pathway, we rank each gene by the average stimulation likelihood value. We observed a large variation in the impact of each gene knockout consistent with the published results from Datlinger et al. [8]. Encouragingly, our results agree with their bulk RNA-seq validation experiment showing greatest depletion of TCR response with knockout of kinases LCK and ZAP70 and adaptor protein LAT. We also find a slight increase in stimulation likelihood values (and therefore stimulation) in cells in which negative regulators of TCR activation are knocked out, including PTPN6, PTPN11, and EGR3.

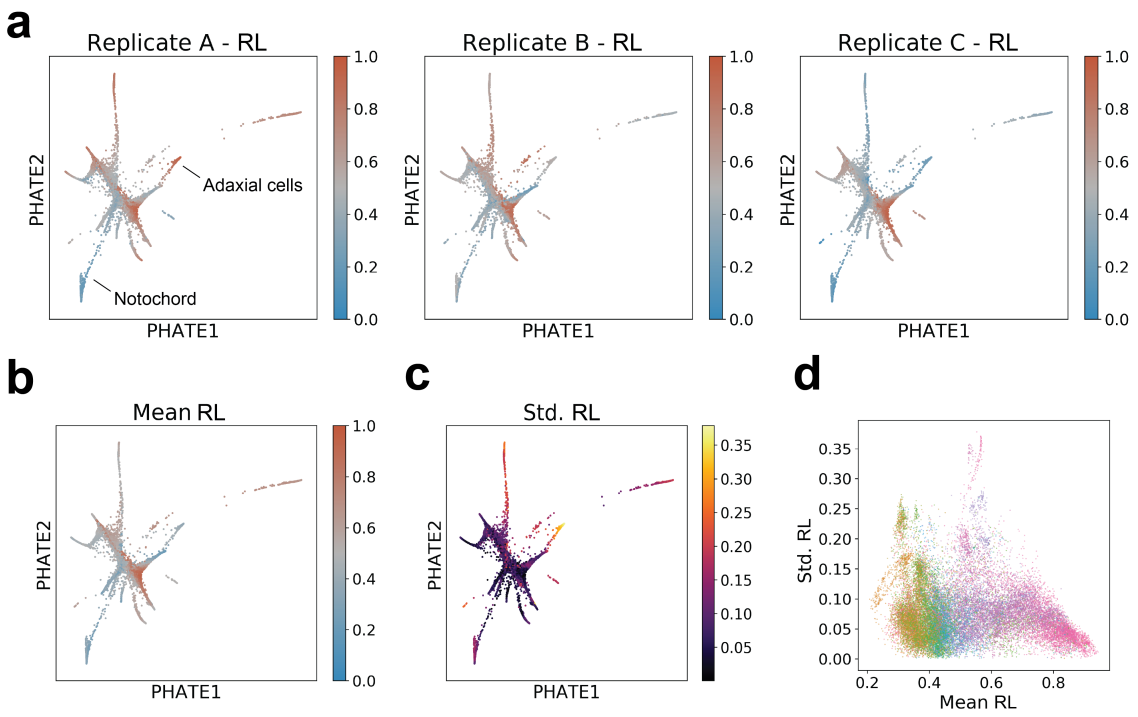

Figure 3.15: Analysis of replicates within the zebrafish data generated by Wagner et al. [75]. (**a**) Because the sample-associated relative likelihood (RL) is calculated by independently filtering a one-hot indicator vector for each condition, to calculate the chordin likelihood for each replicate, we simply row-normalize the smoothed vectors for the two signals indicating matched experimental / control pairs. For example, the "Replicate A - RL" is calculated by normalizing the "chdA" and "tyrA" filtered indicator vectors. We notice comparing replicates that the chordin likelihood for a given cell population may vary. For example, the Adaxial cell population in enriched in the Chd condition in Replicate A, but depleted in Replicate C. Similarly, cells in the Notochord population are depleted in the Chd condition in Replicates A and C, but show minimal change in abundance in Replicate B. (**b**) The average relative likelihood across all replicates is shown for each cell on a PHATE embedding. (**c**) The standard deviation of the sample-associated relative likelihood across all replicates is shown for each cell on a PHATE embedding. Regions that have higher values exhibit greater variation in their response to the experimental perturbation. We should trust the average relative likelihood values for these cells less than for cells with little variation in relative likelihood values. (**d**) A biaxial scatter plot showing the relationship between mean and standard deviation in the relative likelihood for each cell. Color indicates the cluster labels from **Figure 5a** We observe that for cells with the highest relative likelihood, the standard deviation is smaller than for cells with relative likelihood values close to 0.5 creating a slight negative Pearson correlation of -0.18.

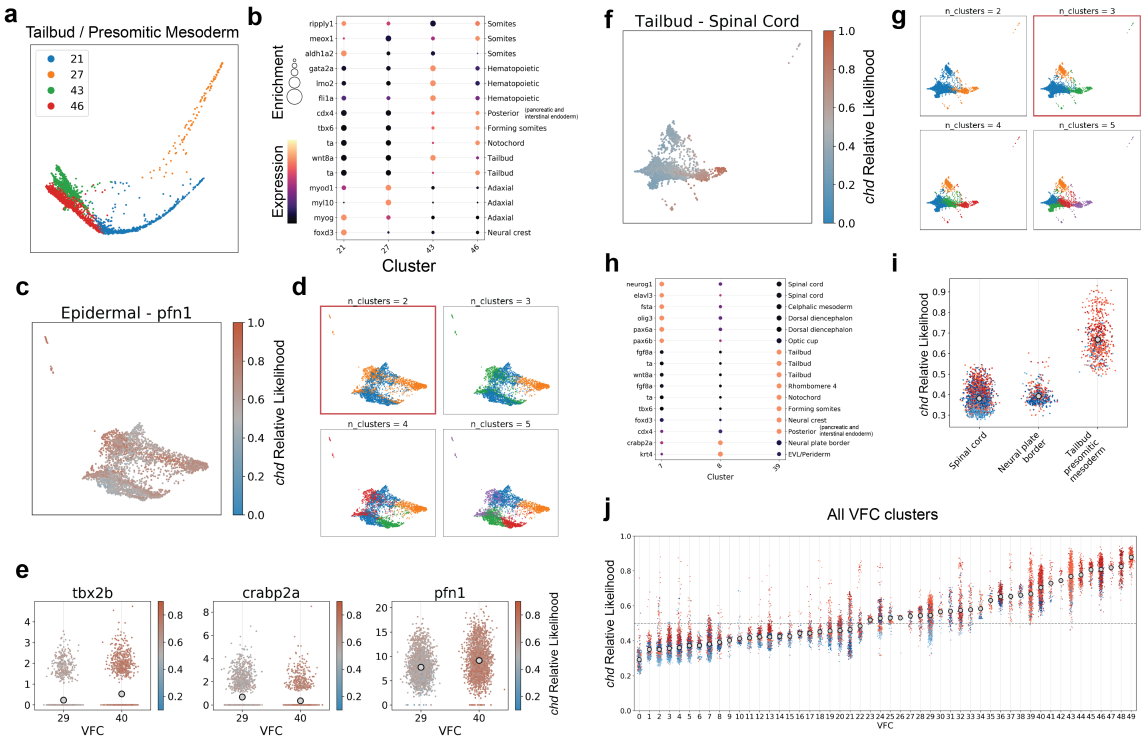

Figure 3.16: Characterization of vertex-frequency clusters in the zebrafish dataset. (**a**) Raw vertex-frequency cluster assignments on a PHATE visualization of the Tailbud - Presomitic Mesoderm cluster. (**b**) Normalized expression of previously identified marker genes of possible subtypes of the Tailbud - Presomitic Mesoderm [76]. The color of the dot for each gene in each cluster indicates the expression level and the size of the dot corresponds to the normalized Wasserstein distance between expression within cluster to all other clusters. (**c**) Distribution of chordin relative likelihood values within the "Epidermal - pfn1" cluster identified by Wagner et al. [75] shown on a PHATE plot. (**d**) Four different values of "n_clusters" that was used to create different VFC clusters with the "Epidermal - pfn1" cluster. We selected n_clusters = 2 because this identified a population of cells with similar chordin relative likelihood values and localization on the PHATE embedding. (**e**) Expression of three significantly differentially expressed genes between the two VFC subpopulations detected in the "Epidermal - pfn1" population. Tbx2b and Crabp2a were identified as markers of the epidermis and neural plate border respectively by Farrell et al. [76]. Because we observed differential expression of these two markers between the VFC subclusters suggests the "Epidermal - pfn1" cells identified by Wagner et al. [75] actually comprises cells originating from two distinct cell populations. (**f**) Distribution of chordin relative likelihood values within the "Tailbud - Spinal Cord" cluster identified by Wagner et al. [75] shown on a PHATE plot. (**g**) Four different values of n_clusters that was used to create different VFC clusters within the "Tailbud - Spinal Cord" cluster. We selected n_clusters = 3 because this identified populations of cells with similar likelihood values and localization on the PHATE embedding. (**h**) Same plot as in (**b**) for the subclusters of the "Tailbud - Spinal Cord". (**i**) Distribution of relative likelihood values within each VFC subcluster show that the three subclusters are biologically distinct with differing responses to the experimental perturbation. (**j**) Repeating the VFC subclustering process for all cells, we identified a total of 50 clusters within the zebrafish dataset generated by Wagner et al. [75]. Compared to the plot in **Figure 5b**, we observed a more restricted distribution of chordin relative likelihood values within each cluster suggesting these labels represent populations of cells that are more homogeneous with respect to the experimental perturbation.

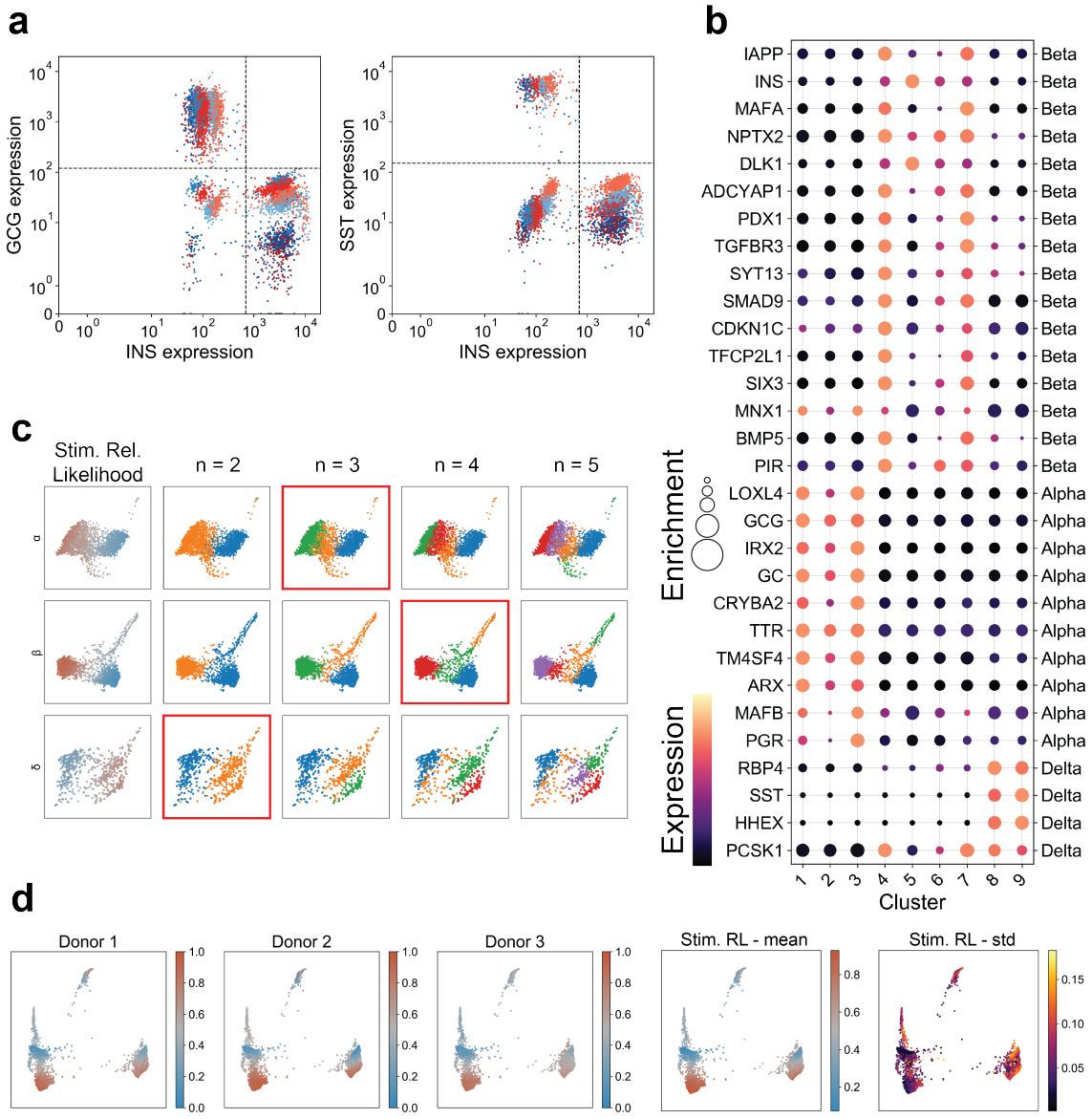

Figure 3.17: Analysis of pancreatic islet cells from three donors. (**a**) Library-size normalized expression of insulin (INS), glucagon (GCG), and somatostatin (SST) shows donor-specific batch effect across islet cells. (**b**) Normalized expression of previously identified marker genes of alpha, beta, and delta cells[89] in each cluster. The color of the dot for each gene in each cluster indicates the expression level after MAGIC and the size of the dot corresponds to the normalized Wasserstein distance between expression within cluster to all other clusters. (**c**) Results of VFC using varying numbers of clusters for each of the three cell types. The red box denotes the selected level of clustering for each cell type. (**d**) The sample-associated relative likelihood is calculated independently for each donor and then averaged to obtain the stimulated relative likelihood used in the main analysis. We also calculate the standard deviation of the relative likelihood for each cell.

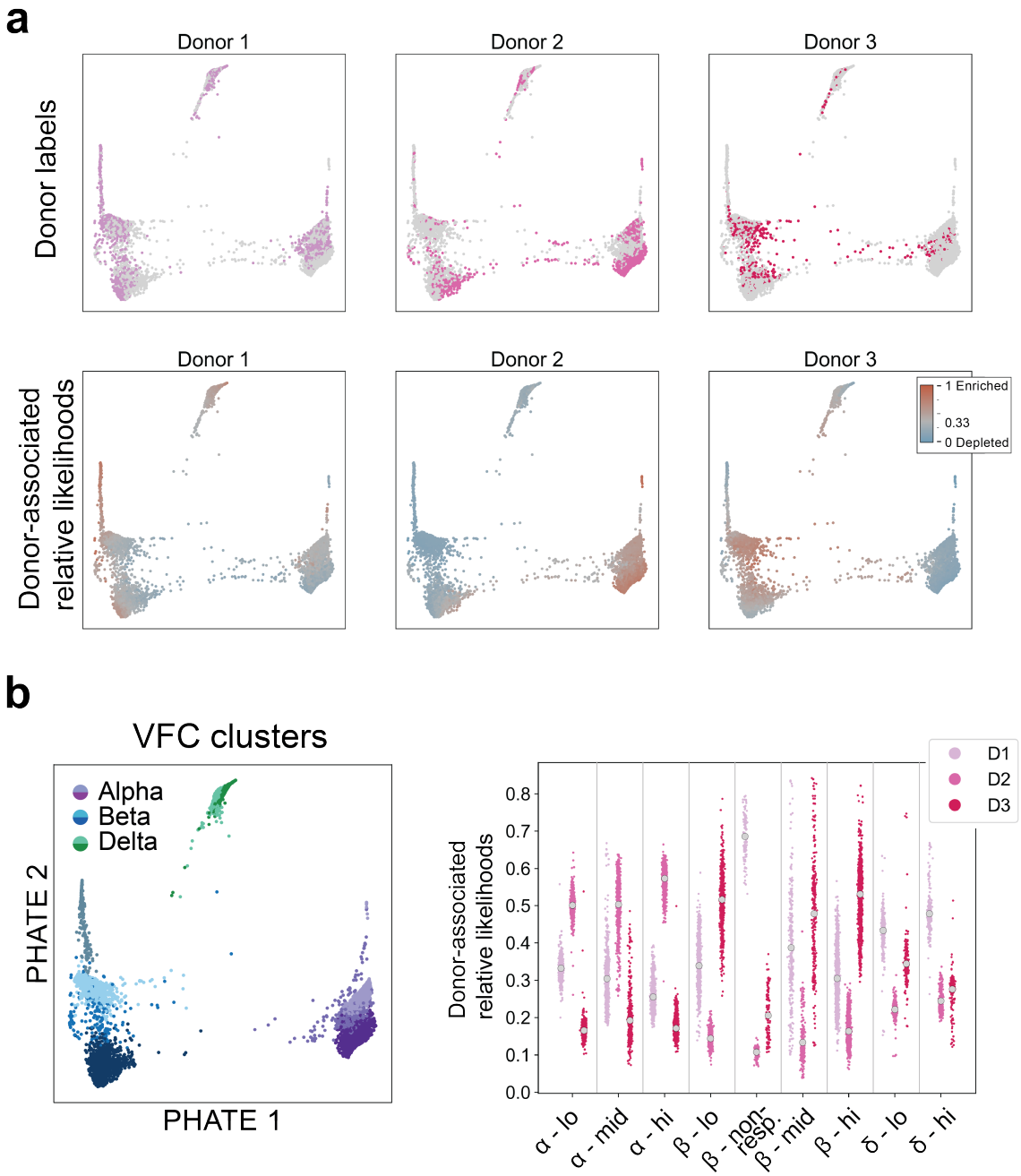

Figure 3.18: Analysis of islet cell profiles across donors. (**a**) The sample labels and sample-associated relative likelihood associated with each donor from which islet cells were obtained. (**b**) Comparison of the donor likelihood values within each vertex frequency cluster identifies changes in enrichment for each cluster in various donors. For example, the β - non-responsive cluster is strongly enriched in donor 1.

