Learning and inference with single cell data

Sumit Mukherjee

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2018

Reading Committee:

Georg Seelig, Chair

Sreeram Kannan, Chair

Su-in Lee

Program Authorized to Offer Degree:
Electrical Engineering

University of Washington

Abstract

Learning and inference with single cell data

Sumit Mukherjee

Co-Chairs of the Supervisory Committee:
Professor Georg Seelig
Computer Science & Engineering
Electrical Engineering

Professor Sreeram Kannan
Electrical Engineering

A recent surge in development of high throughput single cell transcriptome sequencing methods has given rise to single cell data from a variety of cellular contexts. Single cell data can provide a wealth of statistical information about biological processes which are not available through bulk measurement methods. However, single cell data produces a new set of computational challenges because of the inherent noisiness from biological and technical sources. At present, much of the analysis performed on single cell datasets is done with common biological/general purpose tools which are not designed for data containing these noise sources. This makes these learning/inference algorithms highly unsuited for these datasets. Here, we have developed tools specifically designed to study single cell sequenced data and used them to study specific biological problems.

Firstly, we have developed a pre-processing tool called UNCURL which uses a sampling distribution aware approach to estimate the true transcriptomic state of a cell from the heavily sampled observed (single cell RNA-Seq or scRNA-Seq) data. We demonstrate that using the estimated states, instead of observed data, leads to improvements in the performance of downstream algorithms for clustering and lineage inference. UNCURL also allows

users to incorporate available qualitative prior information into the state estimation process, resulting in further enhancements in the performance of downstream algorithms.

Next, we developed PIPER, a method that utilizes differential network analysis on scRNA-Seq data from biological progressions (such as differentiation) to identify the key regulator genes of these processes. PIPER uses a network inference algorithm that is specifically designed for highly sampled count valued data which outperforms commonly used methods. We show that PIPER correctly identifies known key regulators of several biological processes, including the temporal/pseudo-temporal order of their action. PIPER also makes several interesting predictions about genes which can provide starting points for future experimental studies.

Finally, we demonstrate an application of single cell data sources in studying a particular gene network motif. Micro-RNA based incoherent feed forwards loops (IFFLs) have been demonstrated in the past to have biological noise reduction properties. In this work, we study the specific mechanism of their noise reduction property by analyzing their effect on specific components of noise: extrinsic and intrinsic noise. Our study demonstrates that IFFLs increase high frequency noise at the mRNA level, which in turn leads to lower overall noise at the protein level because of the time scale difference between transcription and translation.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

I would like to thank my advisors, Dr. Georg Seelig and Dr. Sreeram Kannan, for their tremendous support and guidance throughout my PhD. I appreciate the biological and theoretical perspectives they each brought to my work. I am especially grateful to Dr. Seelig for allowing me to explore my ideas even when they seemed outside the area of both our expertise! Without this freedom, I would never have pursued computational biology, nor would I have had the opportunity to work with my (now) co-advisor Dr. Kannan. I would also like to thank my collaborators, Dr. Su-In Lee and Dr. Abhyudai Singh, for their help with the work presented in Chapters 4 and 5, respectively. Dr. Lee's exceptional class first introduced me to computational biology and focused my future research.

I am grateful to all of my current and former labmates from Seelig lab: Dr. Alex Rosenberg, Dr. Yuan-Jyue Chen, Dr. Sherry Chen, Dr. Richard Muscat, Dr. Ben Groves, Dr. Sergii Pochekailov, Dr. Alberto Carignano, Dr. Anna Kuchina, Dr. Nick Bogard, Dr. Gourab Chatterjee, Dr. Paul Sample, Sundipta Rao, Sifang Chen, Randolph Lopez, Arjun Khakhar, Alex Baryshev, Ban Wang, Charlie Rocco, Yue Zhang, Johannes Linder, and Erin Wilson. Alex (Rosenberg) initially encouraged me to transition from synthetic biology to computational biology and introduced me to single cell sequencing. Alberto helped me with stochastic modeling, and I enjoyed many laughs at his expense. Working with Yue improved my programming and allowed me to realize the importance of writing good code. Randolph and Arjun listened to the early stages of my ideas. Sifang, Sergii, and Chuhern Hwang (from Carothers lab) always kept me entertained through many interesting conversations on topics ranging from colonizing Mars to U.S. politics. Though I have not spent as much time with Dr. Kannan's lab, I have had very fruitful discussions with Sudipto Mukherjee, Arman

Rahimzamani, and Joshua Fan (with whom I have co-authored a paper).

My best friend Rajaditya, who is a PhD student himself, supported me tremendously in both my academic and personal life. My parents, my wife, and my family have always been there for me. Without their faith in my abilities, I may never have gotten this far.

DEDICATION

To my dear wife, Ladan, and my inspiring parents, Ratna and Samir.

Chapter 1

INTRODUCTION

Single cell assays such as scRNA-Seq and fluorescent in-situ hybridization (FISH) are becoming increasingly popular tools for the study of expression changes in cells during various biological processes. Conventional methods of obtaining such data rely on the use of microarrays and qPCR which provide the average expression level across a large population of cells. These methods are unable to analyze the differences between sub-populations in the cell samples, leading to mis-identification of statistical properties due to the phenomenon known as Simpson's paradox [5] (a simple illustrative figure can be seen in Figure 1.1). For many applications (e.g. comparing expression levels across species or obtaining disease biomarkers), bulk data sources often provide enough information. However, many important biological questions cannot be answered from aggregate data. Many complex tissues, such as the brain, have multiple distinct cell types in which expression levels vary widely and cannot be determined through aggregate data [6]. Single cell data is also very important in capturing the stochasticity in gene expression. Due to the rising needs of the mentioned applications and decreasing costs of sequencing [7], there has recently been a massive surge in the development of many high throughput single cell transcriptomic sequencing methods [6, 8, 9]. The data produced by these methods has led to important insights about novel cell types [6], key markers of various differentiation processes [1], and gene expression changes during development [10].

While the development of these low cost/high throughput methods has given rise to many large single cell datasets, these analyses produce a set of computational obstacles that are uncommon in other forms of biological data. The barriers posed by these datasets can be summarized in two broad categories: 1) challenges pertaining to the properties of the single

cell datasets and 2) challenges pertaining to novel applications unique to single cell data. The former set of challenges are associated with the single cell sequencing data generation scheme, making standard tools unacceptable for these datasets. An example problem of this type is the problem of differential expression analysis. While standard tools exist to perform differential expression analysis, their performance on single cell datasets is often much worse due to the discrete nature of these datasets. This has led to promising work on differential expression analysis tools specfic to such datasets [11]. The latter category of challenges pertains to single cell data applications such as the inference of spatial location of sequenced cells. While single cell sequenced data can potentially provide a wealth of information about transcriptomic signatures of spatial locations within tissues, it is often difficult to ascertain the exact spatial location of a sequenced cell. This problem, which is unique to single cell sequenced data has led to the development of various novel computational tools [12, 13]. A comprehensive survey of computational challenges in single cell sequencing can be found in [14].

In this thesis, we highlight our contributions in developing novel computational tools which aim to solve problems belonging to both of these categories. We also demonstrate the usefulness of single cell data in solving problems in mathematical biology, specifically in the study of signal processing properties of specific network motifs. Our specific contributions to each of these areas have been outlined below:

Contributions to cell state identification and related applications: Single cell RNA-seq (scRNA-seq) data contains a wealth of information which has to be inferred computationally from the observed sequencing reads. As the ability to sequence more cells improves rapidly, existing computational tools suffer from three problems. (1) The decreased reads-per-cell implies a highly sparse sample of the true cellular transcriptome. (2) Many tools simply cannot handle the size of the resulting datasets. (3) Prior biological knowledge such as bulk RNA-seq information of certain cell types or qualitative marker information is not taken into account. Here we present UNCURL, a preprocessing framework based on non-

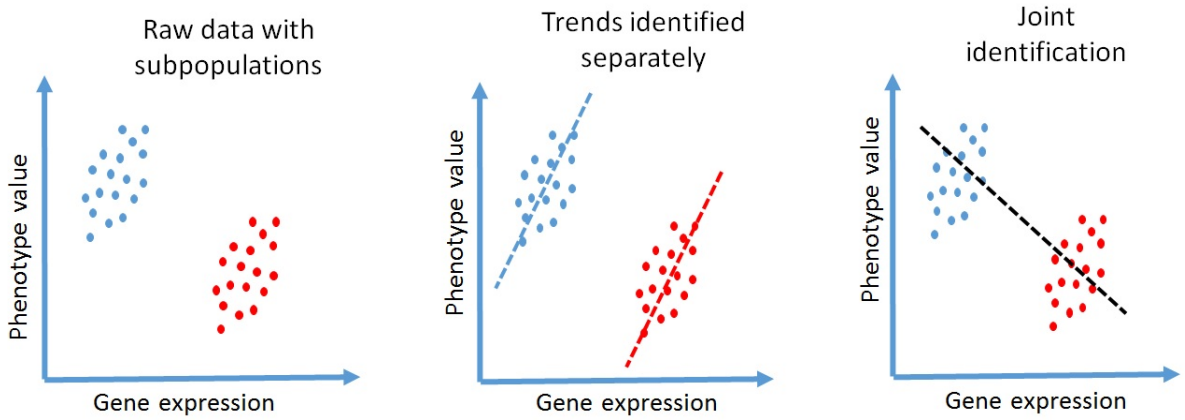


Figure 1.1: An illustration of the Simpson's paradox. When the data sample comprises of multiple sub-groups with different trends, joint identification of the trends of the dataset can lead to an estimated trends which are not representative of any sub-population in the dataset.

negative matrix factorization for scRNA-seq data, that is able to handle varying sampling distributions, scales to very large cell numbers and can incorporate prior knowledge.

We find that preprocessing using UNCURL consistently improves performance of commonly used scRNA-seq tools for clustering, visualization, and lineage estimation, both in the absence and presence of prior knowledge. Finally we demonstrate that UNCURL is extremely scalable and parallelizable, and runs faster than other methods on a scRNA-seq dataset containing 1.3 million cells.

Contributions on differential network analysis for the identification of master regulators: Identifying the gene regulatory networks that control development or disease is one of the most important problems in biology. Here, we introduce a computational approach, called PIPER (PROGRESSIVE network PERturbation), to identify the *perturbed genes* that drive differences in the gene regulatory network across different points in a biological progression. PIPER employs algorithms tailor-made for single cell RNA sequencing

(scRNA-seq) data to jointly identify gene networks for multiple progressive conditions. It then performs differential network analysis along the identified gene networks to identify master regulators. We demonstrate that PIPER outperforms state-of-the-art alternative methods on simulated data and is able to predict known key regulators of differentiation on real scRNA-Seq datasets.

Contributions on understanding the mechanisms of noise rejection in biological network motifs: MicroRNA-based feedforwards loops (IFFLs) are recurrent network motifs in mammalian cells and have been a topic of study for their noise rejection and buffering properties. Previous work showed that IFFLs can adapt to varying promoter activity and are less prone to noise as compared to similar circuits without the feedforward loop. This work studies the mechanisms that lead to noise rejection properties for this and other micro RNA based IFFL network motifs. The effects of microRNA-induced degradation of transcripts and translational inhibition on noise rejection are compared. The mathematical observations suggest that translational inhibition provides a more effective mechanism for noise rejection and buffering. In order to better understand the origin of the noise suppression observed in IFFLs, this work studies how the system responds to the two constitutive components of noise: intrinsic and extrinsic noise. The results indicate that while microRNA-mRNA interaction can cause an increase in mRNA noise for certain parameter regimes, it leads to translational inhibition, reducing noise at the protein level. For intrinsic noise, we observe the effect of increasing mRNA noise primarily translates to increased high frequency fluctuations and reduced low frequency fluctuations. The higher frequency fluctuations are difficult for the much slower translation machinery to follow; hence, protein noise is rejected more effectively.

The thesis is organized in the following manner:

- Chapter 2 contains required background material needed to understand our technical

contributions.

- Chapter 3 contains the details of our contributions to cell state identification and its related applications.
- Chapter 4 contains the details of our contributions to differential network analysis.
- Chapter 5 contains the details of our contributions to understanding the noise reduction properties of microRNA-based IFFLs.
- Chapter 6 discusses conclusions and potential future directions that stem from this work.

Chapter 2

BACKGROUND

While the various chapters have been written to be self sufficient in terms of methods, some background material on machine learning and mathematical modeling is useful to help understand the problems tackled in the various chapters. In the following sections we describe some of the important background material required. Section 2.1 is mostly related to Chapter 3 but parts of it are also relevant to Chapter 4, Section 2.2 is relevant to Chapter 4 and Section 2.3 is relvant to Chapter 5.

2.1 Unsupervised learning background

2.1.1 Clustering

Clustering is a common term used to describe a large set of unsupervised learning methods used to find groups in data. The goal of clustering is to partition the data into groups that have similar 'features' under some distance metric. Many approaches have been proposed to solve this particular problem, the notable among which are hierarchical clustering, k-means clustering, k-mediod clustering, Gaussian mixture models etc. While these algorithms share the goal of partitioning the data set into groups based on their similarity, their approach to the problem vary substantially. In this work we will mostly make use of k-means clustering but we briefly describe the other methods too for background:

- **Connectivity-based clustering:** Hierarchical clustering is an example of this form of clustering. The core idea of this form of clustering is that objects are more related to similar objects (based on some distance metric) than ones that aren't as similar. Clusters of objects are formed based on their distance from other objects in the cluster

and that from objects outside the cluster. A cluster is described by the maximum distance between the various objects in the cluster and given different threshold maximum distances, different clusters are formed. The output of the hierarchical clustering algorithm is in the form a dendogram as seen in Figure 2.1. This clustering approach does not need the user to provide the expected number of clusters in the data, but simply the maximum threshold distance.

- **Distribution-based clustering:** The Gaussian mixture model is an example of this form of clustering. This algorithm assumes that the data can be described by a mixture of multivariate Gaussian distributions. Each object in the dataset is assumed to have a vector of probability values for belonging to one of the Gaussian distributions. The clustering algorithm tries to identify the vector of probabilities for each data point along with the parameters of the various component multi-variate Gaussian distributions. Unlike heirarchical clustering, this algorithm expects the user to provide the expected number of Gaussians in the mixture model.

- **Centroid-based clustering:** An example of this class of clustering methods is the k-means clustering. This class of clustering problems involves finding k centroid vectors (which may or may not include be a part of the dataset) and then grouping the data points into disjoints sets based on their distance from their closest center. This problem is explained in greater detail in the following paragraph.

To define the k-means algorithm, we define a dataset $X \in \mathbf{R}^{N \times n}$ comprising of the points $(x_1, x_2, ..., x_n)$. The objective of k-means clustering is to partition the data into k sets $\mathbf{S} = S_1, S_2, ..., S_k$ such that the intra-cluster distance is minimized. This is equivalent to solving the following optimization problem:

$$arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{x \in S_i} ||x - \mu_i||^2 \tag{2.1}$$

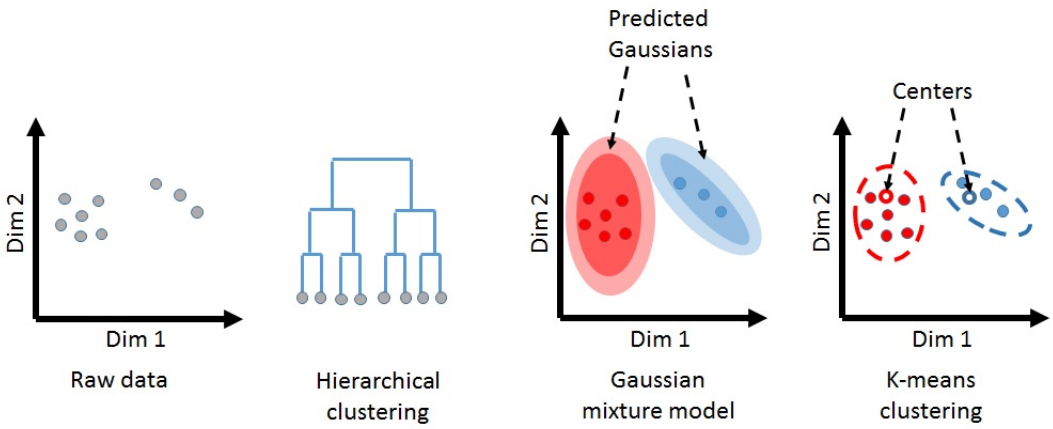


Figure 2.1: Some common clustering algorithms

Where μ_i is the sample mean of the set S_i. Here the distance used is the 2-norm distance. This can be further expanded to any valid distance function. A more detailed treatment of most of these algorithms can be found in [15].

2.1.2 Non-negative Matrix Factorization

A closely related problem to clustering is the non-negative matrix factorization (NMF). NMF is an approach to factorize the data matrix $X \in \mathbf{R}^{N \times n}$ into non-negative matrices $W \in \mathbf{R}^{N \times k}$ and $V \in \mathbf{R}^{k \times n}$ as seen in Figure 2.2. This problem is particularly useful in various application areas such as recommender systems, document clustering and image segmentation, where the non-negativity is particularly meaningful.

While the exact factorization is only solvable in special cases [16], an approximate solution for this problem is obtained by solving the following minimization problem:

$$arg \min_{W \succ 0, V \succ 0} ||X - WV||_F \tag{2.2}$$

Where $||.||_F$ denotes the Frobenius norm. The NMF method has an inherent clustering property and can be reduced to the k-means clustering in a special case by adding the constraint $VV^T = I$. It must be noted here that quite like the k-means clustering problem,

the optimization for NMF is not a convex problem. A comprehensive review of this problem and some common algorithms employed in solving the optimization problem can be found in [17].

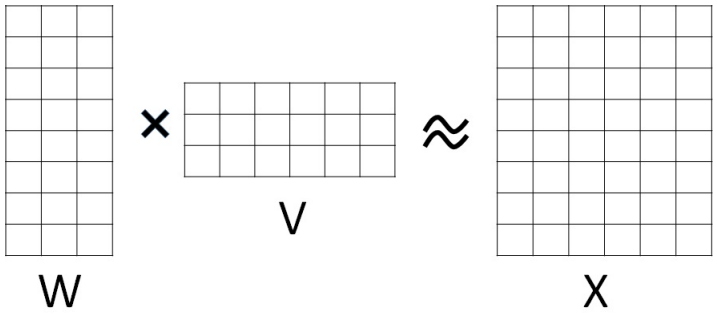


Figure 2.2: Non-negative matrix factorization schematic

2.1.3 Dimensionality reduction

This problem is also called 'low dimensional embedding'. The key goal of all algorithms of this class of problems is to reduce the number of features used to describe the data i.e. to develop a mapping $f : \mathbf{R}^N \to \mathbf{R}^p$, where $p \leq N$. This mapping however, is usually required to preserve the relative distance between points under some definition of distance. Formally, given two metric spaces (X, d) and (X', d'), the map $f : X \to X'$ is called an embedding. This embedding is called distance-preserving if it satisfies the following condition:

$$d(x, y) = d'(f(x), f(y)) \tag{2.3}$$

While there are conditions under which it is possible to find distance-preserving embeddings [18], oftentimes the more realistic goal is to find embeddings which have the lowest distortion i.e. $d(x, y)$ is approximately equal to $d(f(x), f(y))$ for either all points in the dataset or for points close to each other. A simple example of low dimensional embedding using some common techniques is seen in Figure 2.3.