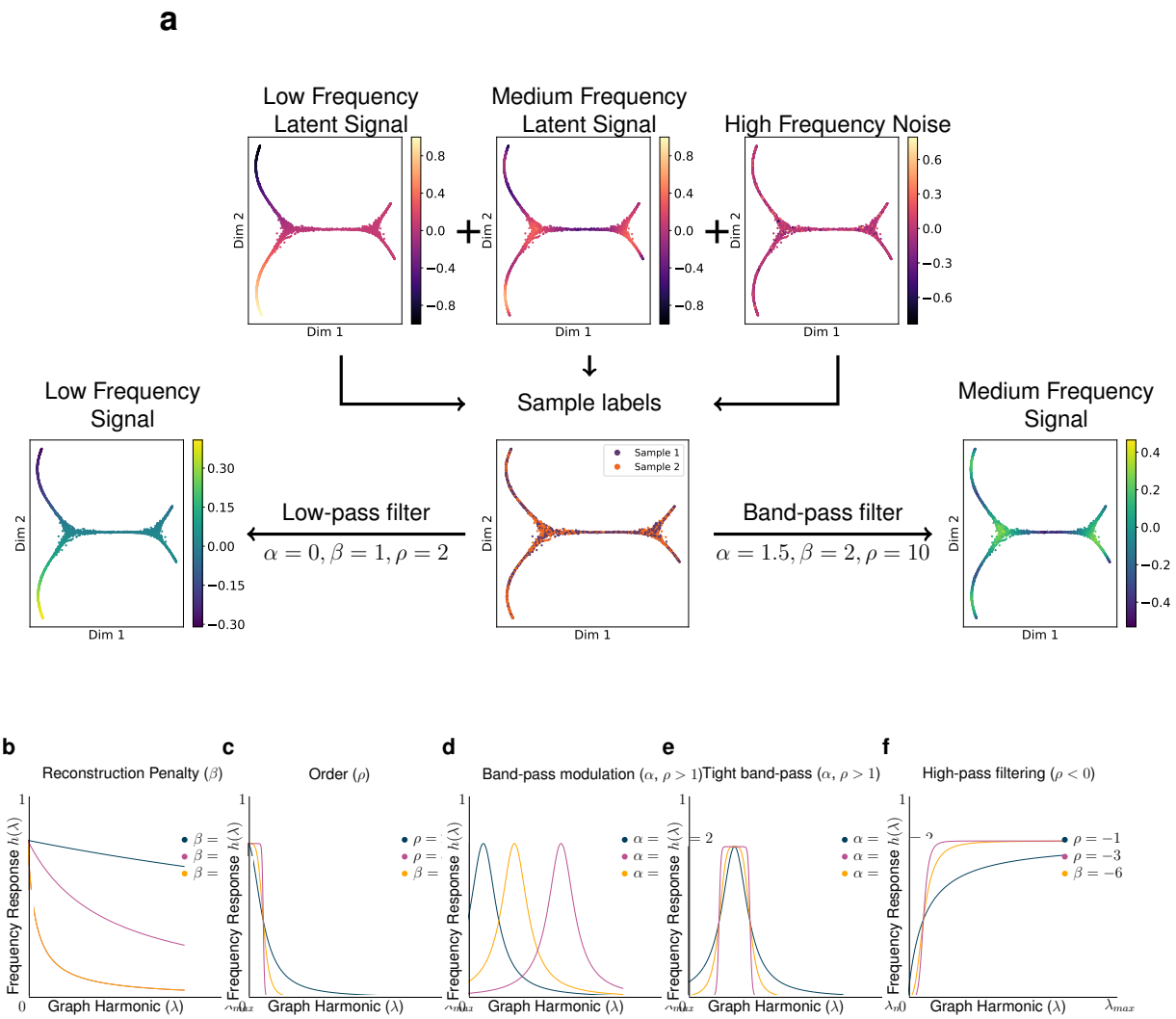

Figure 3.19: Source Separation and Parameter Analysis with the MELD filter. (**a**) Sample labels (center) are obtained that are a binarized observation of a low frequency latent signal (top left), a medium frequency latent signal (top middle), and high frequency noise (top right). Analysis of the sample labels alone is intractable as they are corrupted by noise and experimental binarization. MELD low-pass filters (bottom left) to separate a longitudinal trajectory and band-pass filters (bottom right) to yield the periodic signature of the medium frequency latent signal. Parameters used for this analysis are supplied beneath the corresponding arrows and the laplacian filter is used for illustrative purposes. (**b**) Reconstruction penalty β controls a low-pass filter. For this demonstration, $\alpha = 0, \rho = 1$. This filter is equivalent to Laplacian regularization. (**c**) Order ρ controls the filter squareness. This parameter is used in the low-pass filter of (**a**). For this demonstration, $\beta = 1, \alpha = 0$. (**d**) Band-pass modulation via α. When ρ is even valued, α modulates the central frequency of a band-pass filter. This parameter is used in (**a**) to separate a medium-frequency source from a low-frequency source. (**e**) α and ρ combine to make square band-pass filters. For (**d**) and (**e**), $\beta = 1$. (**f**) Negative values of ρ yield a high-pass filter. For (**b-f**), Laplacian harmonics for a general normalized Laplacian are plotted on the x-axis. The frequency response of the filter given by the colored parameters is on the y-axis.

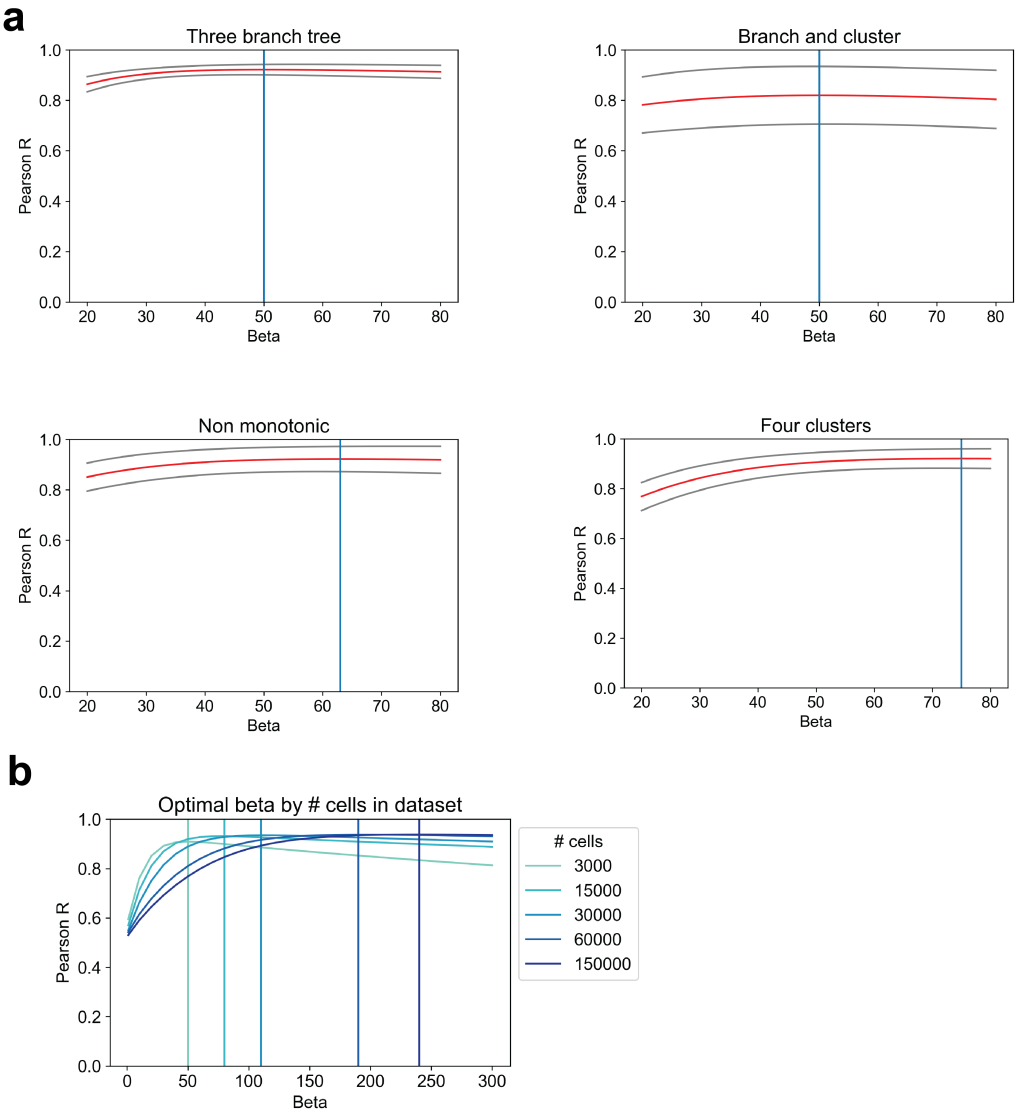

Figure 3.20: Selecting parameters for MELD. (**a**) Results of a parameter search over the β parameter using the four datasets described in the quantative comparisons section. The red line shows the average performance over 10 different datasets of each geometry with one standard deviation marked by the grey lines. We observe reasonably consistent performance of the sample-associated relative likelihood algorithm across all datasets using a β value between 50-75. We chose a value of 60 as the default in the MELD package and used this setting for all experiments. (**b**) We observe that the optimal β parameter for a dataset varies with the number of cells in the dataset. We suggest increasing the default beta parameter for datasets larger than 30,000 cells.

Dataset	Rel. Likelihood	Graph Averaging	kNN Averaging
Branch and Cluster	**0.82 (0.05)**	0.41 (0.05)	0.73 (0.04)
Non-monotonic	**0.94 (0.03)**	0.52 (0.06)	0.85 (0.03)
Four clusters	**0.91 (0.06)**	0.44 (0.07)	0.76 (0.07)
Three Branches	**0.90 (0.03)**	0.48 (0.07)	0.73 (0.07)
T cells [8]	**0.98 (0.01)**	0.72 (0.06)	0.32 (0.04)
Zebrafish [75]	**0.98 (0.01)**	0.53 (0.07)	0.80 (0.07)

Table 3.1: Quantitative comparison of methods for label smoothing over a graph. 40 random seeds were used for each of 4 synthetic datasets. 100 random seeds were used to create sample assignments on the T cell and zebrafish datasets. Average Pearson Correlation with ground truth signal is displayed with standard deviation in parentheses. Top performing algorithm is bolded.

Dataset	VFC	Spectral	Louvain	Leiden	KMeans	CellHarmony
T cell [8]	**0.62 (0.07)**	0.23 (0.11)	0.31 (0.13)	0.34 (0.14)	0.11 (0.04)	0.13 (0.05)
Zebrafish [75]	**0.53 (0.31)**	0.13 (0.15)	0.23 (0.22)	0.19 (0.21)	0.23 (0.20)	0.22 (0.16)

Table 3.2: Quantitative comparison of clustering methods to identify the cell types affected by a simulated experimental perturbation using real world data.

Chapter 4

Finding Archetypal Spaces Using Neural Networks

Authors

Daniel B. Burkhardt[1*], David van Dijk[2*], Matthew Amodio[3], Alexander Tong[3], Guy Wolf[4†], Smita Krishnaswamy[1,3†]

[1]Department of Genetics; [2]Department of Internal Medicine (Cardiology);

[3]Department of Computer Science;

Yale University, New Haven, CT, USA

[4]Department of Mathematics and Statistics, Université de Montréal, Montreal, QC, Canada

*, † These authors contributed equally

Abstract

Archetypal analysis is a data decomposition method that describes each observation in a dataset as a convex combination of "pure types" or archetypes. These archetypes represent extrema of a data space in which there is a trade-off between features, such as in biology where different combinations of traits provide optimal fitness for different environments. Existing methods for archetypal analysis work well when a linear relationship exists be-

tween the feature space and the archetypal space. However, such methods are not applicable to systems where the feature space is generated non-linearly from the combination of archetypes, such as in biological systems or image transformations. Here, we propose a reformulation of the problem such that the goal is to learn a non-linear transformation of the data into a latent archetypal space. To solve this problem, we introduce Archetypal Analysis network (AAnet), which is a deep neural network framework for learning and generating from a latent archetypal representation of data. We demonstrate state-of-the-art recovery of ground-truth archetypes in non-linear data domains, show AAnet can generate from data geometry rather than from data density, and use AAnet to identify biologically meaningful archetypes in single-cell gene expression data.

Contribution

The use of an autoencoder for archetypal analysis was initially proposed by Dr. David van Dijk. I proposed the reformulation of archetypal analysis as learning a transformation from the ambient space to a latent archetypal space bound by a simplex. I designed the sphere projection, dSprites, reproducibility, scRNA-seq, and gut microbiome experiments. I wrote a majority of the text and designed the figures.

4.1 Previous work and Background

The first algorithm proposed for archetypal analysis was principal convex hull analysis (PCHA) as described by [50], which identifies a set of p archetypes constrained to be linear combinations of the data such that the following is minimized:

$$\min_{\mathbf{W},\mathbf{H}} ||\mathbf{X}' - \mathbf{X}'\mathbf{W}\mathbf{H}||_F^2 \tag{4.1}$$

Here, $\mathbf{X}$ is the data matrix with n observations on the rows and m features. $\mathbf{W}$ is an $n \times p$ matrix mapping the data to the archetypes and $\mathbf{W}$ is a $p \times n$ matrix denoting the archetypes in the feature space. Cutler and Breiman [50] then propose an optimization algorithm using alternating least squares.

Subsequent advances focused on improvements to the algorithm for fitting a hull to the data. In [131], it is proposed to solve the PCHA optimization via projected gradient descent. Further improvement to the optimization procedures are formed in [132], which uses an active set strategy. More recently, envelope constraints were tightened in [133] by adding a cost for the sum of the distances of the data points from the convex envelope of the archetypes and another for the sum of the distances of archetypes from the convex envelope of the data points.

The first work to propose AA on a transformed feature space is [131]. There, an algorithm is provided for AA applied to the kernel space of a dataset. In [134], the authors perform archetypal analysis on the representation found in a hidden layer of an image classification neural network in order to define image styles. Although these methods extend AA to non-linear feature spaces, both apply a fixed transformation to the data space. By contrast, our goal is to *find* an optimal non-linear transformation of the data such that the data is optimally described by a simplex. We propose to use a novel neural network regularization for this task.

4.2 Introduction

Archetypal analysis (AA) decomposes each observation in a dataset into a convex combination of pure types or *archetypes*. These archetypes represent extreme combinations of features and thus are extrema of the data space. For example, species adapted to specific environments will have unique and extremal combinations of features [135]. Since each observation is described as a mixture of the archetypes, AA describes the dataset as

varying smoothly between the identified archetypes. This interpretation has several applications for exploratory data analysis. For example, the archetypes can be characterized in the feature space to understand the extrema of a dataset. Additionally, when considering the *archetypal space*, *i.e.* the mixture of archetypes for each data point, AA provides a new factor space for data exploration. A point can now be characterized by its composition of specific archetypes, and distances between points can be calculated from archetypal mixtures. These applications have led to the application of AA for exploratory data analysis in a number of disciplines including astronomy [136], market research [137, 138], document analysis [139, 140], and genomic inference [141, 51, 142].

Because each point is represented as a convex combination of archetypes, there is an inherent trade-off between the archetypes. This limits the number of archetypes identifiable in $\mathbb{R}^n$ to $n + 1$. It is not possible to fit four archetypes to a rectangle in $\mathbb{R}^2$. This constraint well fits systems with an inherent trade-off between features, such as in genomics where typically only relative abundances of genes are considered [142]. In this way, AA bears similarity to Latent Dirichlet Allocation (LDA), a statistical method used for topic analysis that models word occurrences in a document as occurring with some probability over a discrete number of topics with a Dirichlet prior [143]. Thus, the latent features in LDA also form a space bound by a simplex. However in LDA, the topics are known *a priori*, and the goal of AA is to identify the archetypes. Finally, AA implies a data model where each point varies continuously between a set of archetypes, unlike the model of clustering methods where data originates from centroids plus noise. For such cluster-like data sets, AA would need to be applied to each cluster independently.

Identifying archetypes is the primary challenge in AA. Most methods for AA identify archetypes by fitting a simplex to the data space where the vertices are linear combinations of the input data. A limitation of this approach is that if the relationships between features in the dataset are non-linear, then the extrema of the data space may not correspond to the extrema of the data geometry. Take, for example, a triangle projected onto a sphere.

Although the vertices of the triangle remain the extrema of the data geometry, they may no longer conform to extrema of the data space (Fig. 4.2). In this case, linear AA methods fail to capture correct archetypes as shown in Section 4.4.1. Non-linear AA methods have been proposed, such as kernel PCHA [131]. However, in these methods a fixed non-linear transformation is applied to the data after which linear AA is performed. There is no guarantee that any one transformation makes all data sets well-approximated by a simplex.

To overcome these limitations, we propose a new formulation of the problem. Instead of fitting a convex hull to a fixed feature space, our goal is to identify a transformation of feature space $\mathbf{X}$ into an k-dimensional archetypal space where k corresponds to the number of archetypes. In the archetypal space, $\mathbf{Z}$, single activations of each dimension correspond to archetypes (*i.e.* [1,0,0] for a space with 3 archetypes). The space is constrained such that each data point is represented as a convex combination of the archetypes. Because of the convexity constraint, all observations are bound by a k-dimensional simplex. In this reformulation, the goal of AA is to learn the ideal transformation $f(\mathbf{X}) \rightarrow \mathbf{Z}$ and inverse function $f'(\mathbf{Z}) \rightarrow \mathbf{X}$ such that the underlying data geometry is preserved.

To achieve this, we introduce the Archetypal Analysis network (AAnet), a neural network framework for learning and generating from a latent archetypal space. AAnet uses an autoencoder with a novel regularization on the latent layer in which the encoder E learns the transformation from the data space (input) to the archetypal space (bottleneck layer), and the decoder D learns the transformation back to the feature space (reconstruction). Performing AA in this manner also provides powerful generative properties. Single activations of each node in the latent space represent an archetype of the data that the decoder transforms back to the feature space. It is also possible to generate new data with a specific mixture of each archetype. In contrast, the latent space of generative models such as the VAE or the sampling space of a GAN have no accessible semantic structure from which to generate data as a mixture of "pure types". Furthermore, AAnet can sample from the data

geometry independent of data density, which are limitations of VAEs and GANs.

The main contributions of this chapter are:

1. A reformulation of archetypal analysis with the goal of learning an optimal transformation of the data in the feature space into an archetypal space bound by a simplex;

2. A novel regularization on the latent space of an autoencoder such that nodes of the bottleneck layer are archetypes and node activations are loadings of the data onto the archetypes;

3. Demonstration of the generative properties of AAnet on unevenly sampled data with comparisons to a VAE and GAN; and

4. An extensive collection of quantitative benchmarks comparing AAnet against five state-of-the-art archetypal analysis methods.

The remainder of the paper provides a summary of previous work, description of the AAnet framework and implementation, quantitative comparisons of AAnet to existing AA methods on synthetic datasets, application of AAnet to a new single-cell gene expression dataset, and demonstrations of the reproducibility, robustness, and scalability of AAnet.

4.3 Methods

First, we describe our new generalized problem formulation for finding a transformed data space for archetypal analysis, and then we describe our AAnet framework.

4.3.1 Problem setup

Our problem formulation is a generalization of the formulation in Equation 4.1. Instead of the archetypes learned as a linear combination of the original data points, we optimize over

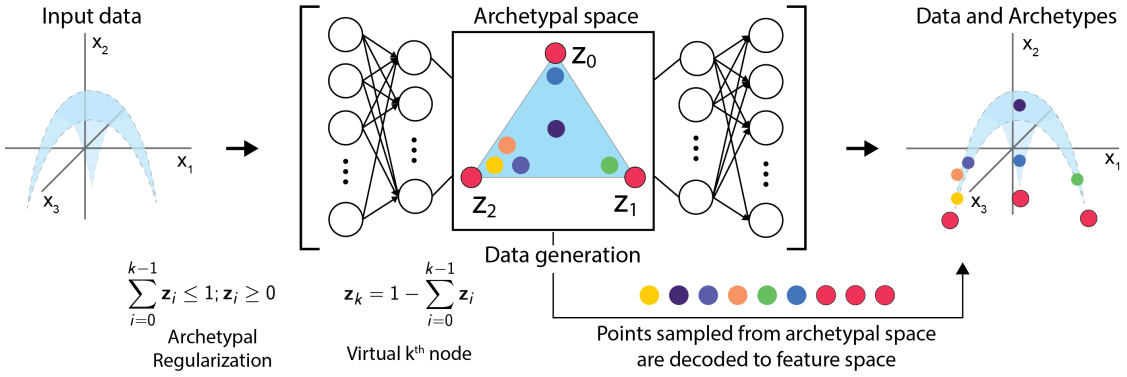

Figure 4.1: Illustrative representation of AAnet. AAnet learns a non-linear transformation of the input data (blue) such that within the embedding layer, the data fits well within a simplex whose vertices (red dots) represent extreme states of the data, also called archetypes. By decoding the points in the latent space, AAnet can be used for exploratory data analysis and data generation.

a general nonlinear transformation $f(X)$ from the feature space to an archetypal space in which the convex constraints are enforced.

The generalized archetypal analysis problem is the following optimization:

$$\begin{aligned} \underset{f, c_1, \ldots, c_k}{\arg\min} \quad & \sum_{i=1}^{n} \|f(x_i) - \sum_{j=1}^{n} \alpha_{ij} c_j\|^2 \\ \text{subject to} \quad & f \text{ is approximately invertible on } X \\ & \sum_{j=1}^{k} \alpha_{ij} = 1, \quad i = 1, \ldots, n \\ & \alpha_{ij} \geq 0, \quad i = 1, \ldots, n, \; j = 1, \ldots, k \end{aligned} \tag{4.2}$$

The inclusion of f in the optimization is unique to our formulation, while previous methods either considered no transformation (i.e., f = identity), or apply a fixed transformation during preprocessing (e.g. kernel PCHA). We note that our requirement that f be approximately invertible is added here to allow the mapping of archetypes $\{c_j\}_{j=1}^{d}$ and hypothetical (convex) combinations of them to the original feature space.

4.3.2 The AAnet Framework

We propose a deep learning approach for solving the optimization problem in Eq. 4.2, by considering f as the output of a neural network we called AAnet (*Archetypal Analysis*

network) (see Fig. 4.1). To consider the approximate invertibility constraint, we base our network on an autoencoder, where the encoder $E(x)$ yields the transformation f, and the decoder $D(x)$ yields its (approximate) inverse. Then, the convex combination constraint is ensured by a novel regularization that we term *archetypal regularization*. This regularization constrains the activations in that layer to be coefficients of the archetypal decomposition of a data point in the latent space of the neural network, and thus the archetypes themselves are naturally represented by one-hot vectors in this space.

Formally, our network is formed by an encoder $z = E(x)$ and decoder $\tilde{x} = D(z)$, with the main MSE reconstruction loss: $\text{MSE} = \mathbb{E}_{x \in X} \left[\|x - \tilde{x}\|^2 \right] = \mathbb{E}_{x \in X} \left[\|x = D(E(x)\|^2 \right]$.

Then, to enforce k archetypes, we expect z to provide us with k activations that sum up to one. However, notice that given such equality, we can directly compute $\alpha_k = 1 - \sum_{j=1}^{k-1} \alpha_j$. Hence, we set the embedding layer in our network to have $k-1$ nodes computed from the encoder layers, which we denote by $E'(x) \in \mathbb{R}^{k-1}$ and an additional virtual node yielding $z = E(x) = [E'(x),\ 1 - \|E'(x)\|_1]$.

The described encoder architecture choice allows us to relax the unit-equality constraint to an inequality constraint, which is more suitable for the optimization used in neural network training. Therefore, our archetypal regularization is formulated as two soft constraints:

$$\|E'(x)\|_1 \leq 1 \text{ and } E'(x_i) \geq 0, i = 1, \ldots, n \tag{4.3}$$

for every $x \in X$, which ensures the embedding layer provides convex combinations of k archetypes given by the k one-hot vectors of $\mathbb{R}^k$. Note, the requirement of data points being well represented by these archetypes is implicitly enforced by the MSE reconstruction loss. The final network loss is then given by reconstruction loss + two archetypal regularizations. Thus, the encoder learns a transformation that represents the data in the bounds of a convex hull, and the decode enforces accuracy of the learned representation. See Fig. 4.1 for a diagram of AAnet.

Latent noise for tight archetypes

By default, AAnet can find archetypes outside the data. However, to encourage the archetypes to be tight, *i.e.* close to the data, we can add Gaussian noise $\sim N(0, \sigma)$ in the latent layer during training. Adding noise has an effect of spreading the data out in the latent space, since the autoencoder has to reconstruct points despite the noise. This, in turn, has the effect of bringing the archetypes closer to the data. We show this effect in Fig. 4.6 where we add increasing amounts of latent noise and plot the latent archetypal space with the data and the archetypes. Finally, we note that the noise here is analogous to the δ parameter in [131], which controls the distance of the archetypes to the data. By default, we set sigma such that the archetypes are close to but not significantly inside the data. In practice we set sigma such that only around 0.1 percent of the data points are outside the convex hull. For all experiments in this chapter, this was achieved with a σ of 0.05.

Geometry based data generation

To generate new data using AAnet, we can sample arbitrary convex activations of the latent space and decode them to the feature space. Since this convex hull represents the boundary of the data geometry, this method allows us to sample directly from the geometry and independently of the input data distribution. For example, we can sample uniformly from the data geometry by sampling uniformly from a simplex and decode these points to the data space. Uniform sampling from a simplex was achieved by sampling from a Dirichlet distribution and then normalizing: $S_{ij} = \frac{-\log(U_{ij})}{\sum_{k=1}^{n_{at}} -\log(U_{ik})},\ i = 1, \ldots, n,\ j = 1, \ldots, n_{at}$, where U is an $n \times n_{\text{at}}$ matrix whose elements are positive and i.i.d. uniformly distributed, n is the number of data points and n_{at} the number of latent archetypes. The resulting matrix S is uniformly sampled on a simplex with n_{at} corners. Finally, we get the generated data via $\hat{x} = D(S)$, where $\hat{x}$ is the generated data and D is the decoder.

4.3.3 Code availability

Code and a tutorial for AAnet is publicly available on GitHub at `https://github.com/KrishnaswamyLab/AAnet`. This repository also includes scripts to run the quantitative comparisons included in this chapter and to reproduce the dSprites image translation experiment.

4.4 Results

Here we evaluate the accuracy and performance of AAnet in finding archetypes in ground-truth non-linear data with defined archetypes. We demonstrate that AAnet recovers interpretable archetypes in benchmark data from machine learning and in a biological dataset. We compare AAnet to 5 other methods. These include three linear archetypal analysis methods: [131] (i.e. PCHA), [133], and [132] as well as two non-linear AA methods: kernel PCHA [131] and PCHA on the latent layer of a neural network [134]. For [134] we exchanged the classifier framework for an autoencoder and refer to the method as "PCHA on AE". We did this modification in order to be able to decode back to the data space, which is required for quantifying the performance of the methods, and because most of our data did not have labels. Full parameter details for AAnet are reported in Section 4.6.1 and details of methods used for comparison are reported in Section 4.6.2.

4.4.1 Archetypes from a triangle projected onto a sphere

To test the ability of AAnet to find archetypes in non-linear data, we uniformly sampled 2000 points on a triangle and projected the data onto a sphere with radius R. To create increasing curvature on the projected triangle, we gradually decreased R from 1000 to 0.75. We then ran AAnet as well as the other methods on this generated data and quantified how well each method performs by computing the MSE between the ground truth archetypes

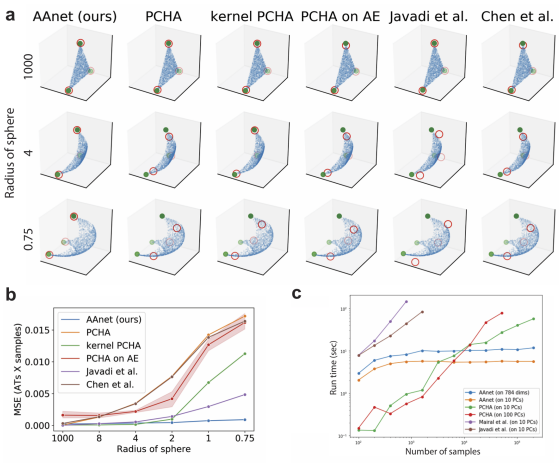

Figure 4.2: (**a**) Points uniformly distributed within a triangle (blue dots) are projected onto a sphere of varying radius (columns). 6 AA methods are compared on their ability to recover the vertices of the triangle (green dots) and learn the correct mixture of archetypes for each point. Red circles mark the recovered archetypes. Right, the MSE between ground truth and recovered archetypal spaces (**b**) and recovered archetypes (**c**) are displayed for each method. Shaded area marks 95% CI over 5 runs.

with which the data was generated and the archetypes inferred by each method (ATs x features). We also computed MSE between the recovered archetypal mixtures and the ground truth mixtures (ATs x samples). We find that with low levels of curvature all methods perform well and are able to find the correct archetypes (Fig. 4.2). However, when increasing the curvature (by decreasing the radius of the sphere) all methods other than AAnet break down, with AAnet being the only method that consistently finds the right archetypes.

4.4.2 Finding archetypes of image translations

We compared the same set of methods on the dSprites dataset, which was designed as a benchmark for disentanglement in unsupervised learning [144]. The dataset consists of three image classes: rectangles, ovals, and hearts. Each class of images varies by 6 independent latent factors: horizontal and vertical offset, rotation, scale, and color. Disentanglement shares an intuitive relationship with AA, because each archetype should correspond to an extreme combination of the latent features of the dataset. Finally, although the transformations are affine in the image space, they are non-linear in the Euclidean pixel

space.

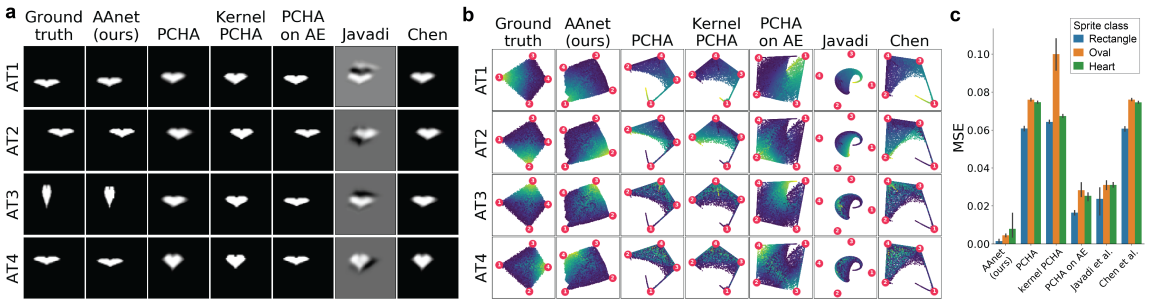

Figure 4.3: Comparison of AA methods on dSprites dataset. (**a**) Ground truth and recovered archetypal hearts. (**b**) Ground truth and recovered archetypal space visualized using the same test set as in **a**. Points are colored by the ground truth loading of each archetype. (**c**) Quantitative comparison of the archetypal space recovered by each method. Each method was run on 5 samples of 15,000 images using each class in the dSprites dataset. Error bars denote 95% confidence intervals over the 5 runs.

To generate images for our comparison, we uniformly sampled points from a four-dimensional simplex. These values were used to adjust the horizontal offset, vertical offset, and aspect ratio for each sprite using scikit-image [145]. Each method was run on 5 different samples of 15,000 images for each sprite. Representative archetypes recovered from each method can be seen in Fig. 4.3a and the archetypal spaces learned for this same batch are visualized in Fig. 4.3b. A full description of the visualization algorithm can be found in Section 4.4.8. To quantify the accuracy of each method, we only considered the MSE between the learned and ground truth archetypal spaces (Fig. 4.3c) because euclidean distances between images are not meaningful. We found that AAnet performed best overall, outperforming the second best method, PCHA on AE, by 80% on average. Example images of input data and visualization of archetypes and archetypal spaces for the ovals and hearts can be found in Fig. 4.10.

4.4.3 Generating from the data geometry with AAnet

Next, we investigate the ability of AAnet to generate data independently of the input data density. The simplex learned by AAnet in the latent space represents the boundary of a non-linear manifold or the geometry of the data. We can sample arbitrary convex com-

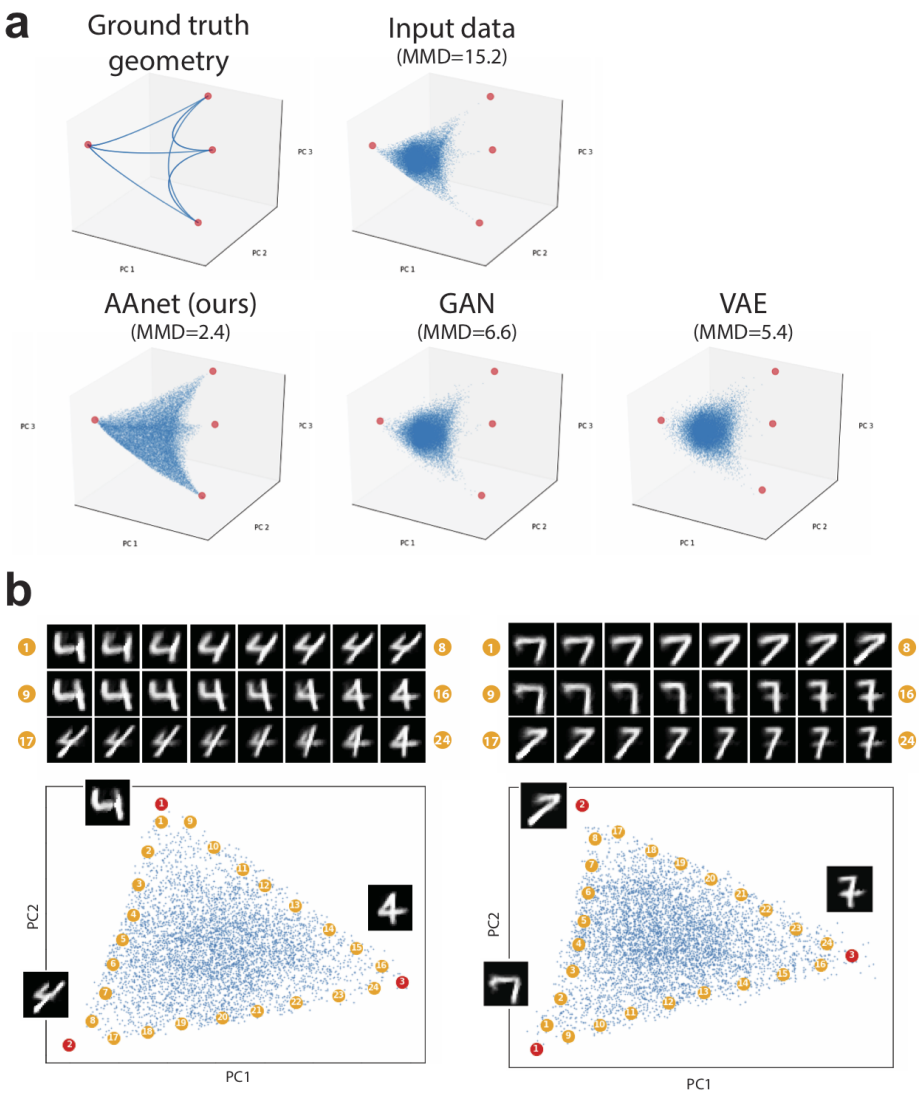

Figure 4.4: (**a**) Above, the ground truth data geometry and non-uniformly sampled input data. Below, data generated from AAnet, a GAN, and a VAE. MMD quantifies the discrepancy between each method and the ground truth geometry. (**b**) We uniformly sample trajectories between two archetypes from the AAnet. Shown above are such trajectories for MNIST 4s and 7s between all pairs of archetypes. Below, visualization of the archetypal space for the input data (blue), archetypes (red) with images, and sampled points (yellow).

binations of the latent space to generate data based on data geometry rather than the data density. Thus, even if the training data is non-uniformly distributed, we can learn its geometry and then sample uniformly from this geometry and decode the sample points back to the feature space.