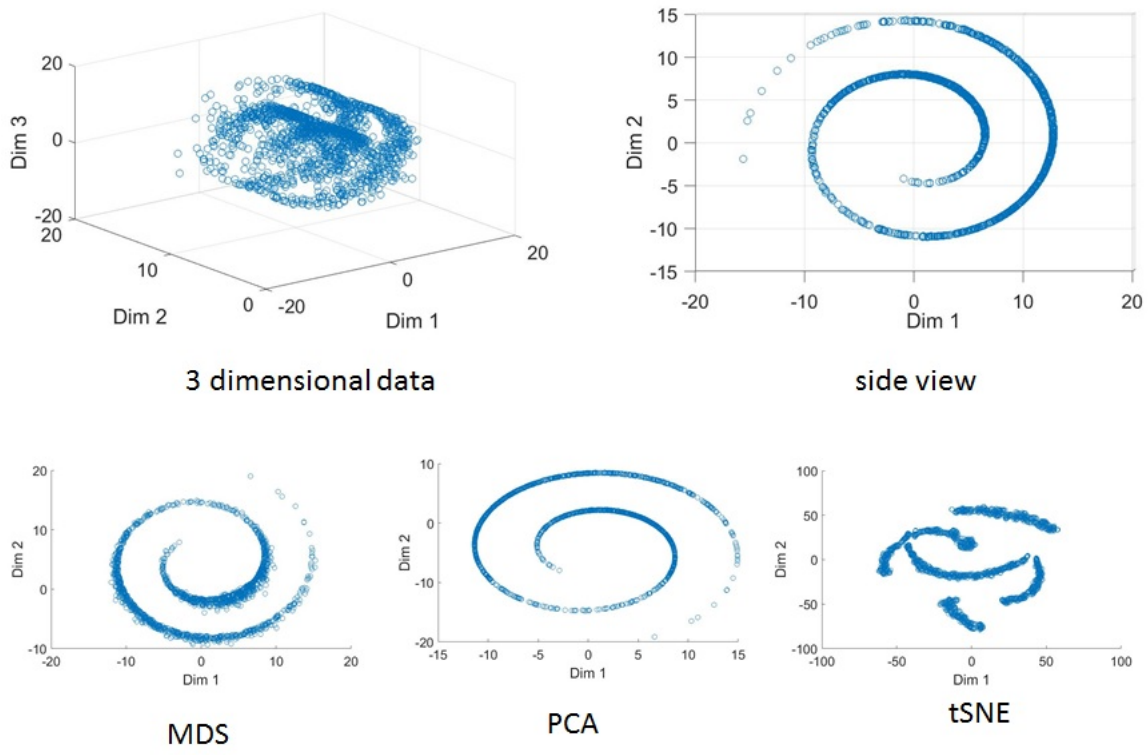


Figure 2.3: Low dimensional embedding of high dimensional data with various algorithms

While there has been a lot of work on dimensionality reduction, only the ones that will be encountered in the following chapters are briefly described here:

- **Principal Component Analysis (PCA):** PCA is one of the most commonly used dimensionality reduction technique in a variety of application areas. Mathematically, PCA is equivalent to finding an orthogonal linear transformation such that, in the new representation, the first dimension has the highest variance, the second dimension has the second highest variance and so on. These dimensions are called the principle components of the data. By choosing the first p principle components in the new representation, the dimensionality of each data point is effectively reduced to $\mathbf{R}^p$. This method is usually chosen to retain the variance of the data while reducing the number

of dimensions. It does not explicitly try to prevent distortion of distances in the new representation.

- **t-distributed stochastic neighbor embedding (t-SNE):** tSNE is an increasingly popular non-linear dimensionality reduction method which aims to place points that are similar to each other in the high dimension, close to each other in the low dimension. This is done by constructing two different probability distributions between pairs of points in the high dimension and those in the low dimensional representation. These probability distributions are constructed in a way such that p_{ij} is high if x_i and x_j are very similar and very low if they are not. The objective of this algorithm is then to identify points in the low dimension such that the *Kullback-Leibler divergence* (used to measure similarity between two probability distributions) is minimized between the two probability distributions.

- **Multidimensional scaling (MDS):** MDS is one of the older algorithms for dimensionality reduction. Unlike the two previous algorithms, this algorithm requires the pairwise distance between the points in the high dimension and not the points themselves. The algorithm then tries to find a low dimensional representation such that the distances between pairs of points in the high dimension and approximately equal to those in that in the low dimension.

- **Locally Linear Embedding (LLE):** LLE aims to preserve the local distances between points in the lower dimension. This is done by first calculating the k neighbors (where k is a user specified parameter) of each data point in the high dimension. Each point's neighbors are then used to linearly approximate it by estimating linear weights for each neighbor. This is done for all points to construct a weight matrix $W \in \mathbf{R}^{n \times n}$, where n is the number of points in the dataset. W_{ij} is non-zero only if point x_j is one of the k neighbors of point x_i. The next step is to estimate a lower dimension representation for which the same W can linearly approximate the lower dimensional

data points. By restricting each row in W to have at most k non-zero elements, this algorithm seeks to only preserve local distances in the lower dimension.

A much more detailed treatment of these algorithms can be found in [19, 20].

2.1.4 Lineage estimation

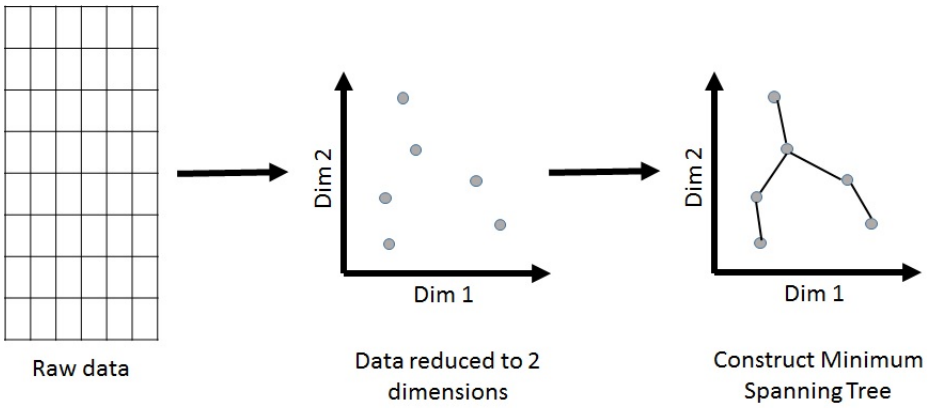


Figure 2.4: Lineage estimation workflow for Monocle [1], a commonly used lineage estimation algorithm

Unlike the previous learning problems, lineage estimation is a problem specific to biology. The goal of lineage estimation algorithms is to estimate the position of a cell in a biological progression event such as differentiation. To do this, let us first specify the problem in terms of biological variables. We assume we are provided with a dataset $X = (x_i, x_2, ..., x_n)$ where each point $x_i \in \mathbf{R}^N$ represents the expression of N genes for the ith cell. Each cell is then assumed to be at some state s_i which can be considered a latent variable for our problem. The key assumption of these algorithms is that the biological progression (say differentiation) is a smooth process. Moreover, the relationship between the true state and the observed data at state s is given by:

$$x(s) = g(f(s), \eta(s)) \tag{2.4}$$

Where, $f(s) \in \mathbf{R}^N$ is the actual gene expression vector at state s, $\eta(s)$ is the biological/technical noise vector at state s and $g : \mathbf{R}^N \times \mathbf{R}^N \to \mathbf{R}^N$ is some function that produces the observed gene expression values. The goal of lineage estimation is to approximate a smooth $f(s)$ given an un-ordered finite set of points $X = (x_1(s_1), x_2(s_2), ..., x_n(s_n))$. We will briefly discuss two of the commonly algorithms used for lineage estimation: Monocle [1] and SLICER [21].

- **Monocle:** Monocle is one of the earliest algorithms for lineage estimation. Monocle first uses Independecent Component Analysis (ICA) to reduce the dimensionality of the data to 2-dimensions, which helps reduce the influence of house-keeping genes. This step is followed by Minimum Spanning Tree (MST) construction to find a minimum weight tree connecting all points in the dataset. This is followed by the construction of a PQ-Tree, which helps identify the main and secondary backbones of the estimated lineage. A schematic of the algorithm can be seen in Figure 2.3.

- **Monocle2:** Monocle2 is an improvement over the original Monocle algorithm. It addresses the issue of noisyness in the observed data by iteratively smoothing the dimensionality reduction along the estimated trajectory. The initial steps of the algorithm strongly resemble Monocle, leading to an initial estimate of the trajectory. The low dimensional embedding is then updated using this estimated trajectory using an approach called reverse graph embedding. The trajectory is then estimated using the new low dimensional embedding. This process is then repeated until convergence.

- **SLICER:** SLICER first constructs an α- convex hull of the data to find the optimal number of neighbors (k) to use for low-dimensional embedding with LLE. Once k has been identified, it is used to reduce the dimensionality to 2-dimensions using the LLE method. This is followed by building another nearest neighbor graph in the low-dimension, which approximates the lineage. After this, there are additional steps to specify the starting cell and identifying the number of branches in the lineage.

While there isn't a comprehensive review on lineage estimation yet, a somewhat more detailed list of methods can be found in Chapter 3.

2.2 Network Analysis background

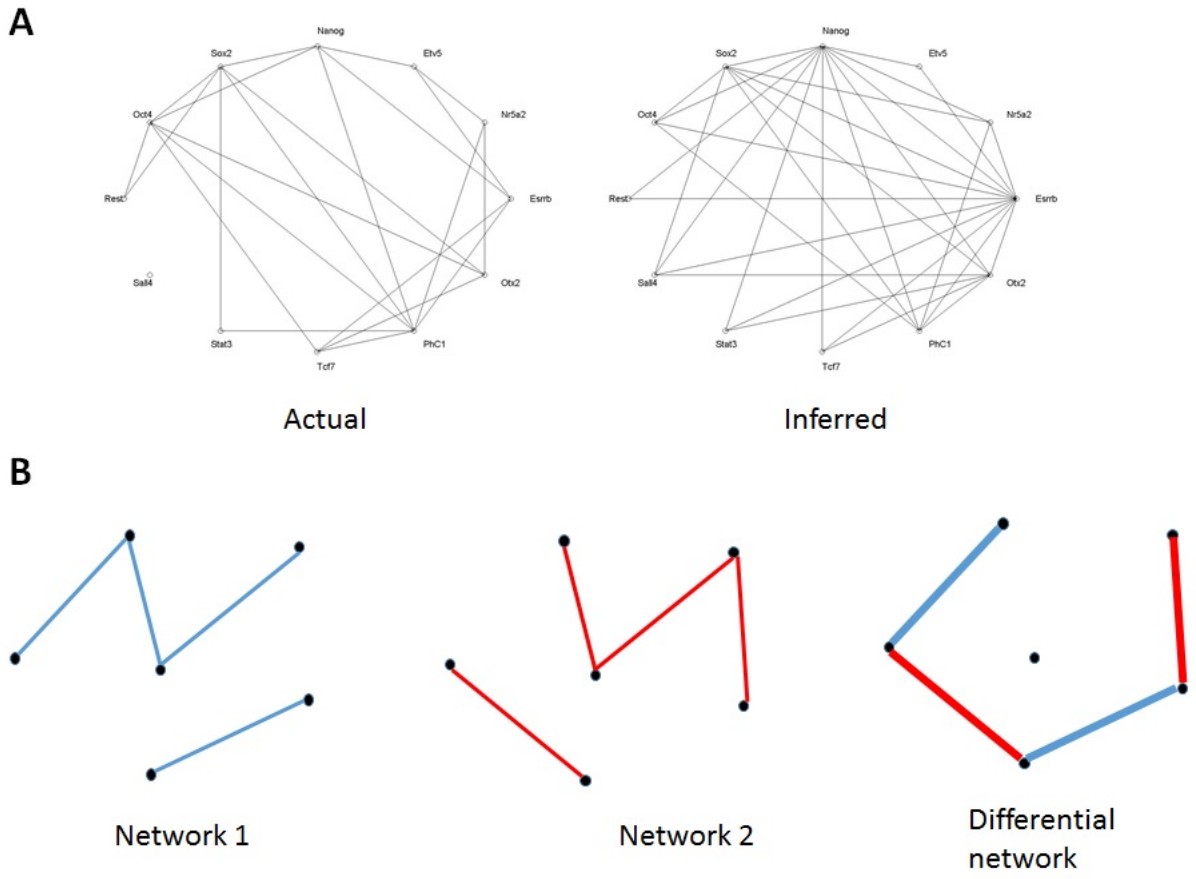


Figure 2.5: a) Illustration of simple network inference task using GGM. b) Schematic of a simple differential network obtained by finding the difference between two networks having the same vertices but different edge connectivity.

2.2.1 Network inference

Different cellular species such as different mRNA, proteins and DNA are constantly interacting with one another. These interactions cause the various species to regulate each other's production and abundance. An important area of study in computational biology is the analysis of how these species regulate each other. A common way to model these interactions is through a graph, where each node represents the individual species and the edges represent the connections between them. Based on the interaction types and the species, these networks can be classified into several types such as transcriptional regulatory networks, signal transduction networks, protein-protein interaction networks etc. While the functions of these biological networks can oftentimes be similar, the approaches employed to uncover the network structure often varies widely between the classes. Here we will mostly focus on the various approaches employed to identify transcriptional regulatory networks.

Transcriptional network identification comprises of predicting which transcription factors regulate which genes based on a set of observed values for each specie. The transcriptional regulation process comprises of various steps such as transcription of the transcripts, translation of the proteins and binding of the proteins to promoter regions of other genes, most of which cannot be simultaneously measured. A common approach is to first infer the conditional dependencies between the observed variables which can be then probed further using experimental methods or other computational strategies. Here we will discuss some of the commonly used methods to infer the structure of gene networks from transcriptomic data:

- **Correlation based co-expression networks:** This is one of the simplest methods to identify the structure of the transcriptional networks. The method is based on the premise that if a gene regulates another gene, then their expression values will be strongly correlated to one another. Hence, by calculating the pairwise correlations between all species under consideration, one is able to see how their expressions correlate with one another. Then a threshold correlation is selected by the user and all gene pairs with correlation values higher than the threshold are predicted to interact

while the ones with lower values are assumed to no interact. This method leads to the estimation of an undirected graph which can oftentimes have spurious connections owing to chains of interactions. Despite this, correlation networks are still widely used for the purpose of rough understanding of genetic interactions.

- **Mutual Information based co-expression networks:** This method is also known as ARACNE (Algorithm for the Reconstruction of Accurate Cellular Networks). It is very similar to the correlation based co-expression network with the primary difference being that ARACNE measures co-expression through Mutual Information (MI) of the species and not correlation. Once the mutual information is calculated, the user is expected to provide a threshold value for the MI, which is then used to determine the structure of the network. Like the correlation networks, ARACNE is also known to predict spurious linkages.

- **Gaussian Graphical Models (GGM):** Unlike the previous two methods, GGM's do not estimate co-expression networks but approximate the distribution of the species as a multi-variate Gaussian distribution, which leads to the identification of the network structure. GGM's rely on the knowledge that the inverse co-variance matrix (Σ^{-1}) captures the conditional dependency structure between genes which can be used to infer the structure of the network i.e. $[\Sigma^{-1}]_{ij} = 0$ implies that the ith and jth species are conditionally independent. This leads to the estimation of an undirected graph. One challenge for GGM's is that the sampe co-variance matrix Σ is rarely full rank and hence the inverse co-variance matrix has to be approximated from the data. This is one of the most popular methods for network inference presently and has given rise to many variations depending on the penalty function added to the approximate estimation problem. A common variant of the GGM is the joint-GGM which allows for the inference of several networks at the same time while forcing them to be similar to each other. This method is particularly useful while identifying the structure of the same network in multiple different conditions.

- **Bayesian networks:** While all of the previous methods identify undirected networks, one of the most formal ways to identify a directed network is through a Bayesian network model. Bayesian networks identify Directed Acyclic Graphs (DAGs) and lead to the estimation of the joint distribution of all the genes. Bayesian networks assume that given the parent nodes of a given node, it is independent of all nodes that are it's descendents (Markov assumption). Because of this property, each directed edge of a Bayesian network can be interpreted as a causal link. Despite this desirable property, the estimation of Bayesian networks is a particularly challenging problem. The estimation comprises of two steps: 1) Identifying the correct DAG to represent the data and 2) Estimating the model parameters for the given DAG. The first step of the process requires searching through all possible DAGs given the set of species, which is a computationally intractable problem. Several heuristic methods have been developed to tackle this problem but till now, Bayesian networks aren't as widely used as GGM and it's variants.

A more comprehensive survey of various gene network inference methods can be found in [22].

2.2.2 Differential network analysis

While the previous sub-section focuses on methods to identify the structure of gene interaction networks, these networks have been shown to vary significantly between different biological conditions [24, 25]. Differential network analsysis focuses on studying the changes in these networks between different biological conditions. While studying biological progressions such as differentiation, several studies have demonstrated that while some gene interactions are preserved between different conditions (often called housekeeping interactions), interactions involving key genes often undergo significant changes [24]. The study of dynamics of networks poses two challenges: 1) Identification of networks in different conditions and 2) Relating the changes in networks with the functional changes between the different conditions.

The simplest approach to identify the network structure at multiple conditions is to identify the networks separately at each condition. However, this approach leads to the identification of many spurious interactions and doesn't make use of all of the available data for each of the estimation tasks. A more commonly utilized approach is the joint estimation of networks with the optimization problem having an additional penalty term forcing the networks to be similar to each other [76]. While these approaches lead to identification of the network structure across multiple conditions, they do not automatically identify the functional changes in gene networks. A commonly used method is to study enriched gene ontologies of among the genes involved in the differential interactions [24]. A comprehensive study of methods commonly employed for differential network analysis can be found in [25].

2.3 Cellular variability and stochastic modeling background

2.3.1 Sources of cellular variability

Cells even within the same cell type exhibit a lot of temporal and cell-to-cell variability. A primary reason for this is the low copy number of various constituent components (gene, mRNA, protein etc.) which makes this stochasticity more evident even with the same clonal population. This is more visible because cellular components interact with each other, leading to the compounding of the stochasticity. There are several sources of variability in cells such as cell cycle effects, chemical gradients in the environment, stochasticity of chemical reactions between species, differences in cell sizes etc. These sources of variability are often grouped into two main types namely: Intrinsic noise and extrinsic noise. These terms were first introduced in‘ [27] which provides a comprehensive mathematical treatment of cellular noise.

Intrinsic noise refers to the noise originating from sources of variability that affect different components of a cell differently. The most common source of intrinsic noise is stochasticity of the chemical reactions that are part of the cellular mechanism, such as transcription and translation. Another common source of intrinsic noise is transcriptional/translational

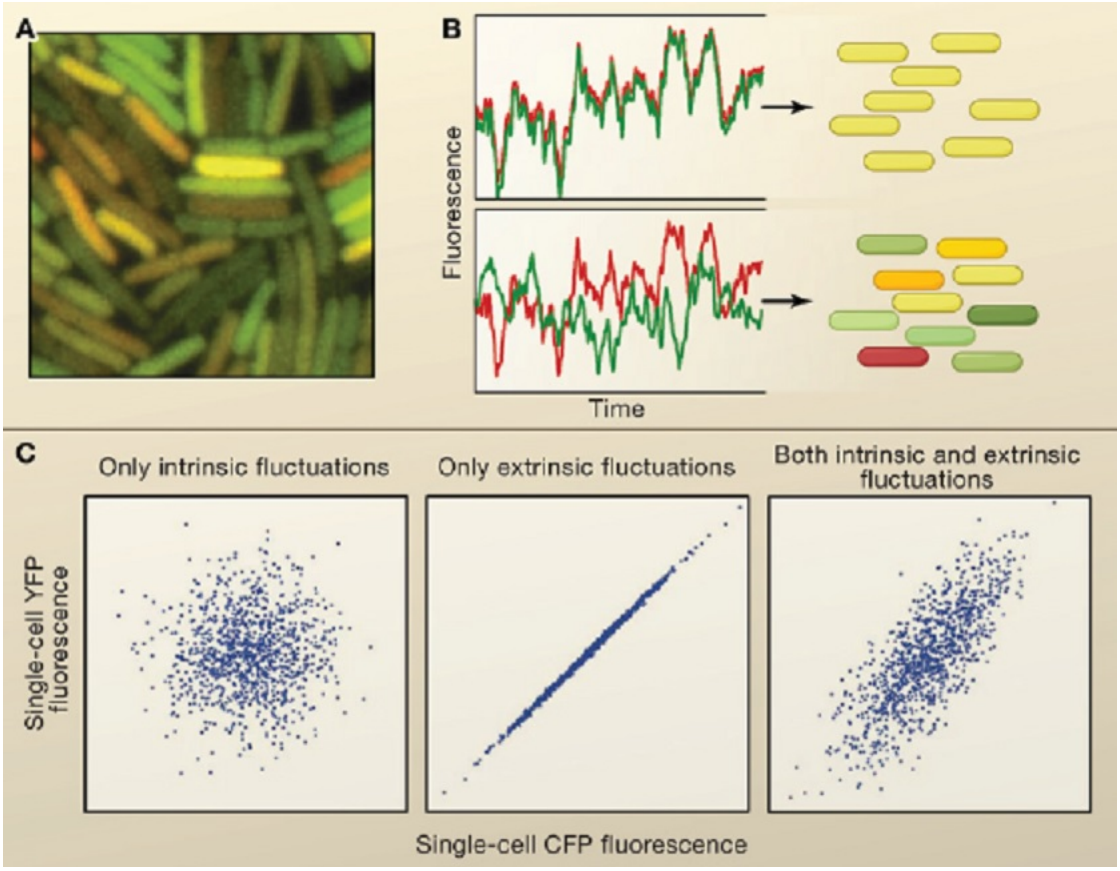


Figure 2.6: Figure demonstrating contributions of different sources of noise taken from [2]. a) Fluorescent imaging demonstrating variability between different cells in a population. Each cell has two fluorescent reporters YFP and CFP. b) Schematic showing extrinsic effect of intrinsic and extrinsic noise. Extrinsic noise causes YFP and CFP fluorescent to act synchronously, while intrinsic noise causes them to act asynchronously. c) Schematic of how a scatterplot of YFP and CFP fluorescent would look like.

bursting, which is the increased transcription/translation in bursts. Intuitively, intrinsic noise makes components of each individual cell act asynchronously.