To test this, we generated a non-linear geometry with four archetypal points embedded in 100 dimensions, as shown in Fig. 4.4a. We then sampled data non-uniformly (prefer-

entially from the center) and trained AAnet, a GAN [146], and a VAE [147] on this data. GANs and VAEs are generative models and are thus able to generate samples in the data space by sampling in their latent spaces. We then sampled from the latent spaces of these three models to generate points in the data space. The GAN and VAE both generate based on the data density, while AAnet can generate from the geometry by sampling uniformly from a simplex in its latent space (Section 4.3.2). To quantify the ability of each model to generate from the geometry, we computed a Maximum Mean Discrepancy (MMD) [148] (using a multiscale Gaussian kernel) between the ground truth geometry and the input data, the data generated by AAnet, the data generated by the GAN, and the data generated by the VAE. AAnet had the lowest discrepancy between the generated data and the ground truth geometry performing 56% and 64% better than the VAE and GAN, respectively.

To demonstrate that the latent space of AAnet provides semantic structure for data generation, we sampled images by interpolating between pairs of archetypes in the latent space of AAnet trained on MNIST digits. Fig. 4.4b shows this for MNIST 4s and 7s. The generated images do not appear in the training data, yet we observe gradual and meaningful transitions between them. Each interpolated image looks like a convex combination of its two corresponding archetypes.

4.4.4 AAnet identifies reproducible archetypes

To show that AAnet can identify robust, reproducible archetypes, we generated archetypes for each MNIST digit 50 times using different random seeds. A subset of these images are shown in Fig. 4.5a. We then calculated r^2 between archetypes identified on subsequent runs of AAnet and random MNIST images of the same digit. For all digits, we notice a significantly higher correlation between archetypes identified in subsequent runs than between archetypes and random data points (t-test, $p < 10e - 16$). R^2 values are shown for a subset of digits in Fig. 4.5b. This shows that AAnet can robustly find the same set of

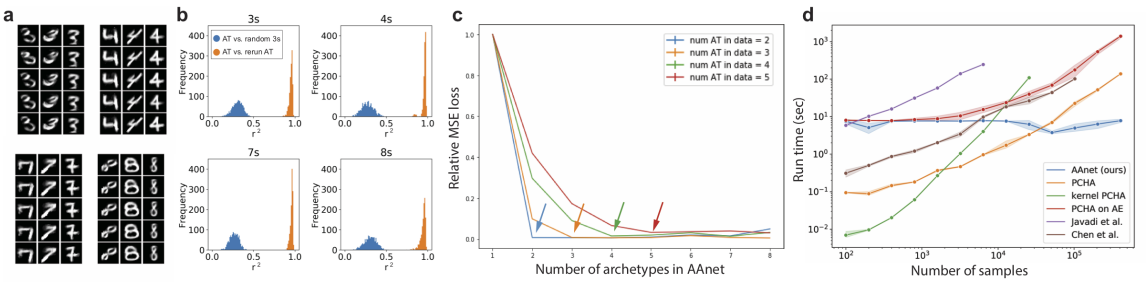

Figure 4.5: (**a**) Running AAnet repeatedly on the same dataset with different random seeds identifies similar archetypes. (**b**) AAnet was run 50 times for each digit. Pearson's r^2 was calculated between all archetypes identified for each digit (orange) and random images the digit (blue). (**c**) To pick the optimal number of archetypes, we use the knee point (arrows) of loss of AAnet run on simplexes with varying numbers of vertices. **d**) Run time of AAnet and other AA methods as a function of the number of data points on generated data with 10 archetypes.

archetypes across different runs.

4.4.5 Optimal number of archetypes

One of the main parameters in AAnet is the number of archetypes in the model. We find that the loss function of AAnet can point us to the optimal number of archetypes, *i.e.* the true number of archetypes present in the data. Increasing the number of archetypes will cause the loss to decrease generally. However, the rate of decrease diminishes, with the loss converging at the right number of archetypes. To quantify this, we generated data with different numbers of archetypes (from 2 to 5) and ran AAnet with increasing numbers of archetypes in the model (1 to 8) and recorded the loss (Fig. 4.5c). We can observe an exponential decrease of the loss with increasing numbers of archetypes in the model. Indeed, the loss plateaus at exactly the correct number of archetypes which can be found using an elbow analysis. This is similar to the approach used by [142] in which they used an elbow analysis of the explained variance by PCHA as a function of increasing numbers of model archetypes to pick the optimal number of archetypes.

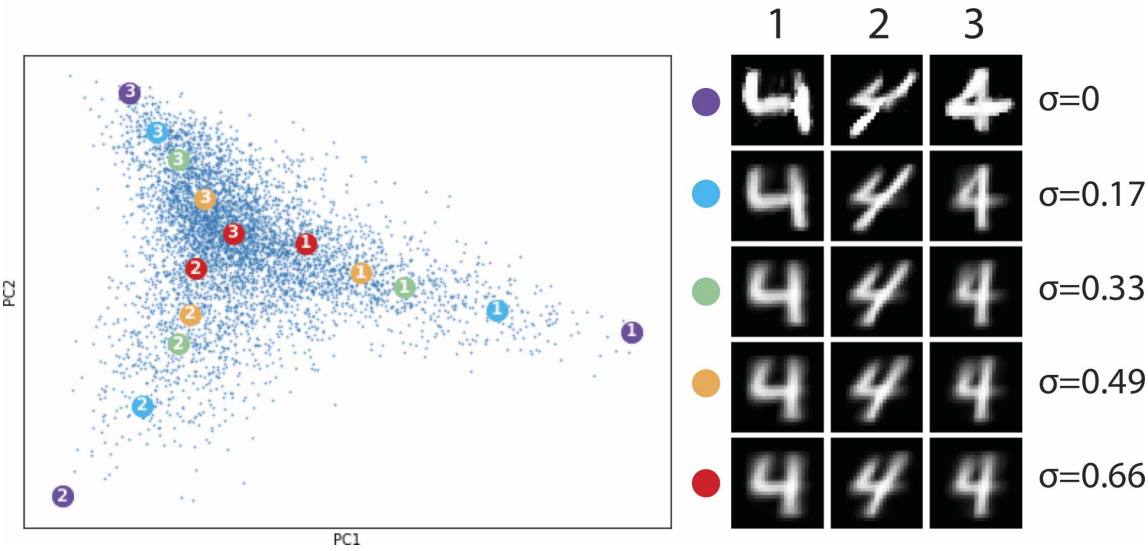

Figure 4.6: Adding increasing amounts of Gaussian noise (with standard deviation σ) to the latent archetypal layer causes the archetypes (circles with numbers) to come closer to (and inside) the data (blue points).

4.4.6 Latent noise for tight archetypes

Archetypes can lie far outside of the data or they can be close to data points. We are able to control the tightness of the archetypes by changing the amount of Gaussian noise we add during training to the latent archetypal layer. Increasing the noise causes the convex hull to become tighter and the archetypes to come closer to the data. To illustrate this, we ran AAnet on MNIST 4s with increasing amounts of noise (see Figure 4.6). We observe that as noise increases the archetypes move closer to and inside the data. With no noise the archetypes represent hypothetical points, as they are effectively outside or in very sparse outer regions of the data. Thus, with less noise the archetypes become more extreme.

4.4.7 Runtime

Another advantage of archetypal analysis with neural networks is that it is scalable. To quantify this, we ran AAnet and the other methods on increasing sample numbers of data generated on a 10 dimensional simplex that was projected into 100 dimensions (Fig. 4.5d). While several methods run faster on smaller data (e.g. PCHA is faster or as fast up to

around 50,000 samples) AAnet has the fastest run time on bigger data. In fact, the run time of AAnet is constant, while the other methods all have exponentially increasing run times with number of data points.

4.4.8 Visualizing the archetypal space

To visualize the archetypal space, we developed a fast interpolation-based method using multidimensional scaling (MDS). First, we perform MDS on the archetypes in the feature space so that the placement of the archetypes in the plot are fixed with respect to each other. Next, the coordinates of the data in two or three dimensions are found by linearly interpolating between the coordinates of the archetypes using the archetypal mixtures learned by each method.

If $\mathbf{A}$ is the n-dimensional MDS coordinates of the archetypes and $\mathbf{W}$ represents that archetypal mixtures of each point in the data, then $\mathbf{X}$, the desired n-dimensional MDS coordinates of the data can be calculated by:

$$\mathbf{X} = \mathbf{WA}$$

In practice, this interpolation method yields similar results to running MDS on a matrix comprising $\mathbf{W}$ concatenated to the archetypes along the zero-th (vertical) axis. However, this visualization method is dramatically faster. Running on 15,000 points, our method completed in 0.05 seconds to generate the coordinates show in Fig. 4.7. Running MDS directly on all points in the archetypal space (a 15000x4 matrix), took 99 minutes to complete. We find that the results for the two visualization methods (neither of which are used for quantification) are qualitatively similar across datasets.

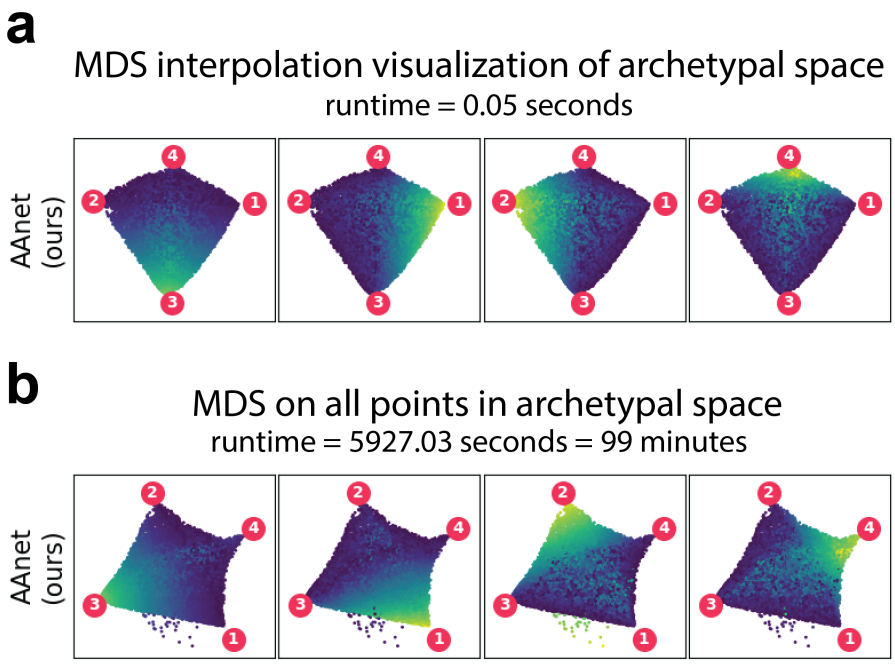

Figure 4.7: Comparison of our MDS interpolation method for visualizing the archetypal space to running MDS directly on all points in the archetypal space. Runtimes reflect time to calculate coordinates for 15,000 points from the dSprites experiment run on 12 cores running at 3.4GHz.

4.4.9 Characterization of tumor-infiltrating lymphocytes using single-cell sequencing

Although immune cell phenotypes have classically been modelled as discrete cell states, recent applications of single-cell RNA-sequencing (scRNA-seq) have found that immune cells are better described as a continuous spectrum of states [49, 149]. To characterize the continuous and non-linear transcriptional state space of immune cells, we applied AAnet to a newly generated scRNA-seq dataset of 3,554 lymphocytes extracted from mouse tumors and selected for expression of the T cell marker CD3. We visualized the dataset using PHATE, a dimensionality reduction method for biomedical data [11]. We found that 6 archetypes best describe the dataset, with each archetype representing a specific region of the overall state space. In Fig. 4.8a, expression of T cell marker genes is plotted on a PHATE embedding with missing gene expression values imputed using MAGIC [17].

We also found that AAnet was able to represent a relatively small subset of around 150 Cytotoxic T cells expressing interferon-gamma (IFNγ), but not profilin 1 (PFN1) (AT 3 in Fig. 4.8a).

Next, we sought to derive a gene signature of each archetype. We decoded the archetypes into the original gene expression space and calculated the percentile expression of all genes in each archetype compared to the input dataset. Fig. 4.8b shows the expression of the top 5 markers for each archetype. These signatures capture known markers of T cell states, such as expression of the IFNγ receptor (IFNGR2) in archetype 2 (Naive T cells)[150], high expression of perforin 1 (PRF1) in archetype 4 (Cytotoxic T cells) [151], and upregulation of CD40L in archetype 1 (activated memory cells) [152]. From these results, we conclude that AAnet is capable of characterizing the state space of a clinically-relevant biological system.

4.4.10 AAnet identifies archetypal states of gut microbiomes

The microbiota residing in the human gut have an impact on human health, yet little is understood about the microbial diversity of the gut microbiome across individuals. Findings from the first datasets of gut microbial diversity suggested that the microbial profiles of individuals fit into one of several discrete clusters called enterotypes [153]. However, more recent analysis suggests that gut diversity is better described by a spectrum of states enriched for different bacterial populations [154, 155]. Recently, access to cohorts of thousands of individual microbiome profiles make it possible to understand the space of human gut microbial composition. To show the utility of AAnet in characterizing this state space, we accessed 8,624 gut microbiome profiles from the American Gut project [156]. Here, bacterial diversity was determined using the 16S rRNA gene. We visualized the data using PHATE and found that the data was well described by 5 archetypes (Figure 4.9).

Examining the abundance of various bacterial populations, we find that these archetypes

represent biologically relevant microbiome states. For example, two classical enterotypes are characterized by high abundance of the Bacteroides and Prevotella genuses, respectively [153]. We find that abundance of the Bacteriodes and Prevotella genuses increases in points closest to archtypes 3 and 5, respectively. This suggests that the classical enterotypes are captured by AAnet. However, we identify three other archetypes characterized by high abundance of Ruminococcaceae and Tenericutes (archetype 1), Alpha-, beta-, and Gammaproteobacteria (archetype 2), and Actinobacteria and Streptococcus (archetype 4) (Figure 4.9b). The significance of these archetypal states remains to be investigated.

Finally, we demonstrate that the archetypes capture non-linear trends in microbial abundance. To show this, we plotted the abundance of various bacterial populations within each individual as a function of the distance of that individual to a target archetype in the latent space (Figure 4.9c). Here, a LOWESS curve is fit to the data and plotted as a dashed red line. For example, examining abundance of the Firmicutes and Proteobacteria, we observe a clear non-linear trend in composition as individuals are increasingly distance from Archetypes 1 and 2 respectively. These results show that AAnet can be used to characterize non-linear trends across features in high-dimensional biological systems.

4.5 Conclusion

The main contribution of this chapter is a non-linear reformulation of archetypal analysis that is solved by our neural network that we call AAnet, which features a novel archetypal regularization that enforces a convex encoding of the data in the latent layer. AAnet is an improvement over existing linear and non-linear AA methods, since AAnet 1) can learn an archetypal space even when the original data is not well fit by a simplex, 2) learns a new and optimal non-linear transformation instead of performing linear AA on a fixed non-linear transformation, such as a kernel, and 3) AAnet can generate data from a geometric description of the data [157] since it learns the boundary of the data geometry rather than

the data density. Such descriptions are especially useful when describing biological phenotypes, since biological entities (cells, people, etc.) can exist in a non-uniform continuum of states. Using this geometric description of the data we can generate new data points by sampling uniformly from the latent archetypal space, which is useful for data that is sparse or missing in certain regions of the geometry.

4.6 Supplement

4.6.1 Neural network parameters

The following parameters were used for experiments using AAnet. We used the same network parameters for the autoencoder networks used for PCHA on AE with two differences. First, the weights on the archetypal regularizations are set to 0 for the AE used for PCHA such that only MSE reconstruction loss was used for training. Second, we removed one hidden layer from the AE on PCHA when training on the dSprites dataset because this improved training of the vanilla AE.

For all datasets, we used 1024, 512, 256, 128 nodes in the four hidden layers of the Encoder and 128, 256, 512, 1024 nodes in the four hidden layers of the Decoder. We used between 1-8 ATs for each dataset as notes in the Results sections. All hidden layers contain LRelu activations, besides layers directly before and after archetypal layer which are linear so that each point is a linear combination of archetypes. For all but the T cell and Gut microbiome datasets, the last layer was Tanh. For the T cell and Gut microbiome datasets, a linear activation was used because these datasets were PCA reduced prior to training. The latent noise σ was set to 0.05 for all datasets and the batch size was 256. The optimizer was ADAM, the learning rate was set to 1e-3, and the weight initialization was Xavier.

4.6.2 Parameters for other methods

For PCHA, we used the Python implementation of the method from [131] provided by Ulf Aslak and available on GitHub at `https://github.com/ulfaslak/py_pcha`. PCHA was run with default parameters varying only the number of archetypes as indicated in the text. To implement kernel PCHA, we first transformed the input data, X with a linear kernel XX'. We also tried using a radial basis kernel $exp(-((X^2)/\sigma))$ with σ defined as the standard deviation of X, but this yielded exclusively higher MSE and poorer qualitative results than the linear kernel.

Implementations of the methods [132] and [133] were obtained from `https://github.com/samuelstjean/spams-python` and `http://web.stanford.edu/~hrhakim/NMF/`, respectively. Both methods were run with default parameters varying only the number of archetypes as indicated in the text.

For the GAN, we adapted code from `https://github.com/changwoolee/WGAN-GP-tensorflow` with a generator with dense layers: [100, 100, 100] to go from 100 dimensional Gaussian latent noise to our 100 dimensional data distribution with 4 archetypes, and discriminator with dense layers: [100, 100, 100, 1]. For the VAE we adapted code from `https://github.com/hwalsuklee/tensorflow-mnist-VAE` to our 100 dimensional data.

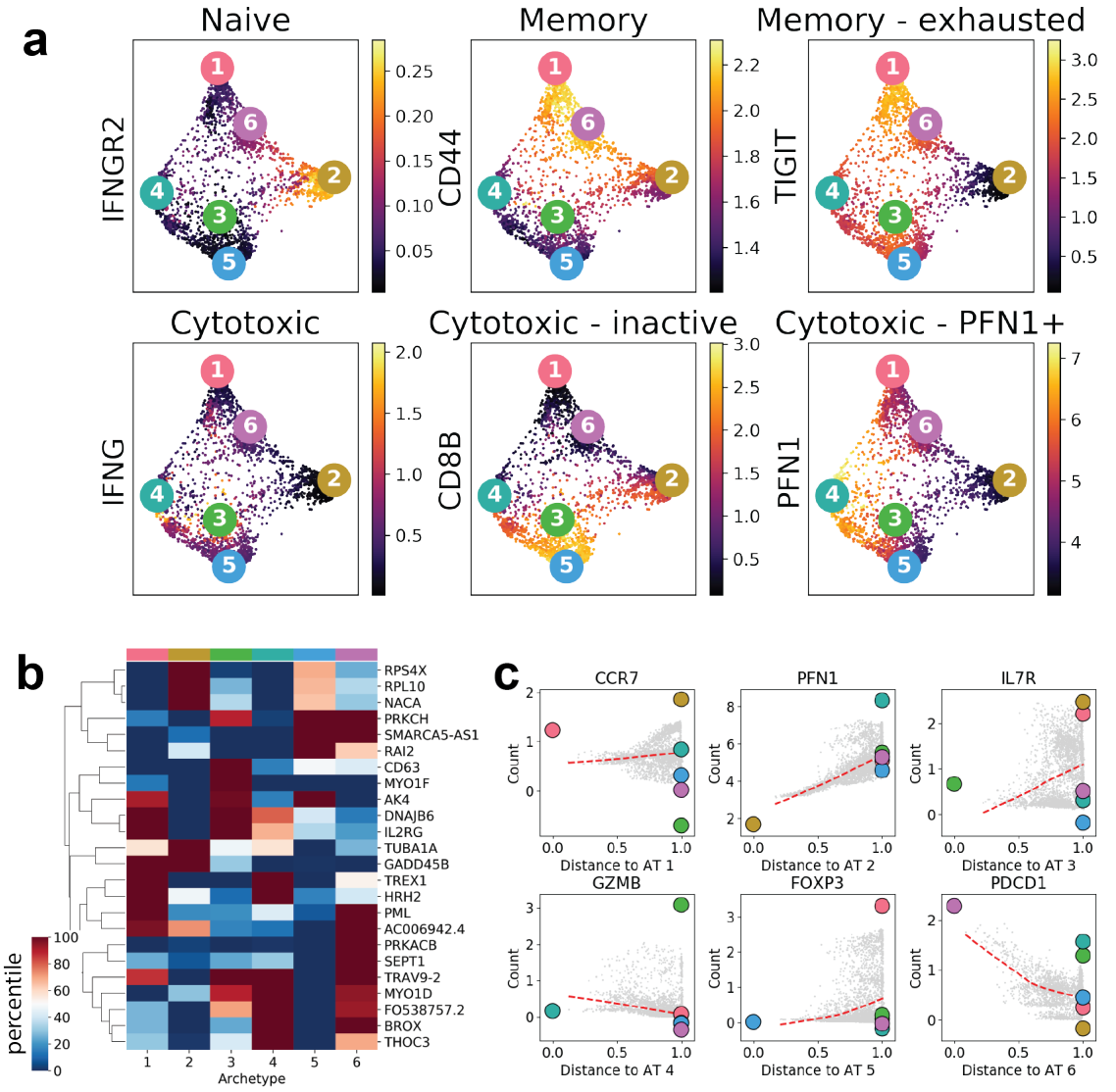

Figure 4.8: (**a**) PHATE visualization of scRNA-seq profiles and archetypes colored by gene expression for markers of T-cell states. (**b**) The top 5 genes from each expression signature of each archetype. (**c**) Plotting each cell (grey) by the distance to the each archetype shows how gene expression changes as distace to the archetype increases. Lowess curves (red) highlight the trends.

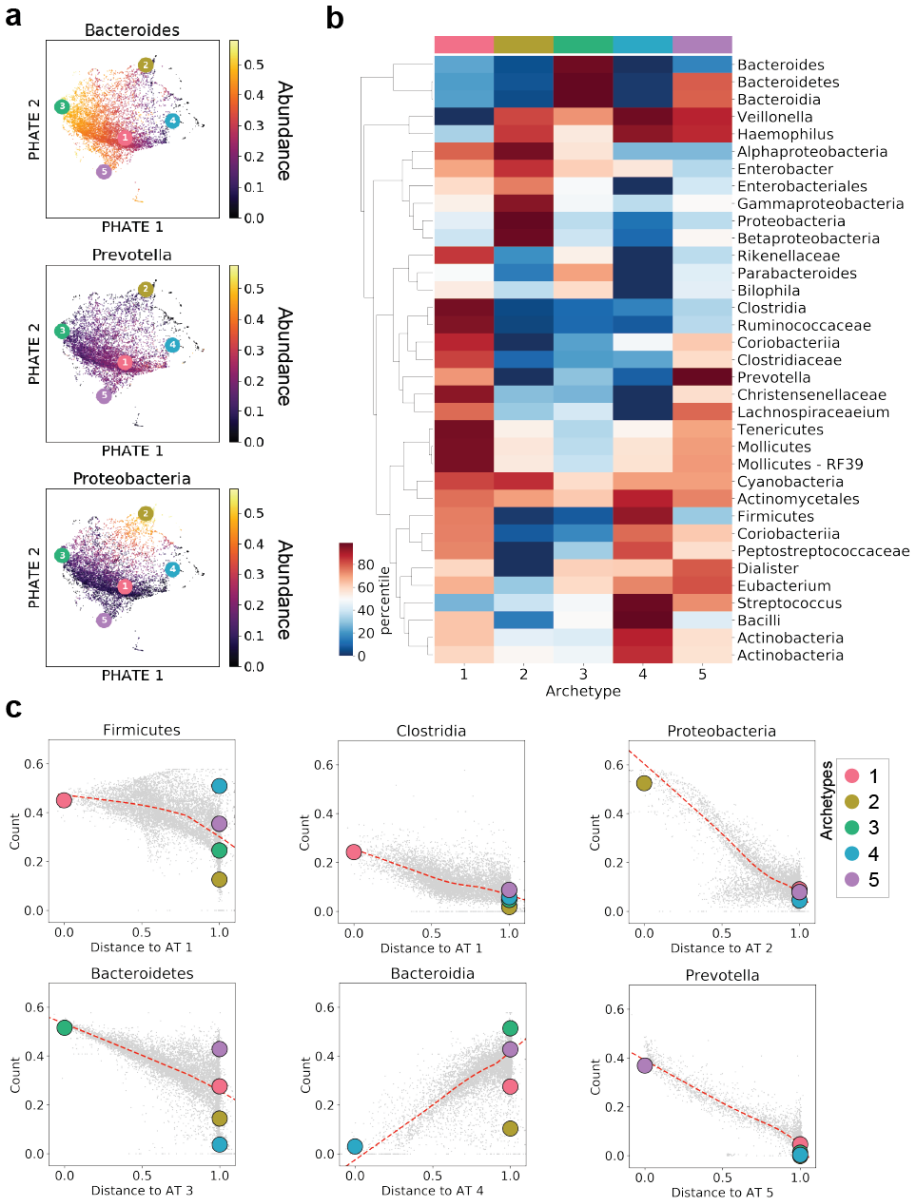

Figure 4.9: AAnet describes gut microbial diversity. (**a**) PHATE visualization of 8,624 gut microbiome profiles from the American Gut Project shows that AAnet captures archetypal states including the two classical Bacteriodes- and Prevotella-enriched enterotypes. (**b**) Abundance of archetypal microbial populations expressed as a percentile compared to the original data. (**c**) AAnet captures non-linear changes in microbial abundance. Here, abundance of each population within each individual (grey dots) is plotted as a function of that individual's distance to an archetype (colored dots). LOWESS on original data is plotted (red-dashed line)

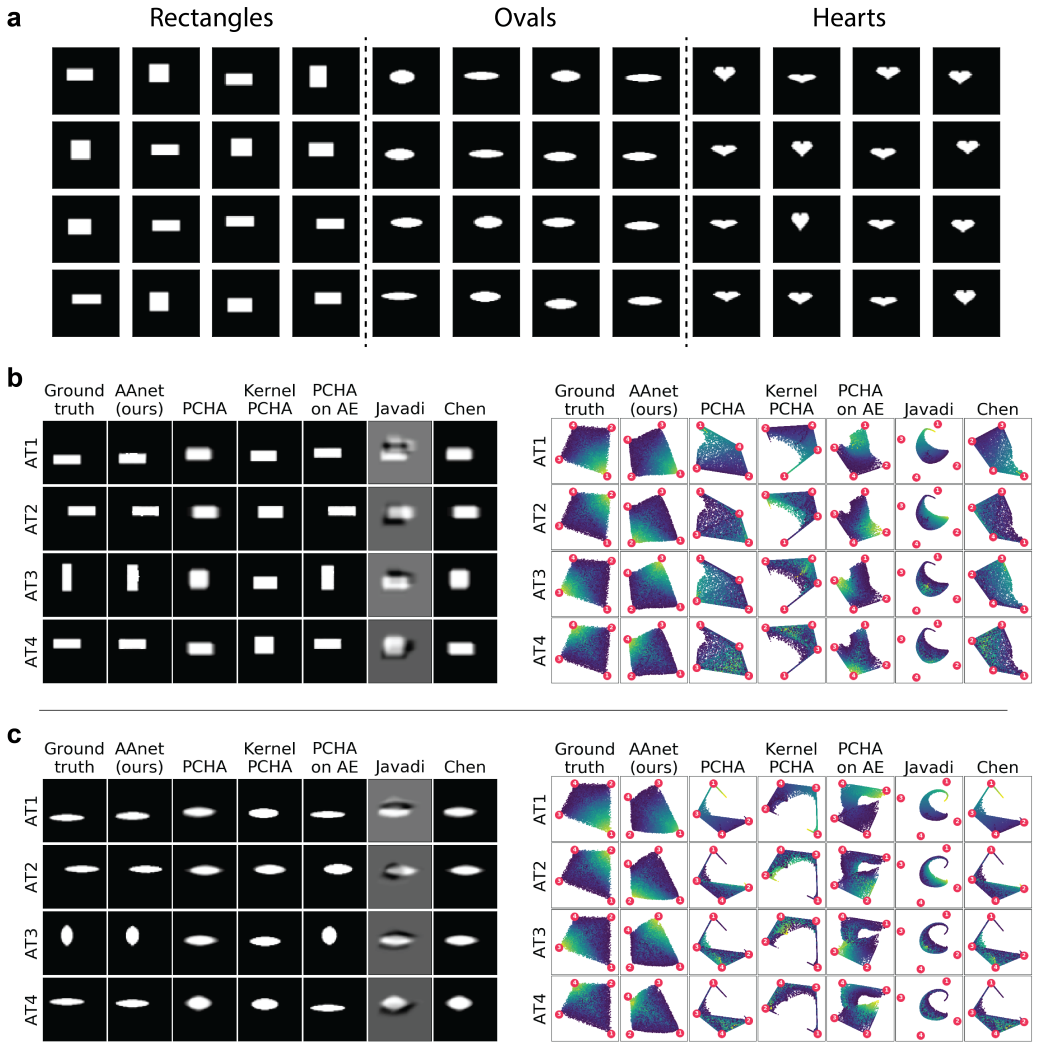

Figure 4.10: (**a**) Random samples of input data used for the dSprites experiment in Section 4.4.2. 16 random samples of each class is shown. (**b**) Left, comparison of archetypes recovered for each method using rectangles generated with the same random seed as in Fig. 4.3). Right, visualization of the archetypal spaces (*i.e.* archtypal mixtures of each point) recovered by each method. (**c**) Same as b, but for ovals. Quantification of the accuracy of the recovered archetypal spaces can be found in Fig. 4.3c).

Chapter 5

Conclusion

In this dissertation, I presented three novel algorithms for single-cell analysis. The first, Vertex Frequency Clustering, is a spectral clustering method adapted for analysis of data distributed over a graph. This approach builds on previous graph spectral techniques [52] by considering frequency patterns in metadata labels at multiple scales. By separating regions where labels are either static or smoothly varying over the graph from regions where the label changing with high frequency between adjacent nodes, we can identify populations of cells with similar responses to an experimental perturbation across a wide range of sizes of each population. Future development in VFC will focus on obtaining faster spectral representations of input labels using wavelets [57]. Avoiding a full eigen-decomposition of the graph Laplacian for the WGFT will enable the VFC to be applied to datasets of hundreds of thousands of cells.

Next, I described MELD, which builds on the way VFC considers spectral patterns in metadata expressed as graph signals. Rather than clustering on a frequency-domain representation of the signal, MELD uses a low-pass filter to perform kernel density estimation of each sample over the graph. These density estimates provide the probability of observing a given cell within a given condition. However, these can also be viewed as the likelihood of the condition given the cell. By considering the relative likelihood of

each treatment condition, we can calculate a measurement of the perturbation effect for every cell in the dataset. This measurement provides previously unavailable resolution into which populations of cells are most or least affected by a perturbation. I showed how this provides new insight into both previously published and new biological datasets and performs better than existing methods in quantitative benchmarks.

Future development of MELD will focus on the ability to measure distances between distributions of samples. Currently, MELD provides density estimates for each sample. However, there is currently no way to determine how similar conditions are to each other. One potential solution to this problem is the application of Wasserstein distances which provides a distance metric between probability distributions. Recent methods have been described to extend Wasserstein distance to discrete domains, such as graphs [158]. This approach could be further used to interpolate between conditions to identify a distribution of cells that are representative of an intermediate condition.

Finally, I presented AAnet, an autoencoder for archetypal analysis. AAnet provides a new framework for archetypal analysis where the goal is to learn a flexible transformation between the ambient space and the latent space, which is bounded by a simplex. The transformation is learned through a novel regularization on the bottleneck layer of the autoencoder that constrains the network to encode the input data as a convex combination of activations of the bottleneck nodes. AAnet is scalable and non-linear with potential applications across data domains. AAnet is especially well suited for single-cell analysis where fitting of a convex hull in the ambient space may not identify extreme states.

These three methods also suggest future approaches for analysis of single-cell experiments. Ongoing work is examining the potential for MELD to be used as a measure of sample density within the latent space of AAnet. This suggests a new way to examine changes in sample composition across experimental conditions. It also provides an opportunity for synthetic data generation across experimental conditions. Theses approaches may be able to generate new insight into a diverse range of biological processes.