Extrinsic noise refers to the noise originating from sources which affect different components of a cell synchronously. Common sources of extrinsic noise are difference in cell sizes,

number of ribosomes/RNA polymerases in each cell, difference in copy number of genes between cells, external chemical gradients etc. These sources of noise do not typically affect the synchronicity of the different components of a cell.

The total noise for each cell is a combination of the extrinsic and intrinsic noise combinations. Figure 2.6 demonstrates the contributions of different sources of noise, taken from [2]. In the following sub-sections, we explain the methods of mathematically model each noise source.

2.3.2 Intrinsic noise modeling

Before we begin the modeling of different noise sources, let us first define a quantitative measure for noise. The most common measure of noise is coefficient of variation (η), which is defined as:

$$\eta = \sqrt{\frac{Var(X)}{E[X]^2}} \tag{2.5}$$

Where, X is the random variable whose noise is being measured, $E[.]$ represents the expectation operator and $Var(.)$ is the variance.

There are many roughly equivalent methods for approximating the intrinsic noise such as the Stochastic Hybrid System (SHS) formulation, Markov processes and Linear Noise Approximation (also called the Fluctuation Dissipation Theorem). Here we will focus only on the modeling of noise using the Linear Noise Assumption since it is the method employed for intrinsic noise modeling in Chapter 5. LNA assumes that the stochastic process in consideration can be modeled as a combination of multiple discrete events, such as a series of chemical reactions. LNA then assumes that the volume of the system Ω is large and the term $\frac{1}{\sqrt{\Omega}}$ has a small value. This assumption implies that there are many molecules of each specie within the volume under consideration which motivates the use of the central limit theorem (CLT) to approximate their distribution. Hence, the LNA approximates each specie to be the sum of a scaled deterministic expected value and a smaller stochastic fluctuation

term given by:

$$X(t) = \Omega x(t) + \sqrt{\Omega}\eta_x(t) \tag{2.6}$$

Where, X is any specie, x is it's deterministic expected value and η_x is the stochastic perturbation term. This approximation then leads to the covariances of the stochastic terms to be as follows:

$$\frac{d\sigma}{dt} = A\sigma + \sigma A^T + B \tag{2.7}$$

Where σ is the matrix of covariances of the stochastic fluctuation terms, A is the Jacobian matrix of the dynamics of the system (derived from the ODE representation of the concentration system) and B is the diffusion matrix (to be defined later). An important property of the LNA is that the matrices A and B are themselves calculated independent of σ, which allows equation 2.7 to have a tractable solution for various linear and non-linear systems. This property is derived on the assumption that the fluctuations themselves are small compared to the average concentrations which leads to the alternate name FDT. The matrix A is defined as follows:

$$[A]_{ij} = \frac{\partial}{\partial x_j}\frac{\partial x_i}{\partial t} \tag{2.8}$$

Where the term $\frac{\partial x_i}{\partial t}$ is obtained from the ODE representation of the system. It is important to note that A matrix is a matrix of constants if the system can be expressed through a linear ODE system. The B matrix is then defined as follows:

$$[B]_{ij} = \sum_k v_{jk} v_{ik} R_k \tag{2.9}$$

Where, v_{ik} is the number of molecules of species i produced by reaction k and R_k is the rate of reaction k. The steady state covariance can then be solved by setting $\frac{d\sigma}{dt} = 0$ in equation 2.7 and solving for σ. This method has a closed form solution at steady state for all linear systems but the non-linear systems do not always have a tractable closed form solution. A common approach to get around this is to linearize the non-linear system around an equilibrium point and using LNA on the linearized system instead.

2.3.3 Extrinsic noise modeling

While the different sources of noise rarely act separately, it is often convenient to study their effects separately. Since extrinsic noise largely depends on the variability in number of cellular components such as RNA polymerases, Ribosomes and external inducer molecules in each cell, they can be thought to affect the rates of the different chemical equations. However, by isolating extrinsic noise from intrinsic noise, we can assume that the reactions inside each cell is deterministic with rates that are specific to that cell. Let us first define an ODE system to describe the system:

$$\frac{dX(t)}{dt} = f(X(t), R) \tag{2.10}$$

Where, f is some function of the states (of cellular species) X and rates R. To study the steady state extrinsic noise of a system, we first find the closed form solution in steady state $X^{eq}(R)$. The noise of species i can then be calculated by finding it's closed form value of the coefficient of variation, given by:

$$\eta_{X_i} = \sqrt{\frac{Var_R(X_i^{eq})}{E_R[X_i^{eq}]^2}} \tag{2.11}$$

However, this expression is difficult to find a closed form solution for, specially for non-linear systems but methods such as the Taylor-Delta expansion [28] are employed to find an approximate solution. In Chapter 5, we consider a simpler problem by only considering the effect of transcription factors (internal or external) on extrinsic noise calculation. A more detailed treatment of different sources of noise and their modeling can be found in [2, 27].

Chapter 3

UNCURL: SCALABLE PREPROCESSING FOR SPARSE SCRNA-SEQ DATA EXPLOITING PRIOR KNOWLEDGE

3.1 Introduction

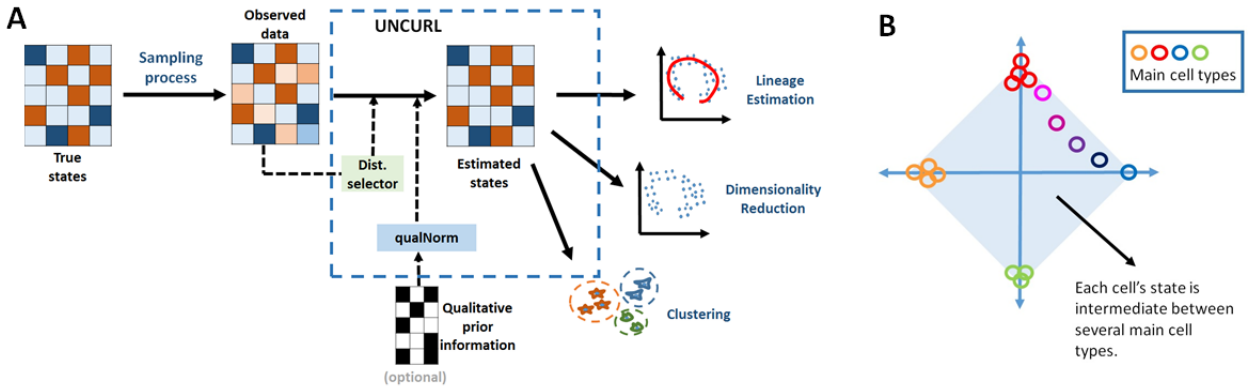


Figure 3.1: A) The primary input for UNCURL is the highly sampled single cell sequenced data and optionally any prior information that is known about the specific dataset. UNCURL then converts the observed sampled data to an estimated version of the true data using a novel sampling model aware matrix factorization. This can then be used in downstream unsupervised learning tasks. B) The convex mixture of cell states assume that all cell states lie in the convex hull spanned by a few extreme cell types.

High-throughput scRNA-seq technologies [9, 29–31] can provide biological insights such as revealing cell type composition [32, 33], cell lineage relationships [34–37], or even spatial relationships [38,39] between cells in heterogeneous multi-cellular systems. Enabling such insights are two key advantages of single cell transcriptomic datasets. First, having information

about individual cells helps avoid aggregation and conflation of traits from disjoint groups of cells within a mixed sample [40]. Second, scRNA-seq can generate a very high-dimensional dataset, both in terms of the number of cells and genes that can be assayed, compared to other methods with single-cell resolutions. However, advanced computational methods are required to extract latent biological information from the raw read-counts, which provide only a heavily sampled version of the full cellular transcriptome [41, 42].

Most commonly used computational tools for cell type identification [43, 44], lineage estimation [34, 36, 37] and similar applications rely on an initial dimensionality reduction step using methods such as PCA [45], LLE [46] or tSNE [47]. However, these algorithms assume that the underlying data is drawn from a Gaussian or a t-distribution, an assumption that does not always hold for scRNA-seq data [48]. The discrepancy between the assumed and actual distribution fundamentally limits the accuracy of the resulting predictions. In addition to such general purpose preprocessing methods, several tools were developed to specifically deal with scRNA-seq data [49–51]. However, these approaches do not scale well with increasing cell number. Finally, all existing methods rely almost exclusively on unsupervised learning and do not incorporate useful and commonly available prior information such as bulk gene expression data or cell type specific marker genes to guide the analysis process.

Here, we introduce UNCURL, a preprocessing framework for scRNA-seq data that addresses these shortcomings by estimating the true transcriptomic state of the cells prior to the sampling effect of RNA-seq. The preprocessed data from UNCURL can then be directly used as input by most major unsupervised learning algorithms commonly used in the context of scRNA-seq data. An overview of the algorithmic workflow of UNCURL can be seen in Fig 1A. The main technical contribution of UNCURL is a generalized non-negative matrix factorization (NMF) that explicitly accounts for the most likely sampling distribution of the dataset. Furthermore, UNCURL incorporates an accelerated optimization method tailored for sparse input data, so as to handle datasets with millions of cells efficiently.

Our algorithm exploits the low-dimensional nature of the true biological state matrix, i.e. it assumes that each cell is in a convex combination of a few archetypal cell-states. Under this

assumption, the true state matrix can be expressed as a product of an archetypal main state matrix, M, comprising of gene-expression in the archetypal states, and a matrix of mixing coefficients, W, a cluster-by-cell matrix for which each column sums to 1. We demonstrate that working with the estimated (and factorized) true state matrix considerably improves performance of state-of-the-art methods as compared to directly operating on the sequencing data.

Additionally, UNCURL allows for the integration of prior information which leads to large improvements in accuracy. To enable semi-supervised learning, UNCURL's toolbox contains a method (qualitative normalization, or qualNorm) for standardizing any prior biological information including bulk RNA-seq data, microarray data or even information about individual marker gene expression to a form compatible with scRNA-seq data. We demonstrate that initialization using prior knowledge in an appropriately standardized manner dramatically improves performance compared to unsupervised learning.

3.2 Methods

3.2.1 State Estimation

Procedure

An implicit assumption shared by many scRNA-seq data analysis tools is that any biological sample contains a limited number of cell types and that any individual cell can be considered a mixture of these cells. Here, we make this convex mixture model explicit, which leads to a model similar to NMF. NMF is classically used when the entries have Gaussian noise [52] and has been found beneficial in analyzing gene expression data gleaned from microarrays [53]. In scRNA-seq, the sequencing process can produce noise following several different distributions such as Gaussian, log-normal, Poisson and Negative Binomial, potentially with zero-inflation [51]. While NMF can be directly applied to the scRNA-seq data [54], utilizing the sampling distribution becomes critical especially when the number of reads per cell is small. While the sampling distribution is carefully modeled in differential expression studies [55], the

most commonly used algorithms for visualization, cell-type identification as well as lineage estimation do not account for this model. Thus, while factoring the matrix, we need to account for the sampling distribution in order to estimate the true cell-state matrix and mixing coefficients accurately from the observed gene expression matrix.

We assume that we are provided with a data matrix $X \in \mathbb{R}^{n \times d}$, where n is the number of genes and d is the number of cells, and k, the number of cell types. Let $X_{g,c}$ denote the count measured for gene g in cell c, and let $X^{\text{true}}_{g,c}$ be the relative abundance of gene g in cell c. We assume that the true matrix has a non-negative decomposition into two factors, i.e., $X^{\text{true}} = M \times W$. Here, $M \in \mathbb{R}^{n \times k}$ is the matrix of cell type means of the k-archetypal states with $M_{g,j}$ denoting the expression of gene g in archetype k. $W \in \mathbb{R}^{k,d}$ is the mixture parameter matrix which stores each cell as a convex combination of the archetypes, i.e., $W_{j,c}$ is the contribution of archetype j to cell c. Thus W satisfies $\mathbf{1}^k W_{j,c} = 1$ and $W_{j,c} \geq 0$.

Since we do not observe directly the relative abundance but only a sampled version, we assume that there is a channel that connects the true abundance to the observed abundance, $\mathbb{P}(x|x^{\text{true}}, \theta)$, i.e, given a value of x^{true} there is a certain distribution on x with some parameters θ. Note that we have not used a subscript for g and c to emphasize that the same distribution is used for all genes. We give three examples here, but our framework works with general distributions: (1) $\mathbb{P}(x|x^{\text{true}})$ is a Gaussian distribution with mean x^{true} and a fixed variance 1 (say). (2) $\mathbb{P}(x|x^{\text{true}})$ is a Poisson distribution with mean x^{true}. (3) $\mathbb{P}(x|x^{\text{true}})$ is such that $\log(x)$ is a Gaussian with mean $\log(x^{\text{true}})$ and variance 1. Our goal now is to maximize the log-likelihood of the observed data matrix X, by finding the optimal M and W. Note that $\mathbb{P}(X|M, W, \Theta) := \prod_{g,c} \mathbb{P}(X_{g,c}|X^{\text{true}}_{g,c})$.

This problem is non-convex but the sub-problems of estimating either M or W with the other matrix fixed are convex problems for many common sampling distributions, including the ones mentioned above. We thus utilize an alternating maximization algorithm to estimate these model parameters as follows:

$W = \text{argmax}_W \log(\mathbb{P}(X|M, W, \Theta))$ subject to $W_{j,c} \geq 0$, and

$M = \text{argmax}_M \log(\mathbb{P}(X|M, W, \Theta))$ subject to $M_{g,j} \geq 0$.

We repeat these two steps iteratively till convergence or till a maximum number of iterations. Once converged, we normalize the columns of W to sum to 1 to ensure the condition $\mathbf{1}^k W_{j,c} = 1$ is satisfied. We note that each of the steps is convex for many distributions and can be solved by gradient descent. Unlike the Gaussian and log-normal, Poisson Log-Likelihood does not have closed form solutions for even the sub-problems i.e. identifying M and W. Hence, Poisson state estimation requires the use of gradient descent based strategies, most of which have sub-linear convergence guarantees only for functions with Lipschitz continuous gradients. This is not true for the Poisson Log-Likelihood. However, [56] utilized a different definition of smoothness and derived a generalized algorithm (NoLips) that is capable of achieving a sublinear rate of convergence for a class of non-Lipschitz continuous functions including the Poisson Log-Likelihood. Here we use a custom alternating minimization approach using the NoLips algorithm to optimize the Poisson Log-Likelihood with additional modifications to allow for faster computation for sparse matrices and ability to parallelize the computation (see supplementary materials).

State estimation for different distributions:

While UNCURL can easily be extended to different sampling distributions, here we have limited ourselves to three of the most common ones, namely: Gaussian, log-normal and Poisson. It is easy to see that the state estimation problem for the Gaussian distribution, if variances are treated as uniform, is identical to the Non-Negative Matrix Factorization (NMF) problem. Thus, we utilize standard NMF solvers for this distribution followed by the column normalization of W as stated above. Similarly, for log-normal data, transforming the data as $Y = \log(1 + X)$ makes the transformed dataset Gaussian distributed and allow us to again use NMF solvers. It has been our observation that column normalizing the Gaussian and log-normal distributed datasets before state estimation can lead to an improvement in results and hence has been performed in this thesis.

3.2.2 Distribution selection

In our program we have a set of possible distributions to choose from. In order to select the distribution to use automatically, we implemented a method using fit error (Fig 2A). First, we fit the different distributions for each gene using maximum likelihood. Then, we compute the distance between the empirical distribution and each of the fitted distributions. This test is similar to the root-mean-square statistic for goodness of fit [57]. The distribution with the minimum such distance is considered the best fit. Finally, we output the best estimation fraction vector which captures the fraction of genes for which each distribution is the best-fit distribution. This is meant to be a guideline for the selection of the sampling distribution during the matrix factorization process.

3.2.3 Initialization for state estimation

Since state estimation is a non-convex problem, its result depends greatly on the initialization. Two commonly used methods for NMF initialization are based on k-means and SVD (singular value decomposition), respectively [58, 59]. In UNCURL, we have two ways to initialize the state estimation: (1) distribution-specific k-means initialization (2) truncated SVD + k-means based initialization. We describe our distribution specific K-means in the rest of the section, particularly for the Poisson case.

In the following sub-sections we develop an explicit framework to utilize prior biological information to initialize the matrix factorization. This relies on clustering each gene into clusters of high and low expression, which is done using k-means clustering for Gaussian and log-normal (after log-transformation) distributions. Here we outline a similar procedure for the Poisson distribution that allows our approach to be consistent across different distributions along with an approach to initialize the clustering.

Poisson k-means++:

k-means++ [60] is a well known seeding method for the k-means clustering algorithm, which tries to identify k points in the data with the highest mutual separation. While popular, its use of Euclidean distances between points makes it a poor fit for non-Gaussian distributed data. To use a similar approach for Poisson sampled data, we use a new distance metric. One intuitive distance would be the Poisson log-likelihood with one data point as the mean (say y) and the other as the point being considered (say x). Let ($ll_p(x|y)$ be defined as the log likelihood of a Poisson distribution with mean y. However, $ll_p(x|y)$ is not symmetric, which is required for a distance measure. We create a normalized version of this function which satisfies all properties of a distance measure:

$$\begin{aligned} d(x,y) &= ll_p(x|x) + ll_p(y|y) - (ll_p(x|y) + ll_p(y|x)) \\ &= (x-y)\log(\frac{x}{y}) \end{aligned}$$

This distance is based on the observation that value of $ll_p(x|y)$ is maximum when $x = y$. Thus, the $d(x, y)$ quantity measures the distance from the maximum value log-likelihood value for both x and y (for the sake of symmetry). This distance then replaces the Euclidean distance used in the standard implementation of k-means++.

Poisson k-means:

Poisson k-means is very similar to the classical k-means clustering with the difference being in the underlying distribution of the data; it is essentially the hard EM algorithm applied with a Poisson distribution. As with state estimation, we assume we are provided the expected number of cell types k and the data matrix $X \in \mathbb{R}^{n\times d}$. The first step of the algorithm involves calculating the Poisson log-likelihood for each cell given a set of means ($M \in \mathbb{R}^{n\times k}$), representing k cell types and then assigning each cell to the cell type for which it has the highest log-likelihood. The logarithm of the probability that cell c with observed counts $X_{.c}$ is sampled from the type i with gene expression $M_{.i}$ is then $ll_p(c|i) = -\sum_j [M]_{ji} + [X]_{jc}\log([M]_{ji})$.

This is called the E step of the algorithm, which partitions the cells into distinct types. This is followed by the M step, where we find the means for each cell type that maximize the log-likelihood. For the case of the Poisson distribution, this is simply the arithmetic mean of the data given by:

$$M_{g,i} = \frac{1}{|S_i|} \sum_{c \in S_i} X_{g,c} \quad \forall g$$

Here, S_i is the set of cell indices for which the log-likelihood is highest for the ith mean. The M step creates a new estimate of the means, which are then used to redo the E step. This procedure is repeated till convergence or until a maximum number of iterations.

3.2.4 Qualitative semi–supervision with QualNorm

In many scenarios where scRNA-seq is carried out, there is a wealth of prior knowledge. For example, there may be FISH images or bulk gene expression data measured through microarray or RNA-seq. Alternatively, marker genes may be known for a subset of cell-types. Two major issues in using such information are the incompatibility between different data types (e.g. FISH images or microarray data with RNAseq data), and variability between experiments using the same technique (e.g. bulk RNA-seq batch effects). We develop a method to specifically account for such variations in order to leverage this prior information. A key benefit with our method is that since the prior information is leveraged in UNCURL preprocessing, it can boost the performance of downstream methods not designed to utilize such prior information.

A basic problem that we need to solve is in deciding how to incorporate such differing type of information into our framework. Our hypothesis is that even though the particular quantitative measurements may not transfer well to scRNA-seq, the qualitative information inherent in such a dataset can be exploited. To do that, we first convert the available prior information into a binary matrix, that can then be imported into our method. In some cases, the prior biological knowledge available may be marker information which is directly in the binary format. In other cases, where prior biological knowledge is available as bulk

gene expression or other real-valued measurements, we first binarize the gene expression by thresholding around the central value for each differentially expressed gene. In our experiments, we have used the Poisson version of t-test [61] to identify genes that are composed of two separate distributions and have limited the input to only include up to 25 'ON' genes per cell type with the highest p-values. Details of this process are described in the Supplementary Methods.