Bibliography

[1] Evan Z. Macosko, Anindita Basu, Rahul Satija, James Nemesh, Karthik Shekhar, Melissa Goldman, Itay Tirosh, Allison R. Bialas, Nolan Kamitaki, Emily M. Martersteck, John J. Trombetta, David A. Weitz, Joshua R. Sanes, Alex K. Shalek, Aviv Regev, and Steven A. McCarroll. Highly Parallel Genome-Wide Expression Profiling of Individual Cells Using Nanoliter Droplets. *Cell*, 161(5):1202–1214, May 2015. ISSN 0092-8674, 1097-4172. doi: 10.1016/j.cell.2015.05.002. 1

[2] Allon M. Klein, Linas Mazutis, Ilke Akartuna, Naren Tallapragada, Adrian Veres, Victor Li, Leonid Peshkin, David A. Weitz, and Marc W. Kirschner. Droplet Barcoding for Single-Cell Transcriptomics Applied to Embryonic Stem Cells. *Cell*, 161(5):1187–1201, May 2015. ISSN 0092-8674, 1097-4172. doi: 10.1016/j.cell.2015.04.044. 1

[3] Jason D. Buenrostro, Beijing Wu, Ulrike M. Litzenburger, Dave Ruff, Michael L. Gonzales, Michael P. Snyder, Howard Y. Chang, and William J. Greenleaf. Single-cell chromatin accessibility reveals principles of regulatory variation. *Nature*, 523 (7561):486–490, July 2015. ISSN 1476-4687. doi: 10.1038/nature14590. 1

[4] D. A. Cusanovich, R. Daza, A. Adey, H. A. Pliner, L. Christiansen, K. L. Gunderson, F. J. Steemers, C. Trapnell, and J. Shendure. Multiplex single cell profiling of chromatin accessibility by combinatorial cellular indexing. *Science*, 348(6237): 910–4, May 2015. ISSN 1095-9203 (ELECTRONIC) 0036-8075 (LINKING). doi: 10.1126/science.aab1601. 1

[5] Sean C. Bendall, Erin F. Simonds, Peng Qiu, El-ad D. Amir, Peter O. Krutzik, Rachel Finck, Robert V. Bruggner, Rachel Melamed, Angelica Trejo, Olga I. Ornatsky, Robert S. Balderas, Sylvia K. Plevritis, Karen Sachs, Dana Pe'er, Scott D. Tanner, and Garry P. Nolan. Single-Cell Mass Cytometry of Differential Immune and Drug Responses Across a Human Hematopoietic Continuum. *Science*, 332 (6030):687–696, May 2011. ISSN 0036-8075, 1095-9203. doi: 10.1126/science.1198704. 1

[6] Marlon Stoeckius, Christoph Hafemeister, William Stephenson, Brian Houck-Loomis, Pratip K. Chattopadhyay, Harold Swerdlow, Rahul Satija, and Peter Smibert. Simultaneous epitope and transcriptome measurement in single cells. *Nature*

Methods, 14(9):865–868, September 2017. ISSN 1548-7105. doi: 10.1038/nmeth.4380. 1

[7] Britt Adamson, Thomas M. Norman, Marco Jost, Min Y. Cho, James K. Nuñez, Yuwen Chen, Jacqueline E. Villalta, Luke A. Gilbert, Max A. Horlbeck, Marco Y. Hein, Ryan A. Pak, Andrew N. Gray, Carol A. Gross, Atray Dixit, Oren Parnas, Aviv Regev, and Jonathan S. Weissman. A Multiplexed Single-Cell CRISPR Screening Platform Enables Systematic Dissection of the Unfolded Protein Response. *Cell*, 167(7):1867–1882.e21, December 2016. ISSN 0092-8674. doi: 10.1016/j.cell.2016.11.048. 1

[8] 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*, January 2017. ISSN 1548-7091. doi: 10.1038/nmeth.4177. 1, 26, 37, 38, 39, 40, 77, 82, 91, 96, 98, 105

[9] Etienne Becht, Leland McInnes, John Healy, Charles-Antoine Dutertre, Immanuel W. H. 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, January 2019. ISSN 1546-1696. doi: 10.1038/nbt.4314. 1, 26, 29

[10] Dmitry Kobak and Philipp Berens. The art of using t-SNE for single-cell transcriptomics. *Nature Communications*, 10(1):5416, November 2019. ISSN 2041-1723. doi: 10.1038/s41467-019-13056-x. 1

[11] Kevin R. Moon, David van Dijk, Zheng Wang, Scott Gigante, Daniel B. Burkhardt, William S. Chen, Kristina Yim, Antonia van den Elzen, Matthew J. Hirn, Ronald R. Coifman, Natalia B. Ivanova, Guy Wolf, and Smita Krishnaswamy. Visualizing structure and transitions in high-dimensional biological data. *Nature Biotechnology*, 37(12):1482–1492, December 2019. ISSN 1546-1696. doi: 10.1038/s41587-019-0336-3. 21, 26, 29, 32, 38, 52, 123

[12] Ashwin Narayan, Bonnie Berger, and Hyunghoon Cho. Assessing single-cell transcriptomic variability through density-preserving data visualization. *Nature Biotechnology*, pages 1–10, January 2021. ISSN 1546-1696. doi: 10.1038/s41587-020-00801-7.

[13] F. William Townes, Stephanie C. Hicks, Martin J. Aryee, and Rafael A. Irizarry. Feature selection and dimension reduction for single-cell RNA-Seq based on a multinomial model. *Genome Biology*, 20(1):295, December 2019. ISSN 1474-760X. doi: 10.1186/s13059-019-1861-6.

[14] Caleb Weinreb, Samuel Wolock, Allon M. Klein, and Bonnie Berger. SPRING: A kinetic interface for visualizing high dimensional single-cell expression data. *Bioinformatics*, 34(7):1246–1248, April 2018. ISSN 1367-4803. doi: 10.1093/bioinformatics/btx792. 1, 26, 29

[15] Mo Huang, Jingshu Wang, Eduardo Torre, Hannah Dueck, Sydney Shaffer, Roberto Bonasio, John I. Murray, Arjun Raj, Mingyao Li, and Nancy R. Zhang. SAVER: Gene expression recovery for single-cell RNA sequencing. *Nature Methods*, 15(7): 539–542, July 2018. ISSN 1548-7105. doi: 10.1038/s41592-018-0033-z. 1

[16] George C. Linderman, Jun Zhao, and Yuval Kluger. Zero-preserving imputation of scRNA-Seq data using low-rank approximation. *bioRxiv*, page 397588, August 2018. doi: 10.1101/397588.

[17] David van Dijk, Roshan Sharma, Juozas Nainys, Kristina Yim, Pooja Kathail, Ambrose J. Carr, Cassandra Burdziak, Kevin R. Moon, Christine L. Chaffer, Diwakar Pattabiraman, Brian Bierie, Linas Mazutis, Guy Wolf, Smita Krishnaswamy, and Dana Pe'er. Recovering Gene Interactions from Single-Cell Data Using Data Diffusion. *Cell*, 174(3):716–729.e27, July 2018. ISSN 0092-8674. doi: 10.1016/j.cell.2018.05.061. 26, 29, 32, 123

[18] Florian Wagner, Yun Yan, and Itai Yanai. K-nearest neighbor smoothing for high-throughput single-cell RNA-Seq data. *bioRxiv*, page 217737, April 2018. doi: 10.1101/217737. 1

[19] Matthew Amodio, David van Dijk, Krishnan Srinivasan, William S. Chen, Hussein Mohsen, Kevin R. Moon, Allison Campbell, Yujiao Zhao, Xiaomei Wang, Manjunatha Venkataswamy, Anita Desai, V. Ravi, Priti Kumar, Ruth Montgomery, Guy Wolf, and Smita Krishnaswamy. Exploring single-cell data with deep multitasking neural networks. *Nature Methods*, 16(11):1139–1145, November 2019. ISSN 1548-7105. doi: 10.1038/s41592-019-0576-7. 1, 7

[20] Brian Hie, Bryan Bryson, and Bonnie Berger. Efficient integration of heterogeneous single-cell transcriptomes using Scanorama. *Nature Biotechnology*, 37(6):685–691, June 2019. ISSN 1546-1696. doi: 10.1038/s41587-019-0113-3.

[21] Ilya Korsunsky, Nghia Millard, Jean Fan, Kamil Slowikowski, Fan Zhang, Kevin Wei, Yuriy Baglaenko, Michael Brenner, Po-ru Loh, and Soumya Raychaudhuri. Fast, sensitive and accurate integration of single-cell data with Harmony. *Nature Methods*, 16(12):1289–1296, December 2019. ISSN 1548-7105. doi: 10.1038/s41592-019-0619-0.

[22] 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, December 2018. ISSN 1548-7105. doi: 10.1038/s41592-018-0229-2. 7

[23] Krzysztof Polański, Matthew D Young, Zhichao Miao, Kerstin B Meyer, Sarah A Teichmann, and Jong-Eun Park. BBKNN: Fast batch alignment of single cell transcriptomes. *Bioinformatics*, 36(3):964–965, February 2020. ISSN 1367-4803. doi: 10.1093/bioinformatics/btz625.

[24] 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.e21, June 2019. ISSN 0092-8674. doi: 10.1016/j.cell.2019.05.031. 1

[25] Ming-Wen Hu, Dong Won Kim, Sheng Liu, Donald J. Zack, Seth Blackshaw, and Jiang Qian. PanoView: An iterative clustering method for single-cell RNA sequencing data. *PLOS Computational Biology*, 15(8):e1007040, August 2019. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1007040. 1

[26] Jacob H. Levine, Erin F. Simonds, Sean C. Bendall, Kara L. Davis, El-ad D. Amir, Michelle D. Tadmor, Oren Litvin, Harris G. Fienberg, Astraea Jager, Eli R. Zunder, Rachel Finck, Amanda L. Gedman, Ina Radtke, James R. Downing, Dana Pe'er, and Garry P. Nolan. Data-Driven Phenotypic Dissection of AML Reveals Progenitor-like Cells that Correlate with Prognosis. *Cell*, 162(1):184–197, July 2015. ISSN 0092-8674. doi: 10.1016/j.cell.2015.05.047. 11, 26, 29

[27] V. A. Traag, L. Waltman, and N. J. van Eck. From Louvain to Leiden: Guaranteeing well-connected communities. *Scientific Reports*, 9(1):5233, March 2019. ISSN 2045-2322. doi: 10.1038/s41598-019-41695-z. 37

[28] F. Alexander Wolf, Fiona K. Hamey, Mireya Plass, Jordi Solana, Joakim S. Dahlin, Berthold Göttgens, Nikolaus Rajewsky, Lukas Simon, and Fabian J. Theis. PAGA: Graph abstraction reconciles clustering with trajectory inference through a topology preserving map of single cells. *Genome Biology*, 20(1):59, March 2019. ISSN 1474-760X. doi: 10.1186/s13059-019-1663-x.

[29] Chen Xu and Zhengchang Su. Identification of cell types from single-cell transcriptomes using a novel clustering method. *Bioinformatics (Oxford, England)*, 31(12): 1974–1980, June 2015. ISSN 1367-4811. doi: 10.1093/bioinformatics/btv088. 11, 26, 29

[30] Amit Zeisel, Ana B. Muñoz-Manchado, Simone Codeluppi, Peter Lönnerberg, Gioele La Manno, Anna Juréus, Sueli Marques, Hermany Munguba, Liqun He, Christer Betsholtz, Charlotte Rolny, Gonçalo Castelo-Branco, Jens Hjerling-Leffler, and Sten Linnarsson. Cell types in the mouse cortex and hippocampus revealed by single-cell RNA-Seq. *Science*, 347(6226):1138–1142, March 2015. ISSN 0036-8075, 1095-9203. doi: 10.1126/science.aaa1934. 1

[31] Wouter Saelens, Robrecht Cannoodt, Helena Todorov, and Yvan Saeys. A comparison of single-cell trajectory inference methods. *Nature Biotechnology*, 37(5): 547–554, May 2019. ISSN 1546-1696. doi: 10.1038/s41587-019-0071-9. 1

[32] L. J. P. van der Maaten. Visualizing High-Dimensional Data Using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008. 1

[33] Emma Pierson and Christopher Yau. ZIFA: Dimensionality reduction for zero-inflated single-cell gene expression analysis. *Genome Biology*, 16:241, 2015. ISSN 1474-760X. doi: 10.1186/s13059-015-0805-z. 2

[34] Nicola Minshall and Anna Git. Enzyme- and gene-specific biases in reverse transcription of RNA raise concerns for evaluating gene expression. *Scientific Reports*, 10(1):8151, May 2020. ISSN 2045-2322. doi: 10.1038/s41598-020-65005-0. 2

[35] Valentine Svensson. Droplet scRNA-Seq is not zero-inflated. *Nature Biotechnology*, 38(2):147–150, February 2020. ISSN 1546-1696. doi: 10.1038/s41587-019-0379-5. 2

[36] Yingying Cao, Simo Kitanovski, Ralf Küppers, and Daniel Hoffmann. UMI or not UMI, that is the question for scRNA-Seq zero-inflation. *Nature Biotechnology*, 39 (2):158–159, February 2021. ISSN 1546-1696. doi: 10.1038/s41587-020-00810-6. 2

[37] Kevin R. Moon, Jay S. Stanley, Daniel Burkhardt, David van Dijk, Guy Wolf, and Smita Krishnaswamy. Manifold learning-based methods for analyzing single-cell RNA-Sequencing data. *Current Opinion in Systems Biology*, 7:36–46, February 2018. ISSN 2452-3100. doi: 10.1016/j.coisb.2017.12.008. 2, 27

[38] Jan Baedke. The epigenetic landscape in the course of time: Conrad Hal Waddington's methodological impact on the life sciences. *Studies in History and Philosophy of Science Part C: Studies in History and Philosophy of Biological and Biomedical Sciences*, 44(4, Part B):756–773, December 2013. ISSN 1369-8486. doi: 10.1016/j.shpsc.2013.06.001. 3

[39] Mira Bernstein, Vin De Silva, John C. Langford, and Joshua B. Tenenbaum. Graph Approximations to Geodesics on Embedded Manifolds. 2000. 4

[40] Barbara Treutlein, Qian Yi Lee, J. Gray Camp, Moritz Mall, Winston Koh, Seyed Ali Mohammad Shariati, Sopheak Sim, Norma F. Neff, Jan M. Skotheim, Marius Wernig, and Stephen R. Quake. Dissecting direct reprogramming from fibroblast to neuron using single-cell RNA-Seq. *Nature*, 534(7607):391–395, June 2016. ISSN 1476-4687. doi: 10.1038/nature18323. 5

[41] Ronald R. Coifman and Stéphane Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1):5–30, July 2006. ISSN 1063-5203. doi: 10.1016/j.acha.2006.04.006. 4, 12, 54, 68

[42] Laleh Haghverdi, Florian Buettner, and Fabian J. Theis. Diffusion maps for high-dimensional single-cell analysis of differentiation data. *Bioinformatics*, 31(18): 2989–2998, September 2015. ISSN 1367-4803. doi: 10.1093/bioinformatics/btv325. 4

[43] D. van Dijk. MAGIC: A diffusion-based imputation method reveals gene-gene interactions in single-cell RNA-Sequencing data. *bioRxiv*, 2017. 4

[44] Jay S. Stanley, Scott Gigante, Guy Wolf, and Smita Krishnaswamy. Harmonic Alignment. In *Proceedings of the 2020 SIAM International Conference on Data Mining (SDM)*, Proceedings, pages 316–324. Society for Industrial and Applied Mathematics, January 2020. doi: 10.1137/1.9781611976236.36. 4

[45] 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, October 2016. ISSN 1548-7091. doi: 10.1038/nmeth.3971. 4

[46] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. *arXiv:1711.05101 [cs, math]*, January 2019. 7

[47] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. *The Journal of Machine Learning Research*, 11:3371–3408, December 2010. ISSN 1532-4435. 7

[48] Honglak Lee, Chaitanya Ekanadham, and Andrew Ng. Sparse deep belief net model for visual area V2. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2008. 7

[49] Lars Velten, Simon F. Haas, Simon Raffel, Sandra Blaszkiewicz, Saiful Islam, Bianca P. Hennig, Christoph Hirche, Christoph Lutz, Eike C. Buss, Daniel Nowak, Tobias Boch, Wolf-Karsten Hofmann, Anthony D. Ho, Wolfgang Huber, Andreas Trumpp, Marieke A. G. Essers, and Lars M. Steinmetz. Human haematopoietic stem cell lineage commitment is a continuous process. *Nature Cell Biology*, 19(4): 271–281, April 2017. ISSN 1476-4679. doi: 10.1038/ncb3493. 8, 123

[50] Adele Cutler and Leo Breiman. Archetypal Analysis. *Technometrics*, 36(4):338–347, 1994. ISSN 0040-1706. doi: 10.2307/1269949. 8, 107, 108

[51] Yael Korem, Pablo Szekely, Yuval Hart, Hila Sheftel, Jean Hausser, Avi Mayo, Michael E. Rothenberg, Tomer Kalisky, and Uri Alon. Geometry of the Gene Expression Space of Individual Cells. *PLOS Computational Biology*, 11(7):e1004224, July 2015. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1004224. 8, 109

[52] David I Shuman, Benjamin Ricaud, and Pierre Vandergheynst. Vertex-frequency analysis on graphs. *Applied and Computational Harmonic Analysis*, 40(2):260–291, March 2016. ISSN 1063-5203. doi: 10.1016/j.acha.2015.02.005. 11, 13, 14, 15, 16, 35, 55, 70, 72, 73, 74, 75, 76, 131

[53] F. Alexander Wolf, Philipp Angerer, and Fabian J. Theis. SCANPY : Large-scale single-cell gene expression data analysis. *Genome Biology*, 19(1):15, February 2018. ISSN 1474-760X. doi: 10.1186/s13059-017-1382-0. 11

[54] 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, May 2015. ISSN 1087-0156. doi: 10.1038/nbt.3192.