The inputs to the framework are the following: 1) A single cell sequenced data matrix $X \in \mathbb{R}^{n \times d}$, 2) the number of cell types expected in the data, k and 3) a binary matrix of dimension $B \in \{0,1\}^{n_0 \times k_0}$, where n_0 is the number of genes for which the information is provided and k_0 is the number of cell types for which the information is provided. We note that the number of cell types k_0 for which prior information is available can be lesser than the number of cell-types k, and the number of genes n_0 for which prior information is available can be lesser than total number of genes in the data n.

Now, the main algorithmic problem is how to utilize the matrix B in solving the state-estimation problem. The obvious approach is to utilize the matrix as an initialization for state estimation. However there are three bottlenecks in our problem. (1) The matrix is binary, not real valued and it is unclear how to utilize such a matrix. (2) Information about every cell type is not available, i.e., $k_o \leq k$. (3) Information is not available about every gene, i.e., $n_o \leq n$. We deal with these issues sequentially.

Suppose we have a binary prior information matrix B such that $n_0 = n$ and $k_0 = k$. We wish to convert this into a real valued matrix M_0 of the same size, where the expression is in the same scale as the scRNA-seq data X. To do this, we use the data matrix X to calculate the high and low levels for each gene by clustering the observed expression values for that gene into two clusters and mapping the high and low values of cluster centers to 1 and 0 respectively. We note that in the case that $n_0 \leq n$ and $k_0 \leq k$, still the same method can be utilized to obtain the real-valued matrix M_0 of size $n_0 \times k_0$.

Next, we take up the issue that information is not available about all genes, i.e., $n_0 < n$. The basic idea that we exploit is the following: the knowledge even of some genes is sufficient

to cluster all observed cells into k_0 types, using a distribution specific clustering (for example, the Poisson k-means algorithm described earlier). The k_0 cluster centers in dimension n can then be used as the means of these clusters, thus effectively giving us an updated M_0 matrix of size $n \times k_0$.

Finally, we proceed to the issue that only some cell-types are specified, i.e., $k_0 \leq k$. In case that $k_0 = k$, we have an initialization for all the matrix M_0, which can be used an initialization for the alternating maximization algorithm. In case that $k_0 = k$, we do one round of distribution specific k-means++ with the first k_0 components initialized by the known data and the rest obtained as maximally distant points from the known points. This returns us the means for all the k components, which we update as M_0 of dimension $n \times k$. This matrix of predicted means M_0 is now used to initialize the various downstream algorithms, in particular serving as an initialization for maximizing W in the alternating optimization.

3.3 Results

3.3.1 Distribution selector correctly predicts the best sampling distribution for a dataset

To verify the accuracy of the distribution selection methodology, we first generated three synthetic datasets using different distributions (Poisson, Gaussian and log-normal). Each gene in the synthetic dataset has a mean that was randomly chosen between 0 and 1, with a constant variance for all genes for the Gaussian and log-normally distributed datasets.

As seen in Fig 2B, distribution selector correctly predicts the dominant distribution for each of the synthetic datasets. On the real datasets, we do not know the underlying distribution, however, we can check which distribution performs best on the downstream task of clustering. We check whether the predicted distribution leads to the highest accuracy.The predicted clusters are identified by assigning each cell to the highest weight class in the cell type fraction matrix, W. The cluster purity is then measured using Normalized Mutual Information (NMI) between the predicted clustering and the true cell types in the data and

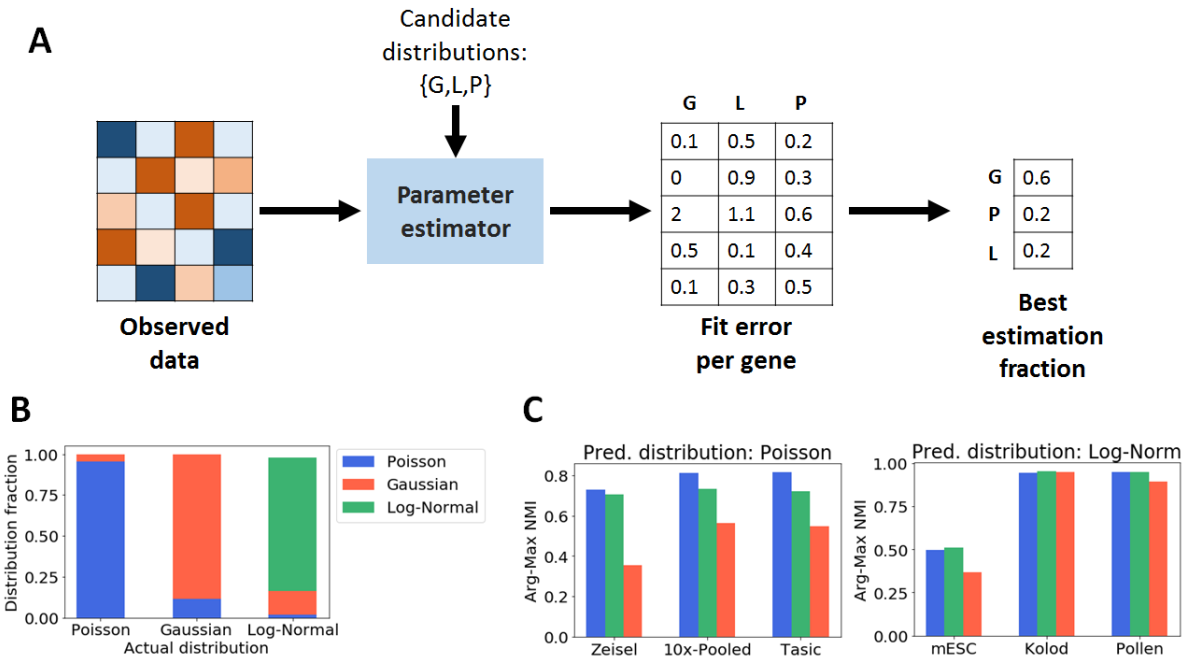


Figure 3.2: Selecting the best sampling distribution for a dataset from a set of distributions using Distribution Selector. A) Overview of Distribution Selector. B) 'Best Estimation Fraction' correctly identifies distributions of synthetic datasets. C) Comparison on different single cell datasets show that using predicted distribution leads to the highest cluster purity (measured using arg-max NMI).

is seen to be highest for the distributions that are predicted to be dominant distribution according to distribution selector as seen in Fig 2C. In general, the Poisson distribution is seen to be a better fit for count or UMI data, while the log-normal distribution is a better fit for normalized (FPKM, RPKM, TPM) data.

Having identified the sampling distribution for a dataset, the state estimation procedure factorizes the data into two matrices, M and W using the distribution, as described in 3.2.1. The product of these factorized matrices then can be treated as an estimate of the true state matrix and used for all subsequent downstream learning tasks.

3.3.2 Preprocessing leads to improvements in clustering and visualization

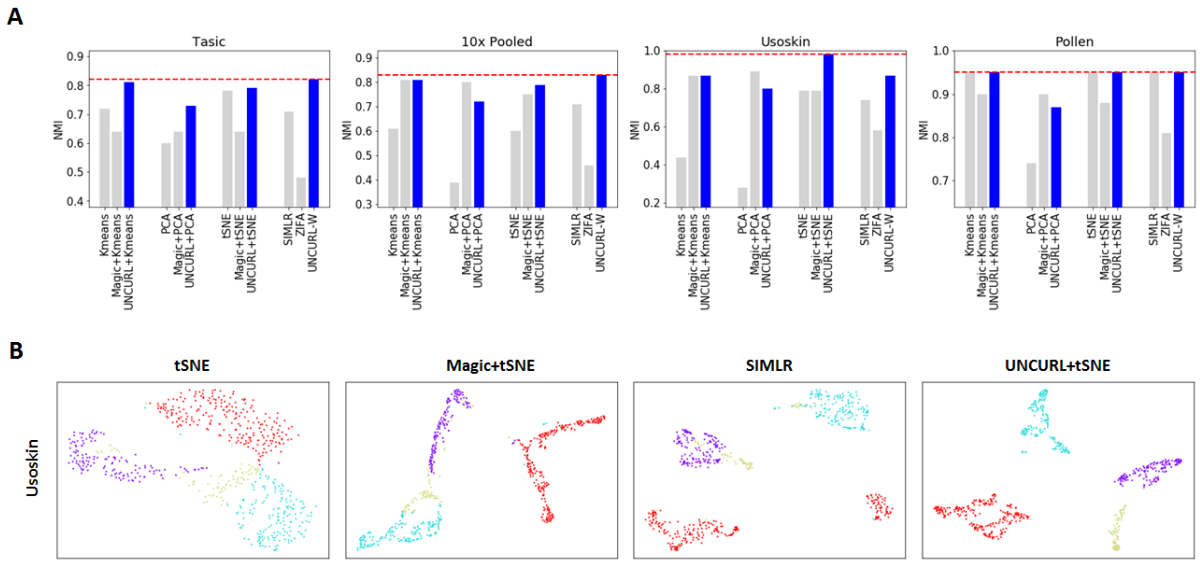


Figure 3.3: Preprocessing with UNCURL leads to improved visualization and clustering performances. A) Comparison of various clustering approaches with and without preprocessing on different scRNA-seq datasets. B-C) Different 2D visualizations of the Usoskin and Tasic datasets respectively.

Clustering and dimensionality reduction are common downstream tasks for scRNA-seq data. These are both unsupervised tasks: clustering involves dividing the cells into different cell types based on similarity of gene expression patterns, while dimensionality reduction involves creating a low-dimensional view of the data for visualization or clustering. Both tasks are useful in identifying cell sub-populations in an unlabelled dataset.

Two of the most commonly used tools for scRNA-seq data are PCA and tSNE [47]. While these algorithms have different underlying mechanisms for converting the high dimensional information to typically 2 or 3 dimensions, they assume Gaussian or t-distributions for the dataset, which is often not reflective of the actual sampling distribution of the data. To alleviate such issues, specialized tools have been developed for scRNA-seq data such as SIMLR [50]

and ZIFA [51], which explicitly account for the distribution of scRNA-seq datasets. While ZIFA explicitly accounts for zero inflation, it does not account for the full sampling distribution. SIMLR uses multiple Gaussian kernels to fit the data without having an explicit sampling model. While these tools lead to improvements over PCA/tSNE, we demonstrate that using UNCURL as a preprocessing tool can greatly improve the performances of these common dimensionality reduction tools and often even make them better than specialized tools such as ZIFA/SIMLR.

UNCURL can be used for clustering in several ways. The simplest clustering method is to assign as the cluster $c_i = \text{argmax}_j W_{j,i}$ for cell i, where W is inferred from state estimation. We call this method UNCURL-W. In addition, clustering can benefit from UNCURL preprocessing by running tSNE or PCA and then k-means on the output of UNCURL.

We demonstrate the utility of using UNCURL as a preprocessing tool by comparing the clustering performance of various methods with and without preprocessing with UNCURL, along with clustering after dimensionality reduction with ZIFA and SIMLR. Additionally we compare with another preprocessing tool, Magic [49]. Performance was measured using the NMI between true and predicted clusters with the selected preprocessing and clustering methods, as done in [50]. As seen in Fig 3A, UNCURL improves the performance of k-means and PCA on all four datasets (a more comprehensive set of results can be seen in Supplementary methods), and it improves the performance of tSNE on most datasets. Moreover, in all four datasets the top performing approach (showed with the red dotted line) is an UNCURL preprocessed approach.

To investigate the effect of UNCURL preprocessing on visualization, we run tSNE on the [62] and the [63] datasets, which are FPKM and Count valued respectively. We compare against the unprocessed tSNE visualization, tSNE after preprocessing using Magic and visualization using SIMLR (which is closely related to tSNE). The Usoskin dataset (Fig 3B) comprises of sensory neuronal cells from four principal neuronal types. We see that while all other approaches end up grouping one or more cell types together, UNCURL leads to complete separation of all principal neuronal types.

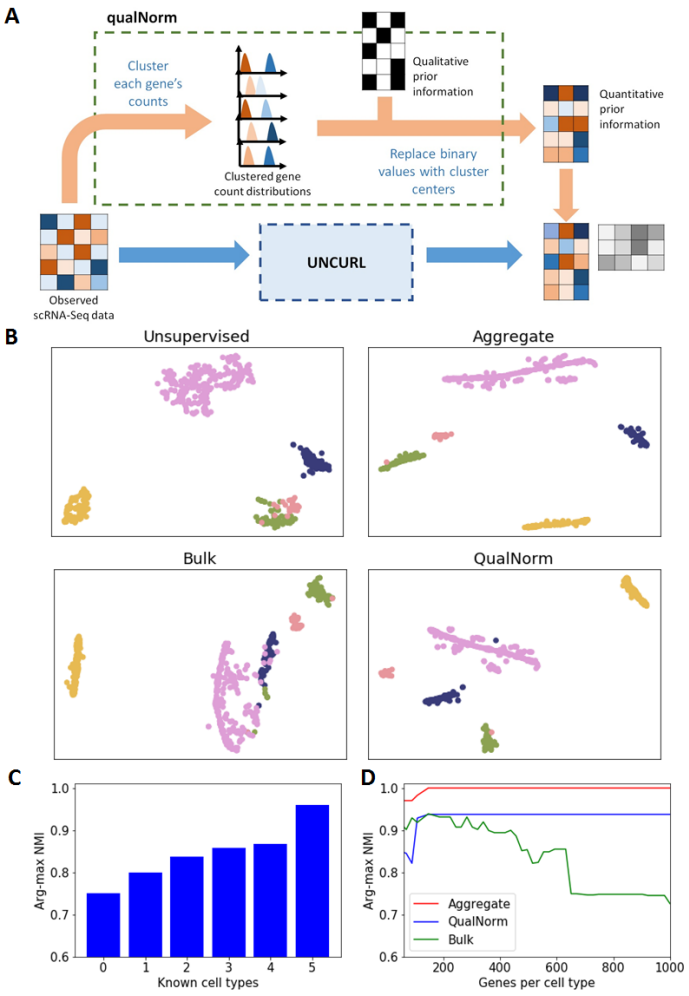


Figure 3.4: Semi-supervision with prior biological information can further improve the performance of UNCURL. A) An illustration of the qualNorm framework to convert qualitative prior information into good initialization points for unsupervised learning algorithms. B) Visualization using tSNE with unsupervised, aggregate, bulk and QualNorm semi-supervision on a subset of the Zeisel dataset (containing 1672 cells and 5 cell types). C) Comparison of improvement in purity with prior information about different number of cell types. D) Comparison of clustering purity with prior information about different number of cell type specific genes.

The Tasic (Fig 3C) dataset is comprised of cells from mouse cortex tissue, with 49 cell types, including neuronal and non-neuronal cells. Using UNCURL along with tSNE, we were able to identify 49 distinct clusters that corresponded very strongly to the cell types identified previously. For the same dataset, the performance of tSNE without preprocessing and Magic preprocessing was seen to be significantly worse, with many cell types being grouped together.

3.3.3 Prior knowledge improves UNCURL

To demonstrate the utility of prior biological knowledge, we consider a dataset with available qualitative information in the form of bulk RNA-seq data obtained from different experimental conditions and consider the effect of semi-supervision on visualization with tSNE. Specifically, we focus on the a subset of the data set of [32], comprised of five non-pyramidal cell types: oligodendrocytes, astrocytes, interneurons, microglia and endothelial cells. An upper bound on the performance with semi-supervision information is obtained when we feed the aggregate means of the true clusters (the means of all cells with each of the ground-truth labels) as the initialization. We compare this with semi-supervision using the output of QualNorm, bulk means and unsupervised preprocessing. In order to test the validity of our QualNorm framework, we compare the performance with aggregate-mean initialization to the performance obtained when we process these aggregate means through the QualNorm framework. In Fig 4B, the four initialization methods are compared, and it is seen that while semi-supervision with aggregate means and QualNorm means lead to a clear separation of cell types and an improved over unsupervised preprocessing, initialization with bulk means leads to worse visualization for this dataset. A similar experiment (see Supplementary methods) with the 10x pooled dataset also leads to a qualitative improvement in tSNE/PCA based visualizations and improvement in clustering purity.

First, we simulate the scenario where we have information available about a subset of cell types by using different number of known means (generated by binarizing the public bulk data and passing it through the QualNorm) and generating the other means using our

version of the distribution informed k-means++ algorithm. We then calculate NMI between predicted and true clusters (using arg-max of W) to quantitatively measure the performance of state estimation. As seen in Fig 4C, we observe that increasing the number of known means leads to improvement in accuracy. Moreover, we also see that prior information about even a subset of cell types is usually enough to improve the performance over the completely unsupervised case.

Then, to test the effect of having information about only a subset of the genes, we chose different number of top cell type specific genes (measured by one-vs-all differential expression) as initialization points for UNCURL and tested their effect on the purity of the clusters. Furthermore, we also tested the effectiveness of three different semi-supervision strategies namely, 1) true cluster centers (generated by taking aggregate means with the known true labels), 2) bulk means and 3) QualNorm means. It is seen in Fig 4D, that while the true cluster centers lead to almost perfect estimation of cluster membership, the QualNorm means lead to better accuracy than using the bulk data for most subset sizes, which we attribute to biases inherent to different sequencing methods. This effect becomes more pronounced as the number of genes being considered increases. An interesting observation here is that information about a few cell type specific genes is enough to very high NMI values, even when the information is qualitative.

While the results in these two cases of missing information highlight the flexibility of the QualNorm framework to handle different amounts of provided prior information, they also demonstrate how having even a little additional information is enough to improve unsupervised learning results significantly.

3.3.4 Improving lineage estimation with UNCURL

One biologically important downstream task post dimensionality reduction is lineage estimation, which is often used to study the dynamics of various genes during cell differentiation or development. Lineage inference aims at identifying smooth continuous manifolds in which cells lie in order to study the gradual change in gene expression during various developmen-

tal processes. While there exist several common tools that exist for this purpose [34, 36, 64], most tools operate directly on the sampled observed data. Additionally, most lineage estimation tools do not allow for the incorporation of any prior biological data beyond selecting the subset of genes to use for lineage inference. Here we demonstrate that the use of UNCURL preprocessing can lead to the estimate of cleaner lineages as well as allow for the incorporation of qualitative prior information into the lineage inference process.

Since it is not possible to obtain ground truth ordering of cells, we first study the effect of UNCURL preprocessing on simulated datasets. We created two separate synthetic lineages (a linear and a branched) using a method described in Supplementary Methods and sampled using a Poisson distribution. For both datasets, we preprocessed the datasets using UNCURL and Magic and tested three commonly used lineage inference tools (Monocle, Monocle2 and Slicer) on both preprocessed and unprocessed datasets (see Supplementary Methods). For the linear dataset, a good measure of lineage accuracy is the rank correlation of the true ordering of the cells with the pseudotime (an arbitrary metric that is commonly used to measure progress along a trajectory). We find that UNCURL improved the rank correlation of all three methods compared to both the unprocessed data and Magic, whereas preprocessing using Magic lead to worse performance for both Monocle2 and Slicer. For the branched data, there isnt a similarly simple way to quantify lineage estimates so we looked at the branch purity (a measure of how well cells have been ordered into the right branches). Even in this case we found that the branch purity obtained in Monocle2 after preprocessing with UNCURL was higher than both unprocessed and preprocessing using Magic.

We now look at real biological data with some amount of ground truth data available. We specifically focus on the dataset of [4] which contains cell types comprised of four stages of olfactory neurogenesis along a linear differentiation path. The cell types were identified using markers specific to different stages of development. While this information itself is of the form of qualitative prior information, using the marker genes used to label the cells would make the problem trivial for UNCURL. Hence, we instead simulate a bulk dataset by using the true labels of the dataset, which is then binarized to generate qualitative marker information

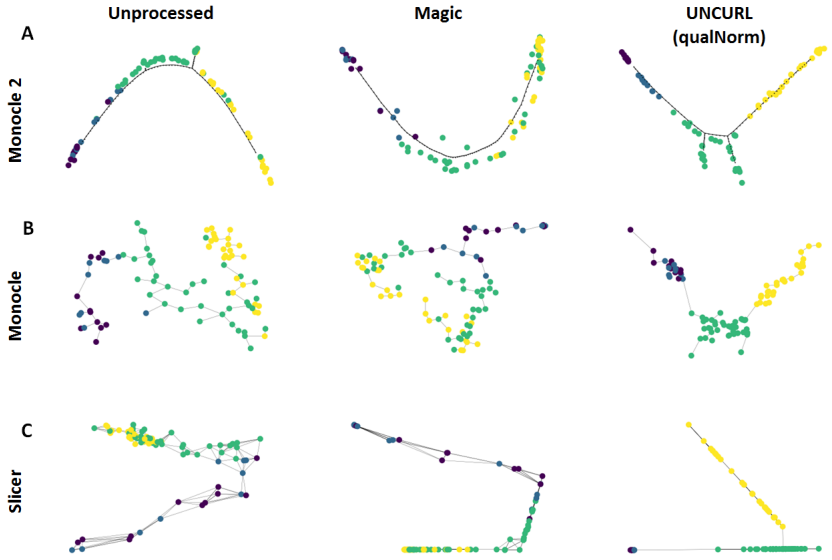


Figure 3.5: Semi-supervised preprocessing using UNCURL can dramatically improve the performance of lineage inference algorithms. A-C) Lineages inferred by Monocle2, Monocle and Slicer respectively with no-preprocessing, Magic preprocessed and semi-supervised UNCURL preprocessed for the dataset of Hanchate et. al. (containing 85 cells from 4 cell types). The lineages obtained after preprocessing using semi-supervised UNCURL lead to much clearer separation of known cell types in the predicted lineages.

for all sufficiently expressed genes (same as used in the original paper). We then use this to use both the qualNorm initialization as well as unsupervised initialization to preprocess the dataset using UNCURL. We then perform lineage inference using Monocle2, Monocle and Slicer on the unprocessed data, UNCURL preprocessed data and Magic preprocessed data. As seen in Fig 5 A-C, QualNorm initialized UNCURL leads to consistent improvements of the lineage inference algorithms when measured by the amount of overlap between the known cell types in the lineage graphs. In addition, preprocessing using UNCURL without semi-supervision leads to qualitatively similar lineages as the unprocessed datasets with slight overlap between consecutive cell types (see Supplementary Methods). On the other hand,

while preprocessing using Magic seems to lead to qualitatively similar lineages for Monocle2 and Slicer, it also leads to the estimation of a major non-existent branch in the case of Monocle (Fig 5B).

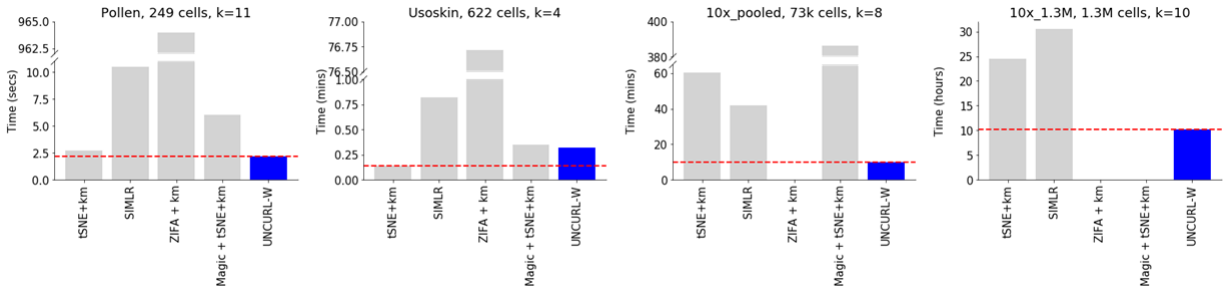


Figure 3.6: Timing comparison of different clustering approaches for various scRNA-seq datasets. UNCURL is faster than other approaches on most datasets of various sizes. Moreover, unlike methods like ZIFA and Magic, UNCURL is scalable for large datasets. A more comprehensive comparison can be found in the supplementary methods.

3.3.5 Scalability

UNCURL is capable of running on larger datasets comprising of up to millions of cells. UNCURL uses a fast public NMF package [65] for the log-normal and Gaussian distributions, while using SPNoLips (Sparse-Parallel-NoLips, described in supplementary methods), which is a custom implementation based on the NoLips algorithm ([56]) for the Poisson distribution. Both implementations are capable of using sparse matrices as input for memory and runtime advantages, and are parallelizable.

The runtime of UNCURL is $O(n_p k)$, where n_p is the number of nonzero elements in the input matrix X, and k is the number of cell types. This means that UNCURL scales linearly in the number of cells in the dataset. To deal with the dependence on k, one possibility is to use UNCURL hierarchically: first run it with a small k on the entire dataset, then partition the dataset based on the assigned clusters, and run UNCURL on the subsets.

The computational performance of UNCURL on various datasets compares favorably to that of other methods as seen in Figure 6. The runtimes of UNCURL is usually less than the other comparable methods for clustering tasks on most datasets as seen in Figure 6 (a more comprehensive comparison is in the supplementary methods). The memory usage was lower than that of comparable methods such as SIMLR and Magic. UNCURL's performance is best on sparser datasets, where more entries are zero, since the NoLips update function only uses nonzero values of the data matrix. Runtime comparisons with Magic and ZIFA are limited because Magic's memory usage is quadratic in the number of cells and ZIFA is slow compared to other algorithms, making it impractical to run on the largest datasets.

3.3.6 Exploratory analysis of the 10x 1.3 million dataset

The scalability of UNCURL allows it to run on very large data matrices, including the 1.3 million-cell dataset from [66]. This dataset is composed of unsorted brain cells from 18-day mouse embryos. We tested UNCURL on both the full dataset and a 20,000 cell subset. Since this dataset does not have any ground truth labels, we used various exploratory methods to characterize the different cell types present.

We empirically chose the number of main cell types to be 10 after experimenting with various values of k. Our selection of k was based on the distinctness of genetic signature of the identified clusters. Fig 7A show the result of UNCURL's clustering and visualization on the 20,000 cell subset while 7B shows the unprocessed tSNE visualization followed by k-means clustering. To test the concurrency of the two approaches, we generated a confusion matrix (Fig 7D) and noticed decent overlap between the clusters resulting from the two distinct methods (with an NMI of 0.45).

To evaluate the clusters generated by UNCURL, we identified the genes most highly expressed in each cluster compared to other clusters (see supplementary method for description). We then created a subset of the expression matrix, where the rows are grouped by cluster-specfic genes and columns are grouped by cluster-specific cells. Visualizing the expression heatmaps before and after UNCURL (Fig 7C) shows that the top cluster-specific

genes are distinctly expressed only in their individual clusters. Moreover, this expression pattern is amplified in the UNCURL processed data compared to the unprocessed data. This pattern is not seen when the data is preprocessed with Magic (see supplementary methods). To further validate our findings, we overlay the average expression of the top cluster-specific genes for each cluster on the tSNE visualization (Fig 7E).

3.4 Conclusion

In this manuscript, we introduced a preprocessing framework for scRNA-seq data. Our framework, UNCURL, uses the estimated sampling distribution of scRNA-seq data together with a convex mixture model assumption to estimate a true state matrix from observed scRNA-seq data. UNCURL further includes a computational framework, qualNorm, which can be used to incorporate prior biological knowledge into an improved estimate of the true state matrix.

By comparing against several benchmarking datasets, we demonstrated that preprocessing using UNCURL leads to superior separation of cell types in reduced dimensions as well as higher cluster purity for clustering tasks compared to prior tools. We further showed that semi-supervision using different types of prior information can lead to further improvement in accuracy of the learning tasks. Furthermore, we demonstrate that semi-supervised preprocessing using UNCURL allows the incorporation of prior information in even lineage estimation tasks. UNCURL scales to large datasets and typically runs faster than prior methods, particularly on large and sparse datasets. The run time for UNCURL scales linearly with the number of cell types in the dataset, but it may be possible to further reduce run-time using a hierarchical strategy.

UNCURL is an efficient preprocessing framework for several unsupervised and semi-supervised learning tasks, but it still has some limitations. While our method accounts for the sampling effect on the data, we do not take into account other sources of variability such as cell cycle effects and biological noise [67]. Moreover, presently the semi-supervision framework can only process prior information that can be converted to a binary format.

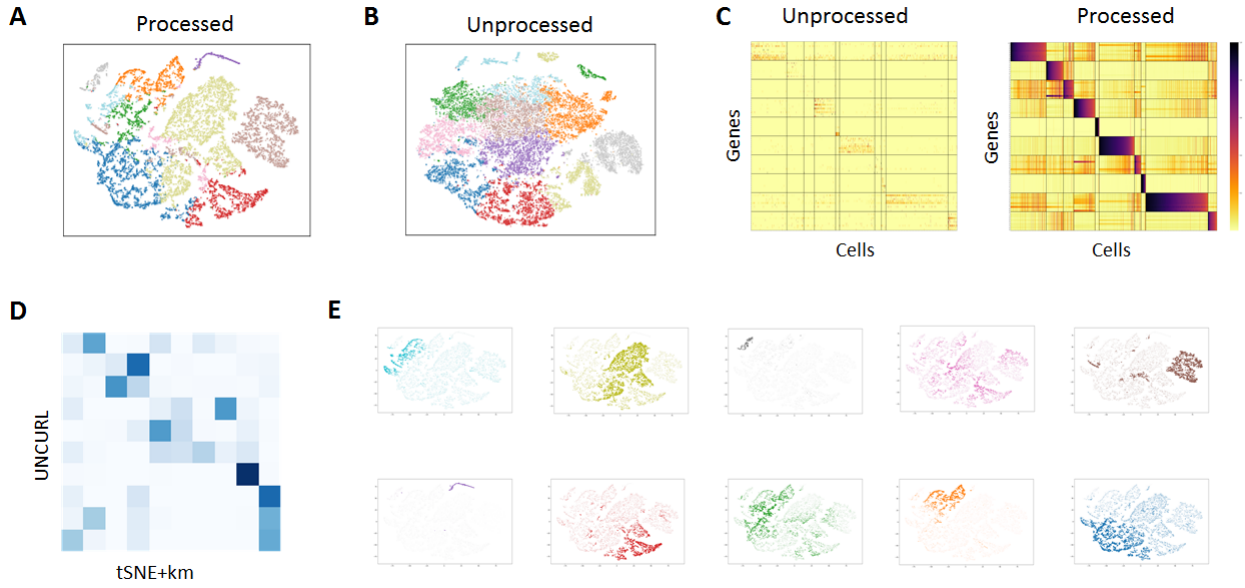


Figure 3.7: Exploratory analysis of the 10x 1.3 million cell dataset with UNCURL. A) tSNE plot on UNCURL preprocessed data with argmax inferred labels. B) tSNE plot without preprocessing with k-means inferred labels. C) Clustered heatmaps showing the top cluster specific genes identified by UNCURL before and after preprocessing. Cells sorted by decreasing W for each cluster. The heatmaps demonstrate that UNCURL identifies distinct sub-populations of cells and preprocessing makes the expression of the top clusters more distinct. D) Confusion matrix between UNCURL and tSNE + k-means labels. E) Average expression of the top cluster specific genes overlaid on the UNCURL processed tSNE plot. The expression for each cluster is colored to correspond the coloring used in A. It can be seen that the average expression of the top genes are very cluster specific, indicating that they identified distinct sub-populations.

While this still leads to improvement in accuracy, not all genes have binary states. Future work will be aimed at developing a learning framework that account for these other sources of variability and a more inclusive semi-supervision framework. We also plan to expand the

number of sampling distributions available in the current software package to include more potential sampling distributions.

Chapter 4

PIPER: IDENTIFYING PROGRESSIVE NETWORK PERTURBATION IN DIFFERENTIATION FROM SINGLE CELL RNA-SEQ DATA

4.1 Introduction

Recent advances in the field of single cell RNA sequencing (scRNA-seq) have enabled us to ask novel biological questions, which creates needs to develop new statistical methods to address them [6, 8, 9]. Unlike bulk RNA-seq or microarray measurements, scRNA-seq captures cell-to-cell variability in gene expression programs. This inter-cellular variation holds the key to inferring how genes transcriptionally regulate each other (i.e., gene regulatory network) and how their expressions and interactions change across cell states. Several studies have recently taken advantage of this data to examine biological processes such as differentiation [1, 68, 69] in a range of cell types and organisms. In this thesis, we present the PIPER approach that aims to infer *progressive network changes* across different cellular states (e.g., differentiation) and the *regulator genes* whose connection with other genes are significantly different between the network estimates. The regulator genes are interpreted as *genes likely to have driven the network differences* (Figure 4.1).

PIPER has three unique advantages over existing approaches: First, although the aforementioned studies have been able to identify gene expression changes during differentiation, existing methods are not designed to identify *key regulators* of these processes and provide mechanistic explanations. Second, PIPER extends existing differential network analysis methods that have been successful at identifying key regulators of disease by detecting changes in gene network structures between different conditions [25]. Unfortunately, existing methods are intended to do pairwise comparisons between two independent conditions

instead of capturing *structured* changes in multi-step or branched progressions. This limitation had not been an issue for bulk RNA-seq or microarray data, where high-volume and high-granularity data from multiple different conditions were rarely available. We need to extend differential network analysis methods to handle *multiple structured* conditions to leverage these data. Finally, PIPER uses the underlying distribution of scRNA-seq data that is more suited to count-valued sequencing data rather than normalized microarray data or bulk RNA-seq data.

PIPER extends previous works on network inference and differential network analysis method in the following ways. (1) *Network inference:* The earliest approaches for identifying gene networks computed correlations or mutual information between genes and preserved interactions only above a threshold value [70]. While simple, such methods often lead to the identification of spurious interactions caused by indirect interactions. This drawback is overcome in partial correlation networks [71]. A complimentary approach uses Gaussian graphical models (GGM) [72] to estimate the *conditional dependencies* between genes, assuming that the data follow a Gaussian distribution. Modifications of these methods make use of prior knowledge about the graph degree distribution [73], block structure [74] and pathway information [75] in order to further improve network inference. In practice, better performance has been achieved when data from multiple conditions are used to jointly estimate the networks at each individual condition [24,76] instead of estimating them separately. To take the maximal advantage of scRNA-seq data, recent work [77] has focused on developing network estimation methods specifically suited for such data and has demonstrated their superior performance when compared to conventional methods on count-valued data. PIPER extends these approaches by using a local Poisson graphical model with a penalty that enforce consecutive states in a progression to have similar graph structures. PIPER thus provides a scalable algorithm for learning a genome-wide network from count-valued data. (2) *Differential network analysis:* The most direct approach to solve this problem is to identify genes whose connections change extensively between pairwise conditions. While efficient, this method relies heavily on the accuracy of the network structure learning pro-

cess. DISCERN [78], a more general framework that utilizes the network parameters (edge weights) instead of network structure, was shown to be robust to errors in network structure estimates. PIPER extends DISCERN by identifying *both* perturbed genes and key regulators from *multiple structured conditions*. PIPER utilizes both perturbation analysis as well as jointly estimated network structures to identify highly connected genes which undergo strong perturbations between different conditions. Additionally, unlike DISCERN, these individual steps of PIPER are tailor-made for *scRNA-seq data* by using a Poisson distribution.

The main contribution of this chapter is the development of a general statistical method that automatically predicts key drivers of progressive gene network changes, using as input scRNA-seq data measured in multiple points over the progression.

4.2 Methods

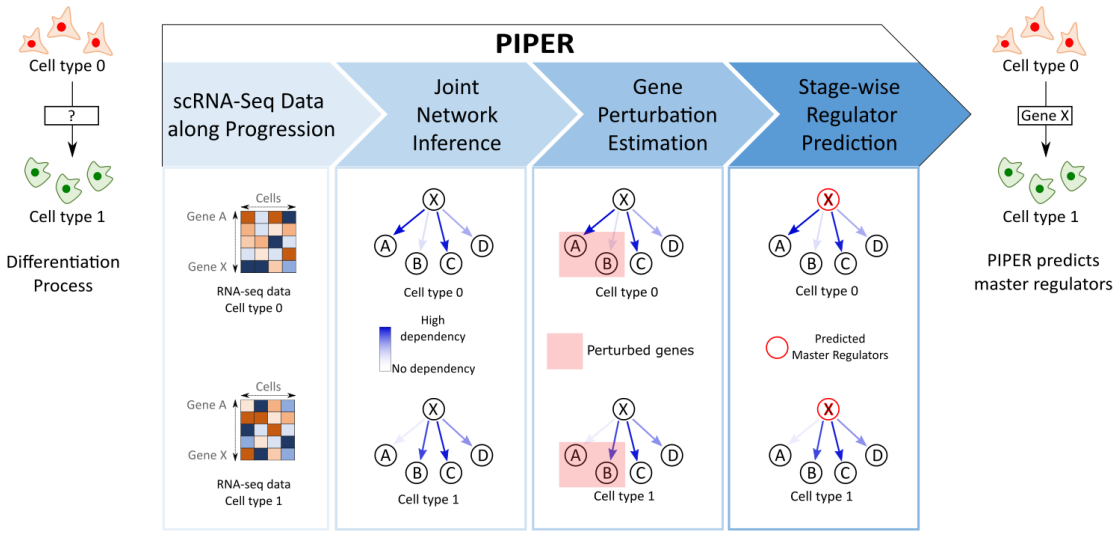


Figure 4.1: An overview of the different steps involved in master regulator prediction by PIPER. The input to PIPER is scRNA-Seq data from different states in a biological progression as seen in the cartoon on the left. The output of PIPER is the predicted set of key regulators for each stage of the progression as seen in the cartoon on the right.

4.2.1 Identifying conditional dependencies between genes

Count-valued scRNA-seq data have been successfully modeled by using the Poisson graphical models [77]. The paper proposed the use of a local model for each node, called the Local Poisson graphical model (LPGM), followed by combining information from each of these local models to infer the structure of the whole network. This is equivalent to solving the following optimization problem:

$$\Theta_{\neq j,j} = arg \min_{\Theta_{\neq j,j}} -\frac{1}{n}\sum_{i=1}^{n}[X_{ij}(X_{i,\neq j}\Theta_{\neq j,j}) - exp(X_{i,\neq j}\Theta_{\neq j,j})] + \rho||\Theta_{\neq j,j}||. \tag{4.1}$$

Here, the jth column of $\Theta \in \mathbb{R}^{g\times g}$ contains the LPGM weights to model the expression level of gene j based on all the other genes. Specifically, $\Theta_{\neq j,j} \in \mathbb{R}^{(g-1)}$ represents the jth column of Θ except the jth element. $X \in \mathbb{Z}_{+}^{n\times g}$ is a matrix of observed count values having n sample points and g genes, X_{ij} represents the value of the ith observation of X_j, the subscript notation $\neq j$ represents a sub-matrix of the original matrix which contains all rows/columns except the jth one (depending on where it appears in the subscript). Finally, $\rho \geq 0$ is the regularization parameter controlling the amount of sparsity in Θ.

To leverage the availability of data from multiple structured states, PIPER extends the estimation step of the LPGM formulation to multiple conditions as seen in equation 4.2.

$$\begin{aligned}[\Theta^1_{\neq j,j}, .., \Theta^t_{\neq j,j}] = \arg \min_{\Theta^1_{\neq j,j},..,\Theta^t_{\neq j,j}} \sum_{k=1}^{t}\Bigg[-\frac{1}{n_k}\sum_{i=1}^{n_k}[X^k_{ij}(X^k_{i,\neq j}\Theta^k_{\neq j,j}) - exp(X^k_{i,\neq j}\Theta^k_{\neq j,j})] + \\ \rho||\Theta^k_{\neq j,j}||_1\Bigg] + \frac{1}{2}\lambda\sum_{k,k'=1}^{t}E(k,k')||\Theta^k_{\neq j,j} - \Theta^{k'}_{\neq j,j}||_2^2\end{aligned} \tag{4.2}$$

This formulation simply estimates the parameters for all conditions (Θ^1 ... Θ^t) together and has an additional regularization term λ which penalizes the 2-norm distance between networks estimated at neighboring states (k and k'). $E(k,k')$ is an indicator function whose value is 1 if the states k and k' are adjacent, otherwise it is 0. We assume that the adjacency of the observed states is known to the user *apriori*. In the biological context, this modification allows us to model several types of progressions such as cell type lineage trees, differentiation

events or spatial dependencies. This technique has been successfully used in the past for GGM's [76] and is shown to result in fewer spurious network linkages.

4.2.2 Identifying highly perturbed genes between networks from different states

Identification of genes whose dependencies change between two conditions has been done in the past by DISCERN [78], an algorithm that ranks nodes whose regulators have been perturbed the most. However, in its original formulation [78], DISCERN was not explicitly designed to work with scRNA-seq and hence made an assumption of a Gaussian distribution for the data. Here we have adapted this method for count-valued processes by assuming that the data has a Poisson distribution.

Scoring perturbed genes requires first identifying the gene regulatory network for two separate conditions followed by calculating the perturbation score. The perturbation score captures how well conditional dependencies of genes in one condition can explain the data for the other condition. The perturbation score is defined as follows:

$$PScore(w_j, w'_j) = \frac{f(w_j, X') + f(w_{j'}, X)}{f(w_j, X) + f(w_{j'}, X')} \tag{4.3}$$

Where w_j and $w_{j'}$ are the vectors of regulator weights for gene j in conditions 1 and 2. While X and X' are the data sets for conditions 1 and 2 respectively. $f(w_j, X)$ is the unpenalized local Poisson graphical model cost function as seen in equation 4.1 (with ρ set to zero).