[55] Vasilis Ntranos, Govinda M. Kamath, Jesse M. Zhang, Lior Pachter, and David N. Tse. Fast and accurate single-cell RNA-seq analysis by clustering of transcript-compatibility counts. *Genome Biology*, 17(1):112, May 2016. ISSN 1474-760X. doi: 10.1186/s13059-016-0970-8. 11

[56] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98, May 2013. ISSN 1053-5888. doi: 10.1109/MSP.2012.2235192. 12, 13, 29, 55, 58

[57] David K. Hammond, Pierre Vandergheynst, and Rémi Gribonval. Wavelets on Graphs via Spectral Graph Theory. *arXiv:0912.3848 [cs, math]*, December 2009. 12, 20, 55, 64, 68, 69, 131

[58] Bin Dong. Sparse Representation on Graphs by Tight Wavelet Frames and Applications. *Applied and Computational Harmonic Analysis*, 42(3):452–479, May 2017. ISSN 10635203. doi: 10.1016/j.acha.2015.09.005. 12

[59] D. I. Shuman, C. Wiesmeyr, N. Holighaus, and P. Vandergheynst. Spectrum-Adapted Tight Graph Wavelet and Vertex-Frequency Frames. *IEEE Transactions on Signal Processing*, 63(16):4223–4235, August 2015. ISSN 1941-0476. doi: 10.1109/TSP.2015.2424203. 12

[60] Y. Jin and D. Shuman. An M-Channel critically sampled filter bank for graph signals. *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017. doi: 10.1109/ICASSP.2017.7952889. 13, 23

[61] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, NIPS'01, pages 849–856, Cambridge, MA, USA, January 2001. MIT Press. 13, 19, 75

[62] Ljubisa Stanković and Ervin Sejdic, editors. *Vertex-Frequency Analysis of Graph Signals*. Signals and Communication Technology. Springer International Publishing, 2019. ISBN 978-3-030-03573-0. doi: 10.1007/978-3-030-03574-7. 13

[63] Nathanaël Perraudin, Johan Paratte, David Shuman, Lionel Martin, Vassilis Kalofolias, Pierre Vandergheynst, and David K. Hammond. GSPBOX: A toolbox for signal processing on graphs. *arXiv:1408.5781 [cs, math]*, March 2016. 20

[64] Daniel B. Burkhardt, Jay S. Stanley, Alexander Tong, Ana Luisa Perdigoto, Scott A. Gigante, Kevan C. Herold, Guy Wolf, Antonio J. Giraldez, David van Dijk, and Smita Krishnaswamy. Quantifying the effect of experimental perturbations at single-cell resolution. *Nature Biotechnology*, pages 1–11, February 2021. ISSN 1546-1696. doi: 10.1038/s41587-020-00803-5. 21, 23

[65] A. Lehtonen, S. Matikainen, and I. Julkunen. Interferons up-regulate STAT1, STAT2, and IRF family transcription factor gene expression in human peripheral blood mononuclear cells and macrophages. *Journal of Immunology (Baltimore, Md.: 1950)*, 159(2):794–803, July 1997. ISSN 0022-1767. 23

[66] Lydia Farack, Matan Golan, Adi Egozi, Nili Dezorella, Keren Bahar Halpern, Shani Ben-Moshe, Immacolata Garzilli, Beáta Tóth, Lior Roitman, Valery Krizhanovsky, and Shalev Itzkovitz. Transcriptional Heterogeneity of Beta Cells in the Intact Pancreas. *Developmental Cell*, 48(1):115–125.e4, January 2019. ISSN 1878-1551. doi: 10.1016/j.devcel.2018.11.001. 23, 46

[67] L. Le Magoarou, R. Gribonval, and N. Tremblay. Approximate Fast Graph Fourier Transforms via Multilayer Sparse Approximations. *IEEE Transactions on Signal and Information Processing over Networks*, 4(2):407–420, June 2018. ISSN 2373-776X. doi: 10.1109/TSIPN.2017.2710619. 23

[68] Malte D Luecken and Fabian J Theis. Current best practices in single-cell RNA-Seq analysis: A tutorial. *Molecular Systems Biology*, 15(6):e8746, June 2019. ISSN 1744-4292. doi: 10.15252/msb.20188746. 26, 52, 83, 84, 85, 94

[69] Karthik Shekhar, Sylvain W. Lapan, Irene E. Whitney, Nicholas M. Tran, Evan Z. Macosko, Monika Kowalczyk, Xian Adiconis, Joshua Z. Levin, James Nemesh, Melissa Goldman, Steven A. McCarroll, Constance L. Cepko, Aviv Regev, and Joshua R. Sanes. Comprehensive Classification of Retinal Bipolar Neurons by Single-Cell Transcriptomics. *Cell*, 166(5):1308–1323.e30, August 2016. ISSN 0092-8674, 1097-4172. doi: 10.1016/j.cell.2016.07.054. 26, 29

[70] Anoop P. Patel, Itay Tirosh, John J. Trombetta, Alex K. Shalek, Shawn M. Gillespie, Hiroaki Wakimoto, Daniel P. Cahill, Brian V. Nahed, William T. Curry, Robert L. Martuza, David N. Louis, Orit Rozenblatt-Rosen, Mario L. Suvà, Aviv Regev, and Bradley E. Bernstein. Single-cell RNA-Seq highlights intratumoral heterogeneity in primary glioblastoma. *Science*, 344(6190):1396–1401, June 2014. ISSN 0036-8075, 1095-9203. doi: 10.1126/science.1254257. 26

[71] Greg Finak, Andrew McDavid, Masanao Yajima, Jingyuan Deng, Vivian Gersuk, Alex K. Shalek, Chloe K. Slichter, Hannah W. Miller, M. Juliana McElrath, Martin Prlic, Peter S. Linsley, and Raphael Gottardo. MAST: A flexible statistical framework for assessing transcriptional changes and characterizing heterogeneity in single-cell RNA sequencing data. *Genome Biology*, 16, 2015. ISSN 1474-7596. doi: 10.1186/s13059-015-0844-5.

[72] Itay Tirosh, Benjamin Izar, Sanjay M. Prakadan, Marc H. Wadsworth, Daniel Treacy, John J. Trombetta, Asaf Rotem, Christopher Rodman, Christine Lian, George Murphy, Mohammad Fallahi-Sichani, Ken Dutton-Regester, Jia-Ren Lin, Ofir Cohen, Parin Shah, Diana Lu, Alex S. Genshaft, Travis K. Hughes, Carly G. K. Ziegler, Samuel W. Kazer, Aleth Gaillard, Kellie E. Kolb, Alexandra-Chloé Villani, Cory M. Johannessen, Aleksandr Y. Andreev, Eliezer M. Van Allen, Monica Bertagnolli, Peter K. Sorger, Ryan J. Sullivan, Keith T. Flaherty, Dennie T. Frederick, Judit Jané-Valbuena, Charles H. Yoon, Orit Rozenblatt-Rosen, Alex K. Shalek, Aviv Regev, and Levi A. Garraway. Dissecting the multicellular ecosystem of metastatic melanoma by single-cell RNA-Seq. *Science*, 352(6282):189–196, April 2016. ISSN 0036-8075, 1095-9203. doi: 10.1126/science.aad0501.

[73] 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.e15, December 2016. ISSN 0092-8674. doi: 10.1016/j.cell.2016.11.039. 26

[74] Xin Gao, Deqing Hu, Madelaine Gogol, and Hua Li. ClusterMap: Comparing analyses across multiple Single Cell RNA-Seq profiles. *bioRxiv*, page 331330, June 2018. doi: 10.1101/331330. 26

[75] Daniel E. Wagner, Caleb Weinreb, Zach M. Collins, James A. Briggs, Sean G. Megason, and Allon M. Klein. Single-cell mapping of gene expression landscapes and lineage in the zebrafish embryo. *Science*, page eaar4362, April 2018. ISSN 0036-8075, 1095-9203. doi: 10.1126/science.aar4362. 26, 37, 39, 40, 42, 78, 82, 86, 97, 99, 100, 105

[76] Jeffrey A. Farrell, Yiqun Wang, Samantha J. Riesenfeld, Karthik Shekhar, Aviv Regev, and Alexander F. Schier. Single-cell reconstruction of developmental trajectories during zebrafish embryogenesis. *Science*, 360(6392):eaar3131, June 2018. ISSN 0036-8075, 1095-9203. doi: 10.1126/science.aar3131. 26, 79, 86, 87, 100

[77] Z. I. Botev, J. F. Grotowski, and D. P. Kroese. Kernel density estimation via diffusion. *Annals of Statistics*, 38(5):2916–2957, October 2010. ISSN 0090-5364, 2168-8966. doi: 10.1214/10-AOS799. 30, 66

[78] David I Shuman, Pierre Vandergheynst, and Pascal Frossard. Chebyshev polynomial approximation for distributed signal processing. In *Distributed Computing in Sensor Systems and Workshops (DCOSS), 2011 International Conference On*, pages 1–8. IEEE, 2011. 32, 68

[79] Luke Zappia, Belinda Phipson, and Alicia Oshlack. Splatter: Simulation of single-cell RNA sequencing data. *Genome Biology*, 18(1):174, September 2017. ISSN 1474-760X. doi: 10.1186/s13059-017-1305-0. 37, 39

[80] Erica A. K. DePasquale, Daniel Schnell, Phillip Dexheimer, Kyle Ferchen, Stuart Hay, Kashish Chetal, Íñigo Valiente-Alandí, Burns C. Blaxall, H. Leighton Grimes, and Nathan Salomonis. cellHarmony: Cell-Level matching and holistic comparison of single-cell transcriptomes. *Nucleic Acids Research*, 47(21):e138–e138, December 2019. ISSN 0305-1048. doi: 10.1093/nar/gkz789. 37

[81] David Fischer. Theislab/Diffxpy. June 2020. 38, 78, 82

[82] Maxim V. Kuleshov, Matthew R. Jones, Andrew D. Rouillard, Nicolas F. Fernandez, Qiaonan Duan, Zichen Wang, Simon Koplev, Sherry L. Jenkins, Kathleen M. Jagodnik, Alexander Lachmann, Michael G. McDermott, Caroline D. Monteiro, Gregory W. Gundersen, and Avi Ma'ayan. Enrichr: A comprehensive gene set enrichment analysis web server 2016 update. *Nucleic Acids Research*, 44(Web Server issue):W90–W97, July 2016. ISSN 0305-1048. doi: 10.1093/nar/gkw377. 38, 46

[83] Shuo-Ting Yen, Min Zhang, Jian Min Deng, Shireen J. Usman, Chad N. Smith, Jan Parker-Thornburg, Paul G. Swinton, James F. Martin, and Richard R. Behringer. Somatic mosaicism and allele complexity induced by CRISPR/Cas9 RNA injections in mouse zygotes. *Developmental Biology*, 393(1):3–9, September 2014. ISSN 0012-1606. doi: 10.1016/j.ydbio.2014.06.017. 42

[84] M. Hammerschmidt, F. Pelegri, M. C. Mullins, D. A. Kane, F. J. van Eeden, M. Granato, M. Brand, M. Furutani-Seiki, P. Haffter, C. P. Heisenberg, Y. J. Jiang, R. N. Kelsh, J. Odenthal, R. M. Warga, and C. Nusslein-Volhard. Dino and mercedes, two genes regulating dorsal development in the zebrafish embryo. *Development*, 123(1):95–102, December 1996. ISSN 0950-1991, 1477-9129. 42, 86, 88

[85] Stefan Schulte-Merker, Kevin J. Lee, Andrew P. McMahon, and Matthias Hammerschmidt. The zebrafish organizer requires *chordino*. *Nature*, 387(6636):862–863, June 1997. ISSN 1476-4687. doi: 10.1038/43092.

[86] Shannon Fisher and Marnie E. Halpern. Patterning the zebrafish axial skeleton requires early *chordin* function. *Nature Genetics*, 23(4):442–446, December 1999. ISSN 1546-1718. doi: 10.1038/70557. 42

[87] V. Ablamunits, D. Elias, T. Reshef, and I. R. Cohen. Islet T cells secreting IFN-Gamma in NOD mouse diabetes: Arrest by P277 peptide treatment. *Journal of Autoimmunity*, 11(1):73–81, February 1998. ISSN 0896-8411. doi: 10.1006/jaut.1997.0177. 44

[88] Miguel Lopes, Burak Kutlu, Michela Miani, Claus H. Bang-Berthelsen, Joachim Størling, Flemming Pociot, Nathan Goodman, Lee Hood, Nils Welsh, Gianluca Bontempi, and Decio L. Eizirik. Temporal profiling of cytokine-induced genes in pancreatic β-Cells by meta-analysis and network inference. *Genomics*, 103(4):264–275, April 2014. ISSN 0888-7543. doi: 10.1016/j.ygeno.2013.12.007. 44

[89] Mauro J. Muraro, Gitanjali Dharmadhikari, Dominic Grün, Nathalie Groen, Tim Dielen, Erik Jansen, Leon van Gurp, Marten A. Engelse, Francoise Carlotti, Eelco J.P. de Koning, and Alexander van Oudenaarden. A Single-Cell Transcriptome Atlas of the Human Pancreas. *Cell Systems*, 3(4):385–394.e3, October 2016. ISSN 2405-4712. doi: 10.1016/j.cels.2016.09.002. 45, 80, 101

[90] Yurong Xin, Giselle Dominguez Gutierrez, Haruka Okamoto, Jinrang Kim, Ann-Hwee Lee, Christina Adler, Min Ni, George D. Yancopoulos, Andrew J. Murphy, and Jesper Gromada. Pseudotime Ordering of Single Human β-Cells Reveals States of Insulin Production and Unfolded Protein Response. *Diabetes*, 67(9):1783–1794, September 2018. ISSN 0012-1797, 1939-327X. doi: 10.2337/db18-0365. 46

[91] Chilakamarti V Ramana, M. Pilar Gil, Robert D Schreiber, and George R Stark. Stat1-dependent and -Independent pathways in IFN-γ-Dependent signaling. *Trends in Immunology*, 23(2):96–101, February 2002. ISSN 1471-4906. doi: 10.1016/S1471-4906(01)02118-4. 46

[92] Anthony J. Sadler and Bryan R. G. Williams. Interferon-inducible antiviral effectors. *Nature reviews. Immunology*, 8(7):559–568, July 2008. ISSN 1474-1733. doi: 10.1038/nri2314. 46

[93] Katherine A. Fitzgerald. The Interferon Inducible Gene: Viperin. *Journal of Interferon & Cytokine Research*, 31(1):131–135, January 2011. ISSN 1079-9907. doi: 10.1089/jir.2010.0127.

[94] Zhiwei Zheng, Lin Wang, and Jihong Pan. Interferon-stimulated gene 20-kDa protein (ISG20) in infection and disease: Review and outlook. *Intractable & Rare Diseases Research*, 6(1):35–40, February 2017. ISSN 2186-3644. doi: 10.5582/irdr.2017.01004. 46

[95] Monica Hultcrantz, Michael H. Hühn, Monika Wolf, Annika Olsson, Stella Jacobson, Bryan R. Williams, Olle Korsgren, and Malin Flodström-Tullberg. Interferons induce an antiviral state in human pancreatic islet cells. *Virology*, 367(1):92–101, October 2007. ISSN 0042-6822. doi: 10.1016/j.virol.2007.05.010. 46

[96] Andrew F. Stewart, Mehboob A. Hussain, Adolfo García-Ocaña, Rupangi C. Vasavada, Anil Bhushan, Ernesto Bernal-Mizrachi, and Rohit N. Kulkarni. Human β-Cell Proliferation and Intracellular Signaling: Part 3. *Diabetes*, 64(6):1872–1885, June 2015. ISSN 0012-1797. doi: 10.2337/db14-1843. 46

[97] Xinyue Chen, Daniel B. Burkhardt, Amaleah A. Hartman, Xiao Hu, Anna E. Eastman, Chao Sun, Xujun Wang, Mei Zhong, Smita Krishnaswamy, and Shangqin Guo. MLL-AF9 initiates transformation from fast-proliferating myeloid progenitors. *Nature Communications*, 10(1), December 2019. doi: 10.1038/s41467-019-13666-5. 50

[98] Emily V. Dutrow, Deena Emera, Kristina Yim, Severin Uebbing, Acadia A. Kocher, Martina Krenzer, Timothy Nottoli, Daniel B. Burkhardt, Smita Krishnaswamy, Angeliki Louvi, and James P. Noonan. The Human Accelerated Region HACNS1 modifies developmental gene expression in humanized mice. *bioRxiv*, page 2019.12.11.873075, December 2019. doi: 10.1101/2019.12.11.873075.

[99] Katherine E. Savell, Jennifer J. Tuscher, Morgan E. Zipperly, Corey G. Duke, Robert A. Phillips, Allison J. Bauman, Saakshi Thukral, Faraz A. Sultan, Nicholas A. Goska, Lara Ianov, and Jeremy J. Day. A dopamine-induced gene expression signature regulates neuronal function and cocaine response. *Science Advances*, 6(26):eaba4221, June 2020. ISSN 2375-2548. doi: 10.1126/sciadv.aba4221.

[100] Katherine Minjee Chung, Jaffarguriqbal Singh, Lauren Lawres, Kimberly Judith Dorans, Cathy Garcia, Daniel B. Burkhardt, Rebecca Robbins, Arjun Bhutkar, Rebecca Cardone, Xiaojian Zhao, Ana Babic, Sara A. Vayrynen, Andressa Dias Costa, Jonathan A. Nowak, Daniel T. Chang, Richard F. Dunne, Aram F. Hezel, Albert C. Koong, Joshua J. Wilhelm, Melena D. Bellin, Vibe Nylander, Anna L. Gloyn, Mark I. McCarthy, Richard G. Kibbey, Smita Krishnaswamy, Brian M. Wolpin, Tyler Jacks, Charles S. Fuchs, and Mandar Deepak Muzumdar. Endocrine-Exocrine Signaling Drives Obesity-Associated Pancreatic Ductal Adenocarcinoma. *Cell*, 181 (4):832–847.e18, May 2020. ISSN 0092-8674. doi: 10.1016/j.cell.2020.03.062.

[101] Neal G. Ravindra, Mia Madel Alfajaro, Victor Gasque, Victoria Habet, Jin Wei, Renata B. Filler, Nicholas C. Huston, Han Wan, Klara Szigeti-Buck, Bao Wang, Guilin Wang, Ruth R. Montgomery, Stephanie C. Eisenbarth, Adam Williams, Anna Marie Pyle, Akiko Iwasaki, Tamas L. Horvath, Ellen F. Foxman, Richard W. Pierce, David van Dijk, and Craig B. Wilen. Single-cell longitudinal analysis of SARS-CoV-2 infection in human airway epithelium. *bioRxiv*, July 2020. doi: 10.1101/2020.05.06.081695. 50

[102] Laleh Haghverdi, Aaron T. L. Lun, Michael D. Morgan, and John C. Marioni. Batch effects in single-cell RNA-Sequencing data are corrected by matching mutual nearest neighbors. *Nature Biotechnology*, April 2018. ISSN 1546-1696. doi: 10.1038/nbt.4091. 52, 53, 89

[103] Y. P Mack and M Rosenblatt. Multivariate k-nearest neighbor density estimates. *Journal of Multivariate Analysis*, 9(1):1–15, March 1979. ISSN 0047-259X. doi: 10.1016/0047-259X(79)90065-4. 55

[104] Gérard Biau, Frédéric Chazal, David Cohen-Steiner, Luc Devroye, and Carlos Rodríguez. A weighted k-nearest neighbor density estimate for geometric inference. *Electronic Journal of Statistics*, 5:204–237, 2011. ISSN 1935-7524. doi: 10.1214/11-EJS606.

[105] Yi-Hung Kung, Pei-Sheng Lin, and Cheng-Hsiung Kao. An optimal k-nearest neighbor for density estimation. *Statistics & Probability Letters*, 82(10):1786–1791, October 2012. ISSN 0167-7152. doi: 10.1016/j.spl.2012.05.017.

[106] Ulrike Von Luxburg and Morteza Alamgir. Density estimation from unweighted k-nearest neighbor graphs: A roadmap. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 225–233. Curran Associates, Inc., 2013.