4.2.3 Identifying clusters of similarly perturbed genes

While there exist methods to identify perturbed nodes between two conditions, they cannot be directly extended to multiple conditions particularly in the form of a biological progression. Here we propose identifying the clusters of similarly perturbed genes in order to study genes potentially regulating them.

In order to identify clusters of similarly perturbed genes, we first normalize the PScore of each gene across all conditions to have a 2-norm of 1. The normalized score for each gene is then tiled at all conditions to estimate the relative perturbation between conditions. The

normalization for each gene also ensures that all scores have roughly the same scale, which is important for clustering and, moreover, it ensures that condition-specific scores for genes that are predicted to be highly perturbed at all conditions are low. Such genes are unlikely to be master regulators since their perturbations are expected to be condition specific. Next, we may perform an optional step of setting a threshold value for the normalized PScore to separate out genes that are not perturbed at any condition. This step is not seen to be important for small networks but is crucial for larger networks. Finally, we apply k-means clustering to obtain clusters of genes that have similar perturbation patterns across different conditions. The appropriate number of clusters can then selected by using well known methods such as the 'elbow method' [79].

4.2.4 Sparsifying network to improve specificity

Our network estimation method leads to the estimation of several networks at each sparsity level (corresponding to each value of ρ). For the purposes of this work, we have estimated several networks in the $\rho = [0.01, 10]$ range. The upper and lower cut-offs for ρ were empirically selected by observing the structure of estimated networks. In order to only retain edges we are confident about, we identify the most conserved network structure between the networks estimated at different values of ρ. This is done by first constructing an unweighted adjacency matrix for each value of ρ ($A_\rho(i,j) = I(|\Theta_\rho(i,j)| > 0)$, where $I(.)$ is the indicator function) and then taking the elementwise product of the the adjacency matrices estimated for all values ρ as follows:

$$A(i,j) = \prod_{\rho} A_\rho(i,j) \tag{4.4}$$

This network is then further sparsified by eliminating edges of which the values of correlation between the two genes is lower than a threshold value. For the purposes of this work, we empirically set this threshold to be the mean of the correlations between all genes in the network. While sparsifying the network might lead to the elimination of some actual conditional dependencies, we argue that specificity is more important to our problem of

master regulator identification than sensitivity. It must also be noted that unlike in [77], we do not force A to be symmetric, thereby allowing it to represent a directed graph.

4.2.5 Identifying key regulators with temporally or pseudotemporally hierarchical data

When the different conditions of the data set are in the form of a temporal/pseudotemporal hierarchy, it roughly corresponds to a process where we have access to intermediate stages of differentiation or development. Such a process can be represented by a directed graph. Each state i will also have an associated sparsified incidence matrix A^i. In such a case, for each state node in the hierarchy, i, we first find the highest valued PScore cluster(s) relevant to this state and collect the indices of the genes in these cluster(s) into a set S. We then find the number of times each gene in our data set is predicted to be a regulator of the genes (in the set S) in the state i as:

$$C_i(k) = \sum_{j \in S} A^i(k, j). \tag{4.5}$$

Here, $C_i(k)$ is the number of highly perturbed genes targeted by gene k in state i.

4.2.6 Identifying key regulators from snapshot data with missing intermediate states

For some of the scRNA-seq data sets, the data are available from a single time point containing only different mature cell types. In such a case, to identify the key regulators in the progenitor cell states that lead to each of the different mature types, one has to infer them from the observed cell types. A natural way to do this could be to look at the observed daughter cell types of a unobserved progenitor cell type and identify the highly perturbed hub genes between the different conditions. Here we propose a strategy to do this when the structure of the differentiation tree is known. First, we construct a graph joining the closest observed cell state where distance is measured by number of nodes along the tree separating a pair of nodes (we assume the true structure is known apriori). The perturbation analysis is then performed for each pair of neighbors (i, j) for this new graph. We then find the cluster for which the mean PScore at the pairwise condition (i, j) is highest and consolidate the

genes in the cluster into a set S. Lastly we find the number of times each gene in our data set are predicted to be regulators of the genes in pairwise condition (i, j) as:

$$C_{ij}^{i}(l) = \sum_{k \in S} A^{i}(l,k) \times \bar{A}^{j}(l,k) \tag{4.6}$$

$$C_{ij}^{j}(l) = \sum_{k \in S} A^{j}(l,k) \times \bar{A}^{i}(l,k) \tag{4.7}$$

Where $C_{ij}^{i}(l)$ is the number of highly perturbed genes targeted by gene l in pairwise-condition (i, j) that are specific to state i. A^{i} is the incidence matrix at state i and $\bar{A}^{i}$ is the complement of the incidence matrix.

4.3 Results

4.3.1 Simulation study I: Testing the efficiency of perturbed node estimation

To test the accuracy of the perturbed node estimation process, we first generated synthetic Poisson distributed data from random graphs with 30 nodes and 100 samples using the methodology described in [77]. To simulate different biological conditions, we randomly deleted 5% of the edges from the original graph. Graphs of different sparsity levels were generated by varying the connection probability between pairs of nodes. 20 random graphs were generated at each sparsity level. We also used these data to demonstrate the usefulness of joint network estimation by testing against a variant of PIPER where the network similarity penalty λ was set to 0. The ranking of perturbed genes for PIPER is obtained by sorting genes according to their PScore calculated between the two conditions. The accuracy at each sparsity level is measured by calculating the mean of the fraction of the N perturbed genes that are ranked in the top N genes according to the different methods over all trials (and values of ρ for PIPER and Treegl). We then compare PIPER against various other specialized methods for identifying perturbation in networks namely DISCERN, Treegl (with perturbation identified with DISCERN), D1 Score, LNS Score and ANOVA (the latter three are computed as described in [78]). For both PIPER and Treegl, the tuning parameter λ (that enforces similarity between successive conditions) is chosen to be the best performing

one at each sparsity level. It can be seen in Figure 4.2a that for graphs of most sparsity levels, both formulations of PIPER outperform all competing methods while the joint network inference seems to have a higher accuracy than $\lambda = 0$ particularly for sparser graphs.

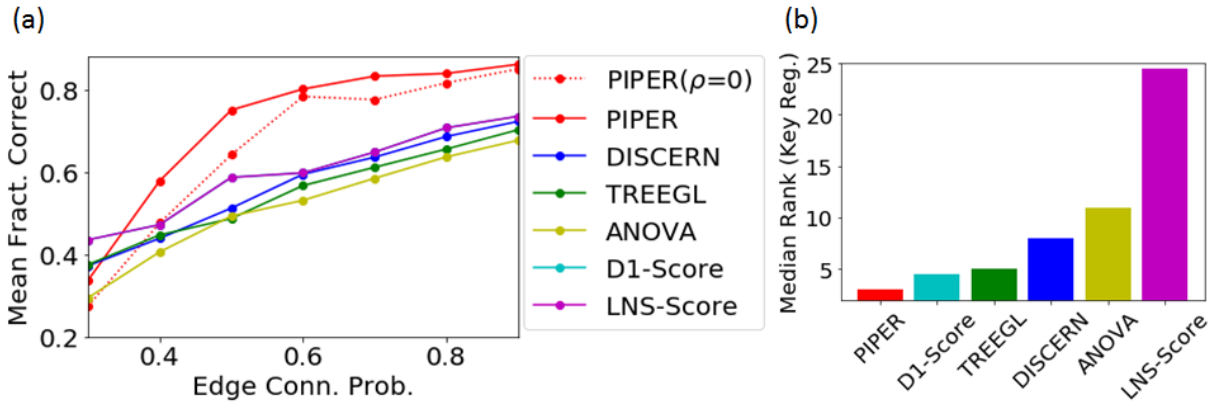


Figure 4.2: a) Testing accuracy of PIPER at identifying perturbed genes for graphs having different sparsity levels. The mean fraction of perturbed genes that are correctly predicted is calculated by averaging across all values of the sparsity parameter ρ. b) Comparison of predicted mean rank of actual key regulator for various methods on scale free graphs of degree 3. PIPER outperforms other methods in predicting the key regulator genes.

4.3.2 Simulation study II: Testing key regulator identification on synthetic data from scale-free networks

Having demonstrated PIPER's accuracy at identifying perturbed genes between networks from two different conditions, we test its accuracy at identifying the key regulators of a simulated biological progression. This is a three state system where a single progenitor state differentiates into two daughter states. The procedure for generating the graph, perturbed graph and the data sets is similar to the previous sub-section with the notable exception being that here we assume the graph is scale-free [80] (average degree = 3) i.e. its node degree distribution follows a power law. Scale–free networks are common in biology and

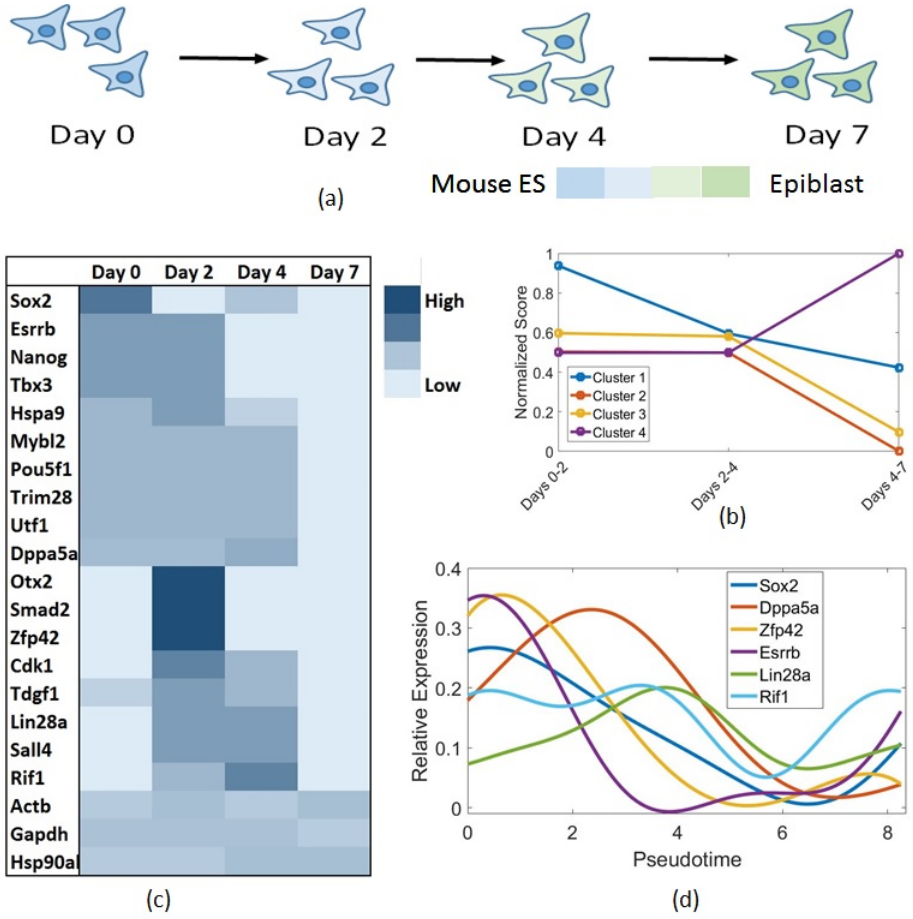


Figure 4.3: time points from which the data is available. b) Clusters of P-scores across multiple days. c) Heatmap of number of different perturbed genes targeted by the predicted key regulators of differentiation across different days. d) Expressions of different predicted key regulators plotted against pseudotime inferred from Monocle. Peaks in expression patterns are closely match the stages on which the regulators are predicted to act.

have many nodes with small node degrees with a few nodes (called hubs) having much higher node degrees. It has also been demonstrated that most key regulators are hubs in their respective networks [81]. Hence, when deleting the edges, we first perform a weighted sampling (where the weights are proportional to the degree of the node) and then select one

of the outgoing edges of the selected node, uniformly at random to delete. This method ensures that hub nodes are more likely to be the main regulators for the perturbed genes. Due to the randomness of the edge deletion, the two daughter networks are also different from one another. We now use this method to generate 20 random set of networks (and data sets) of the same dimension as the previous sub-section with 10% of the edges deleted in the perturbed network. We then proceeded to first obtain the PScores for all genes, and the regulators of the genes with the top N PScores, where N is the number of actually perturbed genes (as in the previous sub-section). We then ranked the predicted regulators of these genes according to their frequency of occurrence. For each trial, we then used this ranked list to find the predicted position of the actual top regulator for each branch of the differentiation process and calculated the average median rank across both branches. We found that PIPER outperformed other algorithms in identifying the key regulators when compared by the median predicted rank of the actual top regulator (as seen in Figure 4.2b).

4.3.3 Case I: PIPER correctly identifies early and late regulators of mouse embyonic stem-cell (mESC) differentiation

For a first application we used scRNA-seq data [8] collected at four time points (day 0, 2, 4, 7) during differentiation of mouse embryonic stem cells to epiblast cells. We assume data from each day represents a different dominant cell state during the differentiation process. We focused on a panel of 89 genes comprising of essential housekeeping genes, key regulators of mouse ES differentiation and differentiation markers [82]. As per the PIPER workflow, we first performed a joint network inference, followed by gene perturbation estimation across successive conditions and clustering of genes on the basic of PScores. This process lead to the identification four distinct clusters of genes (Figure 4.3b).

Next we identified the clusters that are relevant to each day (by looking at the cluster that contained the highest PScore for a particular day): cluster 1 for day 0, clusters 1 and 3 for day 2, clusters 3 and 4 for day 4, and cluster 4 for day 7. For each day we now only look at the genes from the clusters relevant to the day and identify their regulators (shown

in Figure 4.3c). The column normalized heat map shows how many perturbed genes are targeted by each regulator across different days.

We note that most predicted regulators from early stages (days 0, 2 and 4) are known differentiation regulators. For example, Pou5f1, Sall4, Zfp42, Utf1, Sox2, Nanog, Esrrb, Dppa5a, Rif1 are all known to have roles in maintaining pluripotency [83–87]. Of these genes, Nanog and Sox2 have been widely reported to be among the earliest regulators of mouse ES differentiation [87, 88] and have been correctly implicated by PIPER to control early stages of differentiation (days 0 and 2, respectively). Moreover, Rif1, which is predicted to regulate subsequent stages of differentiation (days 2 and 4) is known to be activated by Nanog [87], consistent with PIPER's prediction. Since most of the differentiation process is completed by day 4, we note that the predicted regulators from day 7 are all housekeeping genes.

To further validate the accuracy of the order of regulator action, we used Monocle [1] to order cells by differentiation progress. This cell state information can then be used to plot gene expression as a function of pseudotime (an arbitrary metric representing progression along cell states). While Monocle is not designed to identify the key regulators of differentiation, one would expect the expression patterns of our predicted regulators for different stages of differentiation to show big changes during the predicted period of their effect. This can be seen in Figure 4.3d, hence providing a further validation to PIPER's predictions.

4.3.4 Case II: PIPER identifies genes responsible for lineage formation in neuronal cell types

To test the accuracy of PIPER on branched snapshot data, we looked at a data set [6] containing neurons and non-neuronal cell types from the mouse cortex and hippocampus. We focused on a sub-set of the dataset containing 3 mature cell types deriving from a common precursor (namely astrocytes, oligodendrocytes and interneurons, Figure 4.4a). We aim to identify the key regulators that drive differentiation of the common neuronal stem cell precursor into each of the three terminal cell types.

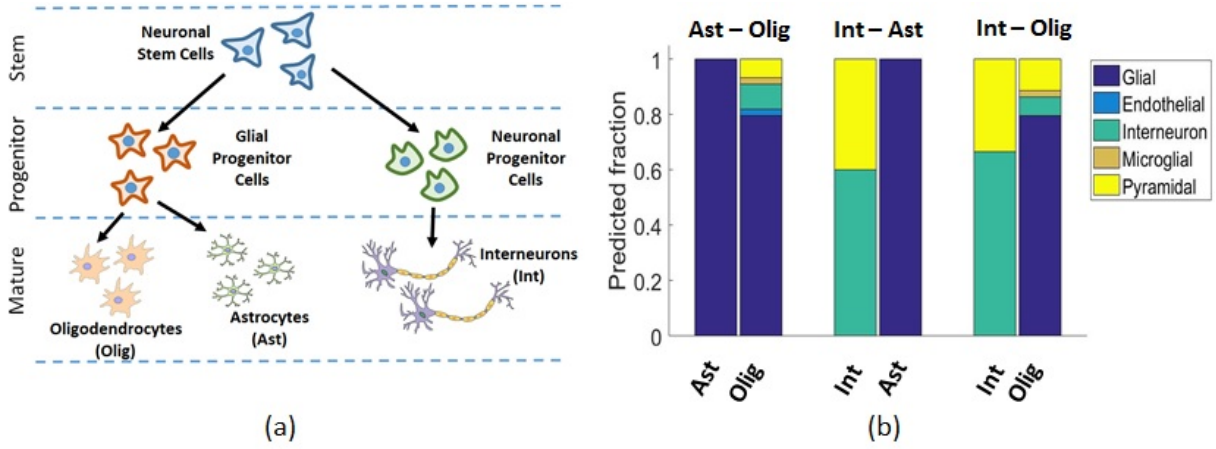


Figure 4.4: Application of PIPER to scRNA-seq data obtained from mouse brain samples correctly identifies genes enriched for different neuronal cell types. a) Neuronal stem cell lineage tree. b) Fraction of predicted regulators of each stage belonging to different progenitor cell types. It can be seen that the regulators of most daugther cell types are specific to the correct progenitor type.

We first identified a subset of genes that are of interest for differentiation. We started with a list of over 3,000 genes that were reported in [6] to be uniquely upregulated in any one of the main cell types. We then limited the list to 528 by excluding genes that do not have at least one count on average in one of the cell types. This filtering step removed genes that have too few counts to lead to a meaningful identification of network structure. We then performed network estimation with all three cell types and estimated the perturbation of each cell type compared to their closest neighboring cell type along the differentiation tree. Upon normalization of Pscores and clustering, we were able to obtain three major clusters of genes which are upregulated in at least one of the cell type pairs. After thresholding, we were left with only 68 out of 528 original genes. Next, we identified the regulators of these genes by sparsifying the networks at each condition as previously described in the Methods section. We thus obtained the list of key regulators by retaining genes which have regulate

more than one perturbed gene. To check the validity of our results we compared against the list of genes unique to each cell type obtained from [6]. The genes specific to each progenitor type is obtained by merging the list of highly expressed genes for each of their daughter types. We note that while this is an approximate way to obtain genes specific to progenitor types, a more comprehensive list would require data from the progenitor states themselves. Figure 4.4b shows the fraction of predicted regulators unique to each of the two conditions that overlap with genes associated with different progenitor types according to [6]. It is interesting to note that not only are most predicted regulators also associated with correct progenitor cell types, when they do overlap with other cell types it is usually with a closely related cell type. Moreover, several of PIPER's predicted key regulator genes such as Dbi, Gsn, Olig1, Lgi3 or Tcf4 are known to be important differentiation regulators of neuronal cell types. Several of the genes identified by PIPER such as Atp2a2, Mal, Hsd17b7, or Pdlim2 are known to play roles in differentiation in other cell types. These genes could be subject of future experimental study to determine their role in neuronal differentiation events.

4.4 Conclusions

In this work we presented PIPER, a tool for identifying key regulators of differentiation events from scRNA-seq data. We demonstrated that PIPER can deal with various forms of data, such as time-series data containing intermediate states as well as data from one time point representing mature differentiated cell types. We benchmarked the individual components of PIPER on synthetic data and tested the complete workflow on two different biological data sets. PIPER predicted known key regulators of each differentiation/development process and notably predicted the correct temporal ordering of regulator action in mouse ES differentiation.

Future work can be aimed at inferring the graph structure of differentiation states along with the prediction of key regulators. This change would address a current limitation of PIPER, which is the need for an accurate differentiation graph to be provided to the algorithm. While this graph might be easy to obtain for well-studied processes, a more general

approach could enable the study of new processes about which enough information is not available.

Chapter 5

EFFECT OF REPRESSION MECHANISMS ON NOISE SUPPRESSION IN MICRO RNA-BASED INCOHERENT FEED FORWARD LOOPS

5.1 Introduction

MicroRNAs are an interesting class of non-coding RNA molecules that have been linked with the regulation of several important biological processes such as differentiation [89, 90]. Changes of micro RNA expression levels have also been directly linked with cancerous changes in cells [91,92]. Despite evidence of their roles in such a large variety of important biological contexts, there still isn't a complete understanding on how microRNAs regulate their targets and the reason for their widespread presence in various biological networks.