[107] Bernard W. Silverman. *Density Estimation for Statistics and Data Analysis*. Routledge, February 2018. ISBN 978-1-315-14091-9. doi: 10.1201/9781315140919. 55

[108] Nathanael Perraudin, Benjamin Ricaud, David Shuman, and Pierre Vandergheynst. Global and Local Uncertainty Principles for Signals on Graphs. *arXiv:1603.03030 [math-ph, stat]*, March 2016. 55, 74

[109] Stephane Mallat. *A Wavelet Tour of Signal Processing: The Sparse Way*. Academic Press, December 2008. ISBN 978-0-08-092202-7. 56, 72

[110] Dengyong Zhou and Bernhard Schölkopf. A regularization framework for learning from graph data. In *ICML Workshop on Statistical Relational Learning and Its Connections to Other Fields*, volume 15, pages 67–8, 2004. 60

[111] Jihun Ham, Daniel D Lee, and Lawrence K Saul. Semisupervised alignment of manifolds. In *AISTATS*, pages 120–127, 2005.

[112] Mikhail Belkin, Irina Matveeva, and Partha Niyogi. Regularization and semi-supervised learning on large graphs. In *International Conference on Computational Learning Theory*, pages 624–638. Springer, 2004. 61, 63

[113] Rie K. Ando and Tong Zhang. Learning on Graph with Laplacian Regularization. In B. Schölkopf, J. C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 25–32. MIT Press, 2007.

[114] Kilian Q Weinberger, Fei Sha, Qihui Zhu, and Lawrence K. Saul. Graph Laplacian Regularization for Large-Scale Semidefinite Programming. In B. Schölkopf, J. C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 1489–1496. MIT Press, 2007.

[115] X. He, M. Ji, C. Zhang, and H. Bao. A Variance Minimization Criterion to Feature Selection Using Laplacian Regularization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(10):2013–2025, October 2011. ISSN 0162-8828. doi: 10.1109/TPAMI.2011.44.

[116] X. Liu, D. Zhai, D. Zhao, G. Zhai, and W. Gao. Progressive Image Denoising Through Hybrid Graph Laplacian Regularization: A Unified Framework. *IEEE*

Transactions on Image Processing, 23(4):1491–1503, April 2014. ISSN 1057-7149. doi: 10.1109/TIP.2014.2303638.

[117] J. Pang, G. Cheung, A. Ortega, and O. C. Au. Optimal graph laplacian regularization for natural image denoising. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2294–2298, April 2015. doi: 10.1109/ICASSP.2015.7178380.

[118] Jiahao Pang and Gene Cheung. Graph Laplacian Regularization for Image Denoising: Analysis in the Continuous Domain. *IEEE Transactions on Image Processing*, 26(4):1770–1785, April 2017. ISSN 1057-7149, 1941-0042. doi: 10.1109/TIP.2017.2651400. 60

[119] Nathanaël Perraudin, Johan Paratte, David Shuman, Lionel Martin, Vassilis Kalofolias, Pierre Vandergheynst, and David K. Hammond. GSPBOX: A toolbox for signal processing on graphs. *ArXiv e-prints*, August 2014. 61

[120] Martin Barron and Jun Li. Identifying and removing the cell-cycle effect from single-cell RNA-Sequencing data. *Scientific reports*, 6:33892, 2016. 65

[121] Mikhail Belkin and Partha Niyogi. Convergence of Laplacian Eigenmaps. In B. Schölkopf, J. C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 129–136, 2006. 66, 67

[122] Ronald R Coifman and Mauro Maggioni. Diffusion wavelets. *Applied and Computational Harmonic Analysis*, 21(1):53–94, 2006. 66

[123] Probal Chaudhuri and J. S. Marron. Scale Space View of Curve Estimation. *The Annals of Statistics*, 28(2):408–428, 2000. ISSN 0090-5364. 66

[124] Nathanaël Perraudin, Nicki Holighaus, Peter L Søndergaard, and Peter Balazs. Designing Gabor windows using convex optimization. *arXiv preprint arXiv:1401.6033*, 2014. 70

[125] Maren Büttner, Zhichao Miao, F. Alexander Wolf, Sarah A. Teichmann, and Fabian J. Theis. A test metric for assessing single-cell RNA-Seq batch correction. *Nature Methods*, 16(1):43–49, January 2019. ISSN 1548-7105. doi: 10.1038/s41592-018-0254-1. 84, 88, 90

[126] Robert A. Amezquita, Aaron T. L. Lun, Etienne Becht, Vince J. Carey, Lindsay N. Carpp, Ludwig Geistlinger, Federico Marini, Kevin Rue-Albrecht, Davide Risso, Charlotte Soneson, Levi Waldron, Hervé Pagès, Mike L. Smith, Wolfgang Huber, Martin Morgan, Raphael Gottardo, and Stephanie C. Hicks. Orchestrating single-cell analysis with Bioconductor. *Nature Methods*, 17(2):137–145, February 2020. ISSN 1548-7105. doi: 10.1038/s41592-019-0654-x. 85, 94

[127] Anskar Y. H. Leung, Eric M. Mendenhall, Tommy T. F. Kwan, Raymond Liang, Craig E. Eckfeldt, Eleanor Chen, Matthias Hammerschmidt, Suzanne Grindley, Stephen C. Ekker, and Catherine M. Verfaillie. Characterization of expanded intermediate cell mass in zebrafish chordin morphant embryos. *Developmental Biology*, 277(1):235–254, January 2005. ISSN 0012-1606. doi: 10.1016/j.ydbio.2004.09.032. 86

[128] Ben Steventon, Claudio Araya, Claudia Linker, Sei Kuriyama, and Roberto Mayor. Differential requirements of BMP and Wnt signalling during gastrulation and neurulation define two steps in neural crest induction. *Development*, 136(5):771–779, March 2009. ISSN 0950-1991, 1477-9129. doi: 10.1242/dev.029017. 87

[129] Carolin Schille and Alexandra Schambony. Signaling pathways and tissue interactions in neural plate border formation. *Neurogenesis*, 4(1):e1292783, January 2017. ISSN null. doi: 10.1080/23262133.2017.1292783. 87

[130] M. D. Luecken, M. Büttner, K. Chaichoompu, A. Danese, M. Interlandi, M. F. Mueller, D. C. Strobl, L. Zappia, M. Dugas, M. Colomé-Tatché, and F. J. Theis. Benchmarking atlas-level data integration in single-cell genomics. *bioRxiv*, page 2020.05.22.111161, May 2020. doi: 10.1101/2020.05.22.111161. 88

[131] Morten Mørup and Lars Kai Hansen. Archetypal analysis for machine learning and data mining. *Neurocomputing*, 80:54–63, March 2012. ISSN 0925-2312. doi: 10.1016/j.neucom.2011.06.033. 108, 110, 114, 115, 127

[132] Yuansi Chen, Julien Mairal, and Zaid Harchaoui. Fast and Robust Archetypal Analysis for Representation Learning. *arXiv:1405.6472 [cs, stat]*, May 2014. 108, 115, 127

[133] Hamid Javadi and Andrea Montanari. Nonnegative Matrix Factorization Via Archetypal Analysis. *Journal of the American Statistical Association*, 0(0):1–22, March 2019. ISSN 0162-1459. doi: 10.1080/01621459.2019.1594832. 108, 115, 127

[134] Daan Wynen, Cordelia Schmid, and Julien Mairal. Unsupervised Learning of Artistic Styles with Archetypal Style Analysis. *arXiv:1805.11155 [cs, stat]*, October 2018. 108, 115

[135] O. Shoval, H. Sheftel, G. Shinar, Y. Hart, O. Ramote, A. Mayo, E. Dekel, K. Kavanagh, and U. Alon. Evolutionary Trade-Offs, Pareto Optimality, and the Geometry of Phenotype Space. *Science*, 336(6085):1157–1160, June 2012. ISSN 0036-8075, 1095-9203. doi: 10.1126/science.1217405. 108

[136] B. H. P. Chan, D. A. Mitchell, and L. E. Cram. Archetypal analysis of galaxy spectra. *Monthly Notices of the Royal Astronomical Society*, 338(3):790–795, January 2003. ISSN 0035-8711. doi: 10.1046/j.1365-8711.2003.06099.x. 109

[137] Shan De Li, Paul Wang, Jordan Louviere, and Richard Franklin Carson. Archetypal analysis: A new way to segment markets based on extreme individuals. 2003. 109

[138] Giovanni C. Porzio, Giancarlo Ragozini, and Domenico Vistocco. On the use of archetypes as benchmarks. *Applied Stochastic Models in Business and Industry*, 24 (5):419–437, 2008. ISSN 1526-4025. doi: 10.1002/asmb.727. 109

[139] Sohan Seth and Manuel J. A. Eugster. Probabilistic archetypal analysis. *Machine Learning*, 102(1):85–113, January 2016. ISSN 1573-0565. doi: 10.1007/s10994-015-5498-8. 109

[140] Ercan Canhasi and Igor Kononenko. Weighted hierarchical archetypal analysis for multi-document summarization. *Computer Speech & Language*, 37:24–46, May 2016. ISSN 0885-2308. doi: 10.1016/j.csl.2015.11.004. 109

[141] Juliane Charlotte Thøgersen, Morten Mørup, Søren Damkiær, Søren Molin, and Lars Jelsbak. Archetypal analysis of diverse Pseudomonas aeruginosatranscriptomes reveals adaptation in cystic fibrosis airways. *BMC Bioinformatics*, 14(1): 279, September 2013. ISSN 1471-2105. doi: 10.1186/1471-2105-14-279. 109

[142] Yuval Hart, Hila Sheftel, Jean Hausser, Pablo Szekely, Noa Bossel Ben-Moshe, Yael Korem, Avichai Tendler, Avraham E. Mayo, and Uri Alon. Inferring biological tasks using Pareto analysis of high-dimensional data. *Nature Methods*, 12(3):233–235, 3 p following 235, March 2015. ISSN 1548-7105. doi: 10.1038/nmeth.3254. 109, 120

[143] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3(Jan):993–1022, 2003. ISSN ISSN 1533-7928. 109

[144] Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. dSprites: Disentanglement testing sprites dataset. 2017. 116

[145] Stéfan van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, and Tony Yu. Scikit-image: Image processing in Python. *PeerJ*, 2:e453, June 2014. ISSN 2167-8359. doi: 10.7717/peerj.453. 117

[146] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks. *arXiv:1406.2661 [cs, stat]*, June 2014. 119

[147] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. *arXiv:1312.6114 [cs, stat]*, May 2014. 119

[148] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(null):723–773, March 2012. ISSN 1532-4435. 119

[149] Elham Azizi, Ambrose J. Carr, George Plitas, Andrew E. Cornish, Catherine Konopacki, Sandhya Prabhakaran, Juozas Nainys, Kenmin Wu, Vaidotas Kiseliovas, Manu Setty, Kristy Choi, Rachel M. Fromme, Phuong Dao, Peter T. McKenney, Ruby C. Wasti, Krishna Kadaveru, Linas Mazutis, Alexander Y. Rudensky, and Dana Pe'er. Single-Cell Map of Diverse Immune Phenotypes in the Breast Tumor Microenvironment. *Cell*, 174(5):1293–1308.e36, August 2018. ISSN 0092-8674. doi: 10.1016/j.cell.2018.05.060. 123

[150] Julie M. Curtsinger, Pujya Agarwal, Debra C. Lins, and Matthew F. Mescher. Autocrine IFN-γ promotes naive CD8 T cell differentiation and synergizes with IFN-α to stimulate strong function. *Journal of Immunology (Baltimore, Md.: 1950)*, 189 (2):659–668, July 2012. ISSN 1550-6606. doi: 10.4049/jimmunol.1102727. 124

[151] David Kagi, Francoise Vignaux, Birgit Ledermann, Kurt Burki, Valerie Depraetere, Shigekazu Nagata, Hans Hengartner, and Pierre Golstein. Fas and perforin pathways as major mechanisms of T cell-mediated cytotoxicity. *Science*, 265(5171): 528–530, 1994. 124

[152] Tak W. Mak and Mary E. Saunders. *The Immune Response: Basic and Clinical Principles*. Elsevier Science, 2006. ISBN 978-0-12-088451-3. 124

[153] Manimozhiyan Arumugam, Jeroen Raes, Eric Pelletier, Denis Le Paslier, Takuji Yamada, Daniel R. Mende, Gabriel R. Fernandes, Julien Tap, Thomas Bruls, Jean-Michel Batto, Marcelo Bertalan, Natalia Borruel, Francesc Casellas, Leyden Fernandez, Laurent Gautier, Torben Hansen, Masahira Hattori, Tetsuya Hayashi, Michiel Kleerebezem, Ken Kurokawa, Marion Leclerc, Florence Levenez, Chaysavanh Manichanh, H. Bjørn Nielsen, Trine Nielsen, Nicolas Pons, Julie Poulain, Junjie Qin, Thomas Sicheritz-Ponten, Sebastian Tims, David Torrents, Edgardo Ugarte, Erwin G. Zoetendal, Jun Wang, Francisco Guarner, Oluf Pedersen, Willem M. de Vos, Søren Brunak, Joel Doré, MetaHIT Consortium, María Antolín, François Artiguenave, Hervé M. Blottiere, Mathieu Almeida, Christian Brechot, Carlos Cara, Christian Chervaux, Antonella Cultrone, Christine Delorme, Gérard Denariaz, Rozenn Dervyn, Konrad U. Foerstner, Carsten Friss, Maarten van de Guchte, Eric Guedon, Florence Haimet, Wolfgang Huber, Johan van Hylckama-Vlieg, Alexandre Jamet, Catherine Juste, Ghalia Kaci, Jan Knol, Omar Lakhdari, Severine Layec, Karine Le Roux, Emmanuelle Maguin, Alexandre Mérieux, Raquel Melo Minardi, Christine M'rini, Jean Muller, Raish Oozeer, Julian Parkhill, Pierre Renault, Maria Rescigno, Nicolas Sanchez, Shinichi Sunagawa, Antonio Torrejon, Keith Turner, Gaetana Vandemeulebrouck, Encarna Varela, Yohanan Winogradsky, Georg Zeller, Jean Weissenbach, S. Dusko Ehrlich, and Peer Bork. Enterotypes of the human gut microbiome. *Nature*, 473(7346):174–180, May 2011. ISSN 1476-4687. doi: 10.1038/nature09944. 124, 125

[154] Ian B. Jeffery, Marcus J. Claesson, Paul W. O'Toole, and Fergus Shanahan. Categorization of the gut microbiota: Enterotypes or gradients? *Nature Reviews. Microbiology*, 10(9):591–592, September 2012. ISSN 1740-1534. 124

[155] Dan Knights, Tonya L. Ward, Christopher E. McKinlay, Hannah Miller, Antonio Gonzalez, Daniel McDonald, and Rob Knight. Rethinking "Enterotypes". *Cell host & microbe*, 16(4):433–437, October 2014. ISSN 1931-3128. doi: 10.1016/j.chom.2014.09.013. 124

[156] Daniel McDonald, Embriette Hyde, Justine W. Debelius, James T. Morton, Antonio Gonzalez, Gail Ackermann, Alexander A. Aksenov, Bahar Behsaz, Caitriona Brennan, Yingfeng Chen, Lindsay DeRight Goldasich, Pieter C. Dorrestein, Robert R. Dunn, Ashkaan K. Fahimipour, James Gaffney, Jack A. Gilbert, Grant Gogul, Jessica L. Green, Philip Hugenholtz, Greg Humphrey, Curtis Huttenhower, Matthew A. Jackson, Stefan Janssen, Dilip V. Jeste, Lingjing Jiang, Scott T. Kelley, Dan Knights, Tomasz Kosciolek, Joshua Ladau, Jeff Leach, Clarisse Marotz, Dmitry Meleshko, Alexey V. Melnik, Jessica L. Metcalf, Hosein Mohimani, Emmanuel Montassier, Jose Navas-Molina, Tanya T. Nguyen, Shyamal Peddada, Pavel Pevzner, Katherine S. Pollard, Gholamali Rahnavard, Adam Robbins-Pianka, Naseer Sangwan, Joshua Shorenstein, Larry Smarr, Se Jin Song, Timothy Spector, Austin D. Swafford, Varykina G. Thackray, Luke R. Thompson, Anupriya Tripathi, Yoshiki Vázquez-Baeza, Alison Vrbanac, Paul Wischmeyer, Elaine Wolfe, Qiyun Zhu, The American Gut Consortium, and Rob Knight. American Gut: An Open Platform for Citizen Science Microbiome Research. *mSystems*, 3(3):e00031–18, June 2018. ISSN 2379-5077. doi: 10.1128/mSystems.00031-18. 124

[157] Ofir Lindenbaum, Jay Stanley, Guy Wolf, and Smita Krishnaswamy. Geometry Based Data Generation. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 1407–1418. Curran Associates, Inc., 2018. 125

[158] Justin Solomon, Fernando de Goes, Gabriel Peyré, Marco Cuturi, Adrian Butscher, Andy Nguyen, Tao Du, and Leonidas Guibas. Convolutional wasserstein distances: Efficient optimal transportation on geometric domains. *ACM Transactions on Graphics*, 34(4):66:1–66:11, July 2015. ISSN 0730-0301. doi: 10.1145/2766963. 132

Copyright Acknowledgements

Data and text from this dissertation has been published as:

D. B. Burkhardt, J. S. S. III, G. Wolf, and S. Krishnaswamy, "Vertex-Frequency Clustering," in 2019 IEEE Data Science Workshop (DSW), Jun. 2019, pp. 145–149, doi: 10.1109/DSW.2019.8755591.

D. B. Burkhardt et al., "Quantifying the effect of experimental perturbations at single-cell resolution," Nature Biotechnology, pp. 1–11, Feb. 2021, doi: 10.1038/s41587-020-00803-5.

D. van Dijk, D. B. Burkhardt, M. Amodio, A. Tong, G. Wolf, and S. Krishnaswamy, "Finding Archetypal Spaces Using Neural Networks," arXiv:1901.09078 [cs, stat], Nov. 2019, Accessed: Feb. 22, 2021. [Online]. Available: http://arxiv.org/abs/1901.09078.