Combined efforts of knockout studies [90, 93, 94] and computational models [3, 95, 96] suggest that miRNA operates at two levels: they buffer noise at the post-transcriptional level [97], and regulate network expression by repressing master-regulator transcription factors [93, 98]. However, experimental evidence suggests that although each miRNA potentially represses several targets, the effect on individual repression events is unusually modest [99, 100]. Moreover, noise reduction at the protein level is only seen to hold for lowly-expressed genes, while the opposite effect is observed for genes with high expression [95]. These confounding properties of miRNAs make it an interesting topic of study in order to understand their functional importance to cells.

A recent study has shown that miRNA modulation of key transcription factors (TF) expression [93] acts alongside direct repression of the TF targets for fine-tuned gene expression [101, 102]. Notably, one particular miRNA based network motif, the Incoherent FeedForward Loop (IFFL) is seen to be particularly enriched in various important gene

networks. MiRNA based IFFLs have been shown to be an effective noise-suppressor and capable of buffering protein expression [96, 101, 103], leading to the speculation that true understanding of the role of miRNAs is only possible through the study of miRNA specific network motifs.

While there have been several experimental studies focused on miRNA based IFFLs [3, 104] which have demonstrated that they are buffered against copy number variations, it remains unclear if and how miRNA based IFFLs reduce both intra-cellullar stochasticities (intrinsic noise) and cell-to-cell variation (extrinsic noise). In particular, experimental results in [3] reveal that strong miRNA-mRNA binding rate could in some cases even lead to an increase of the overall noise in protein expression. This nonlinear property of miRNA has been studied before [95, 96], but its biological interpretation remains unclear.

To understand the role of the individual biological mechanisms that lead to these interesting experimental observations, we have theoretically studied the effect of different steps in the miRNA's repression mechanism on individual components of biological noise, namely extrinsic and intrinsic noise. Furthermore, we study the role of translation inhibition on noise reduction, which has not been investigated before. Our findings lead to identification of parameter ranges where miRNA based IFFLs effectively reduce either component of noise and also demonstrate that IFFLs are more successful at reducing noise than any other miRNA based network motif. Moreover, we find the noise reduction properties of IFFL affects mRNA and protein level noise differently due to the additional post transcriptional regulation through translational inhibition which can lead to more widespread noise reduction at the protein level. This observation is particularly of interest in the case of intrinsic noise, where mRNA-miRNA interaction causes an increase in high-frequency noise but reduction in low frequency noise. While this can lead to the overall increase in mRNA noise, we demonstrate that this might lead to decrease in protein noise, owing to the slower timescale of operation of the translational machinery, resulting in the filtering of the high frequency noise.

5.2 Models and methods

Model

To develop a model for the IFFL system, we first start off by describing the process of transcription and translation. This can be summarized by the following set of chemical reactions.

$$G \overset{\alpha_m}{\rightarrow} M \overset{\alpha_p}{\rightarrow} p$$

Here, the first reaction denotes transcription while latter describes translation. The miRNA-based regulation proceeds primarily through two groups of steps namely miRNA mediated degradation and translational inhibition which affect each of these reactions separately. Many models for miRNA based degradation have been proposed [105–108]. For the purposes of this work we rely on the existing models and make minor adjustments based on observation based on available experimental data [3]. We assume that mRNA (M) and microRNA (S) form an irreversible complex C that is then degraded into the microRNA alone at rate k_c.

$$M + S \overset{\gamma_s}{\rightarrow} C \overset{k_c}{\rightarrow} S$$

We also assume that mRNA can be translated into protein even when bound to the miRNA, although at a lower rate. This factor accounts for the miRNA-induced translation inhibition that has been previously reported [95,108]. Then assuming that the mRNA-miRNA complex formation is fast and so is under quasi steady state, we get the following population model:

$$\frac{dm}{dt} = \alpha_m g_1 - \beta_m m - \gamma_s m s \tag{5.1}$$

$$\frac{ds}{dt} = \alpha_s g_2 - \beta_s s \tag{5.2}$$

$$\frac{dp}{dt} = \alpha_p m(1 + Ks) - \beta_p p \tag{5.3}$$

Here, m is the no. of mRNA transcripts, s is the number of mature miRNAs, p is the number of proteins, K captures the degree of translational inhibition. α_m, α_s, α_p are the

production rates of mRNA, miRNA and protein while β_m, β_s, β_p are their corresponding natural degradation rates. γ_s is the miRNA-mediated degradation rate. The production rates of mRNA and miRNA are dependent respectively on the positive random variables g_1 and g_2, which correspond to the average no. of active genes at the population level of the correspondent transcribed genes. Genes g_1 and g_2 can be transcripted either dependently ($Cov(g_1, g_2) \neq 0$) or independently ($Cov(g_1, g_2) = 0$). The detailed derivation is provided in the supplementary methods.

Extrinsic and Intrinsic Noise Measurements at Steady State

We split the source of cellular noise into intrinsic and extrinsic component (as in [2, 27]). Extrinsic noise arises from cell diversity in populations, like difference in sizes of cells, amount of inducer absorbed, difference in state of cells in terms of the cell cycle etc. Intrinsic noise stems from the inherent stochasticity in all cellular processes like transcription, translational, mRNA/protein degradation etc.

To comparatively study the steady-state noise rejection of noise in microRNA-based systems, we used the square of the coefficient of variation (as in [3]) defined as the ratio:

$$\eta^2 = \frac{\mathrm{Var}(Z)}{\mathrm{E}^2[Z]} \tag{5.4}$$

where Z is the process of interest. The coefficient of variation allows to compare noise in processes that have different means. This is essential in our study, as microRNA-control reduces the level of mRNA through degradation.

Extrinsic Noise Measurement at Steady State. Steady states of the mRNA and protein levels of the general IFFL system according to the model presented in subsection 5.2 are:

$$m^* = \frac{\alpha_m g_1^{ss}}{\beta_m + \gamma_s d g_2^{ss}} \tag{5.5}$$

$$p^* = \frac{\alpha_p \alpha_m g_1^{ss}(1 + \gamma_s K d g_2^{ss})}{\beta_p(\beta_m + \gamma_s d g_2^{ss})} \tag{5.6}$$

where $d = \frac{\alpha_s}{\beta_s}$. In the reasonable assumption that there is no variability in the cellular processes within a cell population, the extrinsic noise source is the variability of the transcription of the genes g_1 and g_2. We split the steady state contribution of the open loop system (X) from the miRNA regulation (Y) as follows:

$$m^* = \frac{X}{Y} = \frac{\frac{\alpha_m}{\beta_m} g_1^{ss}}{1 + \frac{\gamma_s d}{\beta_m} g_2^{ss}} \tag{5.7}$$

$$p^* = \frac{X}{Y} = \frac{\frac{\alpha_p \alpha_m g_1^{ss}}{\beta_p \beta_m}}{\frac{1+\frac{\gamma_s d}{\beta_m} g_2^{ss}}{1+K d \gamma_s g_2^{ss}}} \tag{5.8}$$

The two quantities labelled Y capture the miRNA contribution to the dynamic (Figure 5.1a).

To estimate the extrinsic noise, we need to compute the expected value and the variance of the steady state level of mRNA and protein, which are a function of the unknown random variables g_1^{ss} and g_2^{ss}. Given the nonlinearities in 5.5, it is not possible to compute analytic expressions of these two quantities. Instead, we use the following Taylor-Delta [28] approximations:

$$\mathrm{E}\left[\frac{X}{Y}\right] = \frac{\mathrm{E}[X]}{\mathrm{E}[Y]} \tag{5.9}$$

$$\mathrm{Var}\left[\frac{X}{Y}\right] = \frac{\mathrm{Var}(X)}{\mathrm{E}^2[Y]} - 2\frac{\mathrm{E}[X]}{\mathrm{E}^3[Y]}\mathrm{Cov}(X,Y) + \frac{\mathrm{E}^2[X]}{\mathrm{E}^4[Y]}\mathrm{Var}(Y) \tag{5.10}$$

In this setting, the coefficient of variation of the process $\frac{X}{Y}$ is:

$$\eta^2 = \frac{\mathrm{Var}\left(\frac{X}{Y}\right)}{\mathrm{E}^2[\frac{X}{Y}]} = \eta_X^2 - 2\frac{\mathrm{Cov}(X,Y)}{\mathrm{E}[X]\mathrm{E}[Y]} + \eta_Y^2 \tag{5.11}$$

which leads to the simple expression:

$$\eta^2 = \eta_X^2 + \eta_Y^2 - 2\rho_{XY}\eta_X\eta_Y \tag{5.12}$$

where ρ_{XY} is the correlation between the random variables X and Y, σ is their standard deviation and η_X,η_Y are their respective standard deviations. For the sake of simplicity, we will use ρ in place of ρ_{XY} in this work when this expression is not ambiguous. An interpretation of this formula is that the process Y introduces noise in the system (represented by

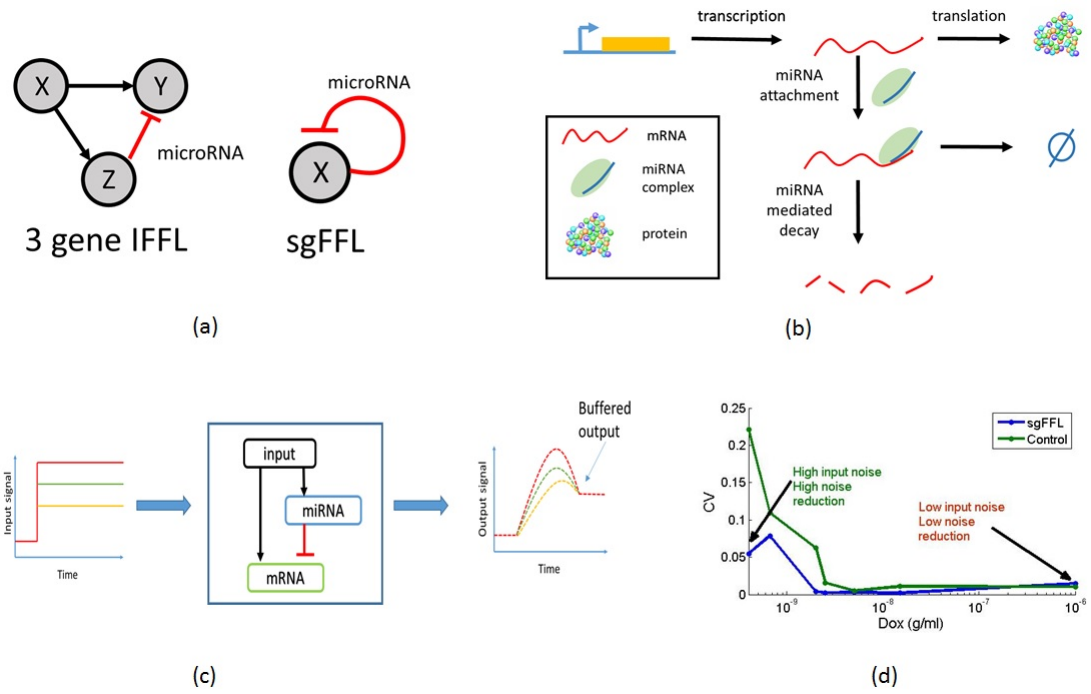


Figure 5.1: microRNA based IFFL's lead to noise reduction and buffering in gene circuits. a) Some common types of microRNA based IFFL motifs. b) Different mechanisms of microRNA based gene regulation. c) microRNA's based IFFL's display steady state buffering to different amount of input, leading to reduction in extrinsic noise. d) Experimental data from [3] showing protein noise reduction due to IFFL at different levels of input. It is seen that when the input noise is high, the noise reduction is high due to IFFL but the noise reduction is negligible when the input noise is low.

the quantity η_Y^2), but it also cancels part of the noise components of X if the two processes are positively correlated (according to the quantity $\rho\eta_Y\eta_X$).

Intrinsic Noise Measurement at Steady State. Similar to the extrinsic noise case, we studied the dynamic of intrinsic noise using the coefficient of variation as defined in 5.4. The variance of the system due to stochasticity in the biochemical reactions is usually computed

mathematically using master equations [109]. Since the dynamic described in 5.1 is nonlinear, we approximate the solution of the master equations with a Linear Noise Approximation (LNA) [109]. LNA works under the hypothesis that the variance of the steady state distribution of the population scales with the system size. In practice this means that the count of a specie X (miRNA, mRNA or protein) can be split in its deterministic and stochastic components:

$$X(t) = \Omega x(t) + \sqrt{\Omega} n_x(t) \tag{5.13}$$

Here Ω is the system size term, $x(t)$ is the macroscopic specie concentration, $n_x(t)$ is the noise term.

Under these assumptions, the LNA approach states that the steady state covariance matrix of the system is the solution of the differential equation at steady state:

$$\frac{d\sigma}{dt} = A\sigma + \sigma A^T + B \tag{5.14}$$

where σ is the matrix of covariances, A is the Jacobian matrix for the dynamics of averages and B is diffusion matrix that depends on the size of the random events. Specifically, A and B are defined as

$$A_{ij} = \frac{\partial E[J_i^+]}{\partial E[n_j]} - \frac{\partial E[J_i^-]}{\partial E[n_j]}$$
$$B_{ij} = \sum_k v_{jk} v_{ik} R_k$$

where J_i^+ and J_i^- are the total fluxes of production and elimination of species i, R_k is the rate of reaction k and v_{ik} is the number of molecules produced of species i by reaction k. Equation C.9 can be then solved at steady state to obtain the variances of the species involved. The coefficient of variance is next computed using equation 5.4.

To compute the matrix of fluxes at steady state, we linearize system 5.1 around the final equilibrium to obtain:

$$\frac{dm}{dt} = \alpha_m g_1 - \Gamma_m m - \gamma_s m^\star s - \gamma_s m^\star s^\star \tag{5.15}$$
$$\frac{dp}{dt} = A_p m + \alpha_p K(s - s^\star) m^\star - \beta_p p \tag{5.16}$$

where $\Gamma_m = \beta_m + \gamma_s s^\star$ and $A_p = \alpha_p(1 + Ks^\star)$. Moreover, to account for the stochasticity introduced by gene switching ($G_{on} \underset{\lambda^+}{\overset{\lambda^-}{\rightleftharpoons}} G_{off}$) at the single cell level, we add two extra differential equations to the model

$$\frac{dg_1}{dt} = \lambda_1^+(1 - g_1) - \lambda_1^- g_1 \tag{5.17}$$

$$\frac{dg_2}{dt} = \lambda_2^+(1 - g_2) - \lambda_2^- g_2 \tag{5.18}$$

that track the average activity of genes g_1 and g_2 at the population level (as shown in [110]).

Measuring Steady State Autocorrelation

The steady state autocorrelation of different species can be used to analyze the distribution of noise at different frequencies. This will be used to analyze how the microRNA-mRNA interaction affects the noise characteristics of the IFFL system. From 5.13, we can estimate the dynamic values of the different noise terms as in [111] in terms of the starting noise values as:

$$N_x(t) = N_x(0)e^{-At} \tag{5.19}$$

Where $N_x(t) = \begin{bmatrix} E[n_{g_1}(t)] & E[n_{g_1}(t)] & E[n_m(t)] & E[n_s(t)] & E[n_p(t)] \end{bmatrix}$. For the purposes of the upcoming portions, let us define the steady state covariance function of two species X and Y as $C_{XY}(t) = {\sigma_{XY}}^2(t) = E[(X(0) - E[X(0)])(Y(t) - E[Y(t)])]$ and $R_{XY} = E[X(0)Y(t)]$. The terms $R_{n_m n_g}(t)$ and $R_{n_m n_s}(t)$ are then calculated at steady state. These can now be used to calculate the autocovariance of n_m at steady state i.e. $C_{n_m}(t)$. Now, realizing that at steady state $\Omega x(t) = E[X(t)]$ for any specie, implies that ${\sigma_{XY}}^2(t) = \Omega R_{n_x n_y}(t)$. Therefore the covariance of M can be easily calculated by rescaling by Ω.

Parameter Estimation

Using the model previously introduced (equations 5.1), we performed parameter estimation with time course from available datasets. In particular, we considered the single gene miRNA-based FFL (sgFFL) systems studied in [3]. The datasets in the study contain mRNA reads for

each cell measured using FISH (Fluorescence In Situ Hybridization), qPCR data for miRNA and Flow Cytometry data for protein (mCherry) expression levels. The parameter values were allowed to vary between realistic parameter ranges and MATLAB was used to perform the parameter inference. The parameter estimation process is explained in Supplementary methods.

Simulations

Stochastic simulations were run using the parameters estimated as described in the previous subsection using the Gillespie simulation technique [112]. Two sets of simulations were run corresponding to i) intrinsic noise only ii) intrinsic as well as extrinsic noise. For the former, standard Gillespie simulations were run (1000 times) for each induction level. Each cell was assumed to have the same amount of inducer and all simulations were run for 250 hours, which is enough time for the trajectories to reach steady state. For the extrinsic and intrinsic noise simulations, the amount of inducer molecules each cells get was assumed to be different. The number of inducer molecules each cell got was randomly chosen from a Gaussian distributed with the mean corresponding to different inducer levels. The variance was assumed to be half the mean. Ensemble mean and variances were calculated from the resulting trajectories.

5.3 Results

The IFFL is the only miRNA-regulated process with extrinsic noise-cancellation properties

To understand the extrinsic noise reduction properties of miRNA regulation, we studied the values of η_Y that satisfy condition : $\eta^2 \leq \eta_X^2$. In our framework, we could see η_Y as the extrinsic noise contribution of the miRNA regulation, while η_X is the constant extrinsic noise of the open loop system. Since both quantities are positive by definition, the noise reduction condition could be satisfied only for positive values of the correlation ρ. This suggests that miRNA regulation could cancel out the noise that arises from the same source

in correlated signals, according to a factor $2\rho\eta_X\eta_Y$. However, the formula also incorporates a noise term, η_Y^2, that represents the extra variability introduced by the miRNA regulation. Since expression 5.12 describes a parabola, one can conclude that there is a limited range of values of η_Y that leads to extrinsic noise rejection, before the extra noise introduced by the miRNA machinery leads to worst performance than in the open loop (Figure 5.2). Simulations show that the formula we propose (5.12) is a good approximation of the behavior for low values of η_Y ('parabolic' behavior), while the behavior is closer to a nonlinear 'switch' after the minimum value of the parabola is reached (Figure 5.2). The minimum of the parabola corresponds to the maximum noise rejection that could be achieved. This is reached for $\eta_Y = \rho\eta_X$, for which the noise rejection ratio is:

$$\frac{\text{IFFL extrinsic noise}}{\text{Open loop extrinsic noise}} = 1 - \rho^2 \tag{5.20}$$

Thus, optimal noise rejection is achieved for perfect correlation ($\rho = \pm 1$) between the two genes ($\rho(X, Y) = \pm 1 \rightarrow \rho(g_1, g_2) = \pm 1$). Since η exits quickly the 'noise-rejection' area after the optimal value (Figure 5.2), a safe condition for noise rejection is:

$$\eta_Y \leq \rho\eta_X \tag{5.21}$$

At the mRNA level (eq 5.5), condition 5.12 is guaranteed iff $\rho(g_1, g_2) > 0$, which occurs when the same upstream gene positively regulates both g_1 and g_2. This corresponds to the Incoherent FeedForward Loop (IFFL). Hence, uncorrelated miRNA or Coherent FeedForward Loops (CFFLs) does not have extrinsic noise cancellation properties according to our formula. At the protein level (eq 5.5), two conditions need to be satisfied to have $\rho_{XY} > 0$:

$$\rho(g_1, g_2) > 0 \tag{5.22}$$

$$1 - K\beta_m > 0 \tag{5.23}$$

Condition 5.22 is the same as for mRNA, which implies that only IFFLs can achieve noise rejection at the protein level. This is easily verified Condition 5.23 guarantees that the

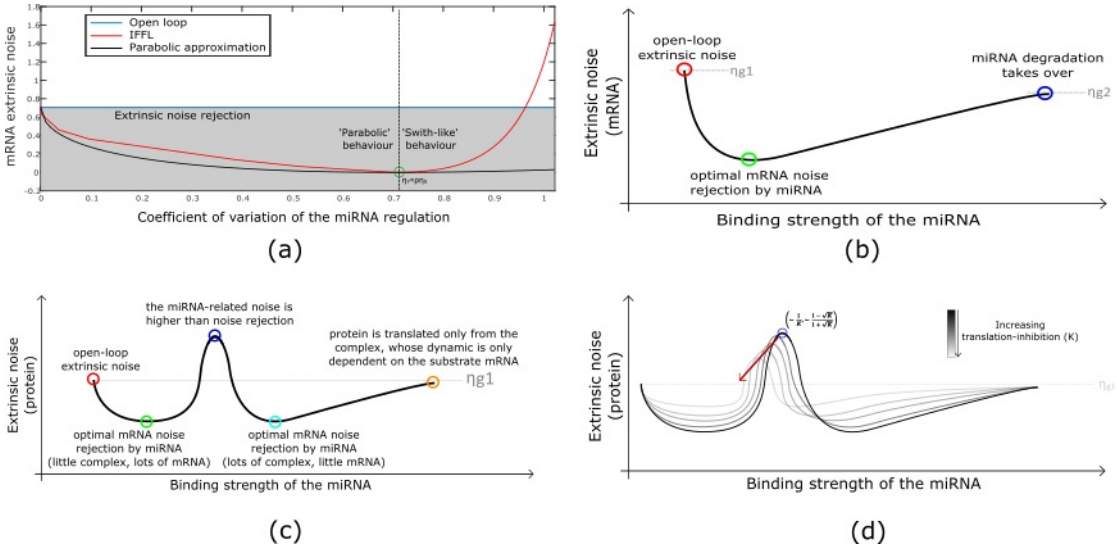


Figure 5.2: Extrinsic noise rejection in microRNA based IFFL's. a) Comparison between the approximation formula 5.12 and simulation of the IFFL system. b) Noise rejection at the mRNA level for a miRNA-based IFFL. By increasing the miRNA-mRNA binding strength from 0 to positive values, the extrinsic noise reaches a minimum for which the noise-cancellation effect of the miRNA is predominant. c) Noise rejection at the protein level for a miRNA-based IFFL. By increasing the miRNA-mRNA binding strength from 0 to positive values, the extrinsic noise reaches a minimum for which the noise-cancellation effect of the miRNA is predominant. After the minimum is reached, the noise introduced by the system overcome the beneficial noise-cancellation effect. When the binding rate is even higher, the mRNA is more present as the miRNA-mRNA complex than mRNA alone, and the complex extrinsic noise is dependent on the free mRNA only. d) Effect of increasing translation-inhibition on the noise-rejection property of the miRNA-based IFFL.

miRNA-mRNA complex does not translate into protein faster than mRNA alone would be degraded. If this were to happen, more protein would be produced from the complex than from the open loop mRNA. Therefore the miRNA regulation would not repress protein expression, but instead facilitate it. Hence, translation-inhibition ($K > 0$) could cancel the

noise-rejection property of the IFFL.

High miRNA-mRNA binding rate could lead to increased extrinsic noise

We investigated the relationship between the miRNA-mRNA binding rate (γ_s) and the extrinsic noise rejection properties of the IFFLs. We showed in the previous section that the amount of noise introduced by the miRNA regulation (η_Y^2) could be balanced by noise cancellation ($-2\rho\eta_X\eta_Y$). Hence, we computed the value of η_Y at the mRNA and the protein level and studied their dependencies to γ_s. These quantities are:

$$\eta_{Ym}(\gamma_s) = \frac{\gamma_s d\sigma_{g_2}}{\beta_m + \gamma_s d\mathrm{E}[g_2]} \tag{5.24}$$

$$\eta_{Yp}(\gamma_s) = \frac{\frac{\gamma_s d - K\beta_m}{(1+Kd\gamma_s \mathrm{E}[g_2])^2}\sigma_{g_2}}{\frac{\beta_m+\gamma_s d\mathrm{E}[g_2]}{1+Kd\gamma_s\mathrm{E}[g_2]}} \tag{5.25}$$

It could be easily shown that $\eta_Y m(\gamma_s)$ in 5.24 has a monotonic dependency on γ_s. In fact, the following hold:

$$\frac{d\eta_{Ym}}{d\gamma_s} > 0,\ \forall\gamma_s > 0 \qquad\qquad \lim_{\gamma_s\to\infty}\eta_{Ym}(\gamma_s) = \eta_{g_2} \tag{5.26}$$

these properties can be easily verified by studying 5.24. Properties 5.26 show that there is a maximum amount of noise that could be introduced by the miRNA regulation to the system at the mRNA level, and this is bounded by the amount of noise in the input signal g_2. The mRNA extrinsic noise tends to this value in the limit, althoughs for high values of γ_s, $m^* \to 0$. In this range, the dynamic of the mRNA is entirely defined by fluctuations in the miRNA, as it is all bounded to it. The limit tells us that if $\eta_{g_2} \leq \rho\eta_{g_1}$ then condition 5.21 is satisfied, and noise rejection is achieved for all values of γ_s. Since miRNA is less stable than mRNA, then this condition is often not satisfied. Hence, there exists a range of values of γ_s for which extrinsic noise is higher in the IFFL than in the open loop system (Figure 5.1).
At the protein level, high binding rate does not necessarily imply high noise. When the miRNA-mRNA binding rate γ_s is low, the miRNA regulation noise (η_Y) is low because the process of protein translation is unaffected by it. As γ_s increases, the process Y becomes

more and more relevant till it reaches its peak for γ_s^{max}. For even higher values of γ_s, protein translation occurs mainly through the miRNA-mRNA complex, because little mRNA is left. In this regime, the protein noise is independent on miRNA, since at steady state the complex is dependent only on the amount of the substrate mRNA. Hence, η_Y decreases till it reaches 0. In fact, $\eta_{Yp}(\gamma_s)$ is not monotonic on γ_s, and it reaches a maximum for:

$$\gamma_s^{max} = \sqrt{\frac{\beta_m}{K}}\frac{1}{d\mathrm{E}[g_2]} \qquad \rightarrow \qquad \eta_{Yp}(\gamma_s^{max}) = \frac{1-\sqrt{\beta_m K}}{1+\sqrt{\beta_m K}}\eta_{g_2} \tag{5.27}$$

We noticed that the maximum value of $\eta_{Yp}(\gamma_s^{max})$ is positive iff $1 - \beta_m K > 0$, which is assured by the noise-rejection condition 5.23. Equation 5.25 also shows that the value of η_{Yp} as $\gamma_s \rightarrow \infty$ is 0. Hence the maximum amount of noise that could be introduced by the miRNA regulation to the system at the protein level is reached for η_{Yp} (γ_s^{max}, Figure 5.1). This quantity is less or equal than η_{g_2}, satisfying the equality for $K = 0$ (complete translation-inhibition). In this equality case, the dynamic of the protein noise is equivalent to the mRNA one.

In summary, the protein noise follows closely the mRNA noise for low binding rate. As the binding rate increases, the miRNA-mRNA complex becomes more dominant and competes with mRNA to translate into protein: these two separate translation events lead to a bimodal protein population ('threshold effect'), which increases the total extrinsic noise (maximum highlighted in blue in Figure 5.2(c)). At higher binding rate, the miRNA-mRNA complex mostly dominates translation: protein noise is then dependent on the miRNA noise, reaching a second minimum (highlighted in light blue in Figure 5.2(c)).

Translation-inhibition modulates the functional range of miRNA extrinsic noise rejection

To address the importance of translation-inhibition for the miRNA-induced noise rejection, we analyzed the dependency of our results in the previous section to perturbation of the translation-inhibition parameter K. From 5.21 and 5.27 We notice that to obtain noise-

rejection independently on the value of γ_s, the following needs to be satisfied:

$$\frac{1-\sqrt{\beta_m K}}{1+\sqrt{\beta_m K}}\eta_{g_2} \leq \rho\eta_{g_1} \tag{5.28}$$

Condition 5.28 shows that for a fixed value of γ_s, the performance of the miRNA-IFFL could be tuned by increasing or decreasing the translation-inhibition parameter K or the mRNA degration β_m. In particular, increasing these parameters leads to a decrease in the maximum value of η_Y, which helps to guarantee condition 5.28. However, tuning there parameters have opposite effects on optimality: as shown in expression 5.27, the optimal value of γ_s is reduced if K is increased, and it increases if β_m is increased.

In summary, increasing translation-inhibition leads to a wider functional range of acceptable γ_s, but it reduces their noise-rejection effect on the protein extrinsic noise, as shown in Figure 5.1. In the special case for which the mRNA and miRNA are transcribed at the same rate ($g_1 = g_2 = g$), there exist an optimal miRNA-mRNA binding constant γ_s that reduce the extrinsic noise to 0. In fact, expression 5.12 is simplified to

$$\eta = (\eta_X - \eta_Y)^2 \tag{5.29}$$

At the mRNA level, it could be easily observed that η goes to 0 for large values of γ_s. Extrinsic variations of mRNA levels would be canceled out by equivalent variations of miRNA: each cell would then be reduced to the same mRNA steady state, defined by the fixed biochemical rates of degradation. This quantity tends to 0 as γ_s tends to ∞, so all the mRNA would exist in the miRNA-mRNA complex form. This behavior is conserved at the protein level if there is full translation inhibition. If one assumes that some of the complex gets translated, then there is an optimal value of γ_s for which extrinsic noise is minimized, and the protein concentration is not reduced to 0. This value is equivalent to:

$$\gamma_s^{opt} = \frac{1}{dE[g^{ss}]}\sqrt{\frac{\beta_m}{K}} \tag{5.30}$$

This values is a ratio between mRNA that escapes miRNA degradation (naturally degraded at rate β_m or translated as miRNA-mRNA complex) and total amount of miRNA ($dE[g^{ss}]$). These results are summarized in Figure 5.3.

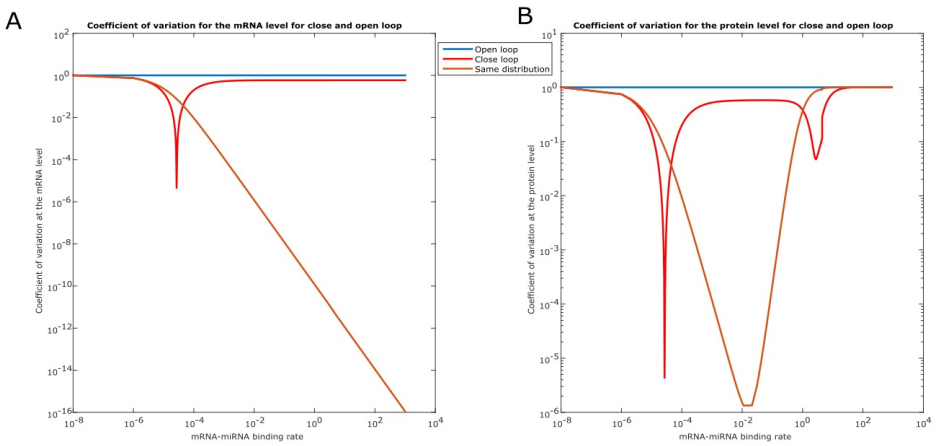


Figure 5.3: Extrinsic noise at the mRNA and at the protein level when mRNA and miRNA are expressed at the same time. A)mRNA extrinsic noise decreases as the miRNA binding rate increases, while B)protein noise reaches a minimum at a finite binding rate value depending on the amount of translation-inhibition

For systems with comparable means, IFFL systems lead to better intrinsic noise rejection

In order to study how microRNA based IFFLs effect of intrinsic noise, we make use of the Linear Noise Approximation method described in the 'Methods' section to calculate the different moments of the different components (g_1,g_2,M,S and P). To compare the intrinsic noise levels in IFFL systems to open loop systems, we obtain the closed form expressions for the coefficient of variance for both systems. Initially, we make no assumptions on the $\sigma_{g_1 g_2}$ term initially, which captures the steady state correlation between the two genes. This leads to the expression for mRNA noise in IFFLs as follows

$$\eta^2_{m,IFFL} = \underbrace{\frac{1}{m^\star} + \frac{1-g^\star}{g^\star}\frac{\Gamma_m}{\Gamma_m + K_1}}_{\text{inherent mRNA noise}} \underbrace{- \frac{\sigma^2_{g_1 g_2}\alpha_s \gamma_s}{g_1} f_1(K_1, K_2, \beta_s, \Gamma_m)}_{\text{miRNA noise reduction terms}} + O(\gamma_s^2) \tag{5.31}$$

Where, $K_1 = \lambda_1^+ + \lambda_1^-$, $K_2 = \lambda_2^+ + \lambda_2^-$ and f_1 is a strictly positive non-linear function in all constitutive variables. The star indicates the steady state value of any quantity. For any

IFFL system, $\sigma^2_{g_1 g_2}$ is always positive so the rightmost term (apart from the $O(\gamma_s^2)$ is always negative. Hence, this term necessarily leads to reduction of η^2. Since the sgFFL system has the largest value of $\sigma^2_{g_1 g_2}$, this system has the largest noise reduction among all IFFL systems. In general, higher correlation between genes leads to better intrinsic noise reduction. For uncorrrelated genes, this term is equal to zero. However, in order to compare it to the open loop system, we obtain a similar expression for the open loop system as seen below

$$\eta^2_{m,open} = \frac{1}{m^\star} + \frac{1-g^\star}{g^\star}\frac{\beta_m}{\beta_m + K_1} \tag{5.32}$$

It is difficult to compare equations 5.31 and 5.32 because the means would be different in the two cases and would lead to an intractable condition that would need to be satisfied for lower noise in the IFFL systems. However, it can be seen that when the $O(\gamma_s^2)$ is small enough to be ignored and the means of the open loop system and the IFFL system are similar ($\beta_m \simeq \Gamma_m$), then the noise in the IFFL systems is necessarily lower than the open loop system. It can also be seen that within this operating range, higher values of γ_s and α_s lead to better noise reduction. However, the higher order terms are all additive terms (positive) and result in increase of noise. Thus, when γ_s isn't small enough for $O(\gamma_s^2)$ terms to be ignored, then the intrinsic noise of the Open Loop system can be lower than the IFFL system, even for the same mean value. We randomly chose parameters from a realistic parameter regime and compared how the coefficient of variation of Open Loop and IFFL systems vary with γ_s as seen in Figure 5.4. It can be seen that the IFFL has higher CV than the Open Loop system at very low and very high values of γ_s but has lower noise at an intermediate parameter regime. The chosen parameter values and other related plots can be found in Supplementary methods.

For the protein level, we recalculate the moments using M, S and P as the three components, leaving out g_1 and g_2 to simplify the final expressions. Here the σ_{MS} serves the same function as $\sigma_{g_1 g_2}$, which leads to the following expression for protein noise for the general

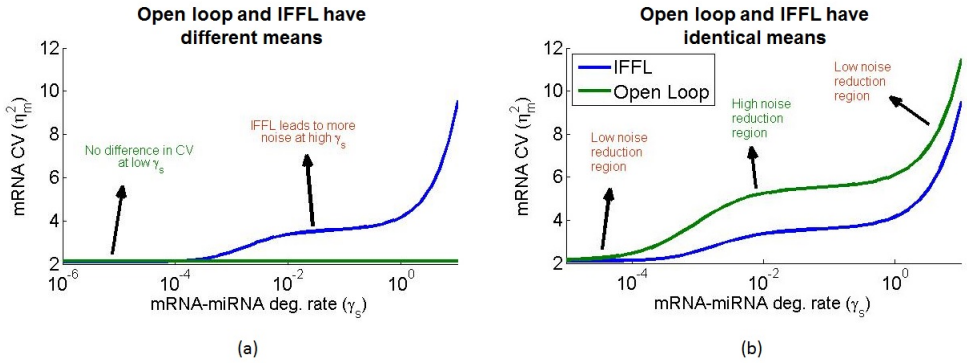


Figure 5.4: Coefficient of Variation at the mRNA level plotted against different values of micro RNA - mRNA binding strength for IFFL and open loop systems with identical & different means. a) When the means of the two systems are allowed to be different, IFFL's are seen to offer no significant reduction in intrinsic noise and increase in γ_s leads to increase in mRNA coefficient of variance. b) When the means of IFFL and open loop systems are identical, IFFL's lead to decrease in noise. At very low and very high values of γ_s the noise rejection is seen to be low whereas optimal noise rejection happens at intermediate values.

case.

$$\eta^2_{p,IFFL} = \overbrace{\frac{1}{p^\star} + \frac{A_p}{p^\star(\beta_p+\Gamma_m)}}^{\text{inherent protein noise}} \overbrace{- \frac{A_p\gamma_s\sigma^2_{MS}}{p^\star(\beta_p+\Gamma_m)}\left(\frac{1}{\Gamma_m}+\frac{1}{\beta_p+\beta_s}\right)}^{\text{mRNA-miRNA noise}} + \underbrace{\frac{\alpha_p K\sigma^2_{MS}}{p^\star}\left(\frac{1}{\beta_p+\Gamma_m}+\frac{1}{\beta_p+\beta_s}\right)}_{\text{translation inhibition noise}} + O(\gamma_s K, K^2) \tag{5.33}$$

Here, the terms containing $\gamma_s K$ and K^2 can be ignored because usually $\gamma_s K, K^2 << 1$. Similarly, the open loop noise can be derived in a similar fashion and is seen to be

$$\eta^2_{p,open} = \frac{1}{p^\star} + \frac{\alpha_p}{p^\star(\beta_p+\beta_m)} \tag{5.34}$$

Like in case of the mRNA, it can be seen that because of the difference in the mean protein values between the open loop and IFFL cases, it is difficult to make any unifying assertions

about increase/reduction of protein noise. It can however be seen that stronger steady state correlation between mRNA and microRNA leads to better noise reduction. Since this is highest in case of sgFFL's, they will have the highest noise reduction among systems with comparable parameter values. It can also be seen that incomplete translational inhibition (high K) leads to lower noise reduction at the protein level, since the contribution of translational inhibition is additive unlike miRNA based degradation. In the limiting case where $K = 0$, we see that for systems with similar means ($\Gamma_m \simeq \beta_m$, $A_p \simeq \alpha_p$), there is a higher guarantee of noise reduction at the protein level compared to the open loop system than at the mRNA level. This happens because for most systems $K < \gamma_s$. This observation seems to suggest that protein noise is better rejected than mRNA noise due to the presence of the additional translational inhibition pathway, as was also observed in case of extrinsic noise. It is also interesting to note that [3] observed experimentally that microRNA based IFFL's lead to lower noise when the means of the IFFL and open loop systems are identical, validating our assertions.

Additionally, the authors of [113], compared translationally regulated feedback loops having similar means to open loop systems and found intrinsic noise to be reduced only when the hill coefficient for the system was greater than 1 (meaning in the presence of cooperativity). Our observations suggest that microRNA based IFFLs offer an advantage over translational regulation based systems by virtue of the post transcriptional regulation by microRNAs. This observation is consistent with [114], where the authors demonstrate that post-transcriptional control offers advantages over translational feedback in intrinsic noise reduction.

MicroRNA-mRNA interaction modulates protein noise by reducing mRNA noise at lower frequency

In this sub-section we study in greater detail the advantages imparted by the post transcriptional control by microRNAs and their effect on protein and mRNA noise. As noted in the previous sections, the noise reduction offered by IFFLs at the mRNA level is more restrictive

than at the protein level. We try to investigate this anomalous effect by studying the effect of mRNA-microRNA interaction on the frequency spectrum of mRNA. We hypothesize that the mRNA-microRNA interaction increases the mRNA intrinsic noise at higher frequencies while reducing noise at lower frequencies. In such as case, the translation process being much slower, would behave like a low pass filter and filter out the high frequency components of the noise. This in turn would lead to an overall reduction in noise. To verify this hypothesis we calculate the autocovariance of the steady mRNA for the IFFL system as seen below

$$C_M(\tau) = \hat{A}e^{-\Gamma_m|t|} - \hat{B}e^{-\beta_s|t|} - \hat{C}e^{-K_1|t|} - \hat{D}e^{-K_2|t|} \tag{5.35}$$

Where, A,B,C and D are functions of the various covariance terms, whose closed form expressions are provided in the supplementary methods. As described in the methods section, the R_M is simply $R_M(\tau) = \sigma_M^2(\tau) + M^2$. We then calculate the power spectrum of the stochastic signal as in [115], by taking a Fourier transform of this we see that it is of the form

$$S_M(\omega) = -\hat{C}\frac{2K_1}{\omega^2 + K_1^2} + \underbrace{\hat{A}\frac{2\Gamma_m}{\omega^2 + \Gamma_m^2}}_{\text{higher in IFFL}} - \underbrace{\hat{B}\frac{2\beta_s}{\omega^2 + \beta_s^2} - \hat{D}\frac{2K_2}{\omega^2 + K_2^2}}_{\text{reduced due to IFFL}} + M^2\delta(\omega) \tag{5.36}$$

Since, β_s is usually smaller than Γ_m on account of microRNA's being more stable than mRNA, this causes a reduction of noise at low frequencies and increase of noise at higher frequencies (compared to the open loop). Moreover, unlike the open loop system, for any microRNA regulated system there is a reduction of noise at lower frequencies but increase of noise at higher frequencies. Thus, irrespective of the correlation between the mRNA and the microRNA, there will be increase in intrinsic noise at high frequencies. While this might increase overall noise at the mRNA level, this increase of noise helps reduce protein level noise since the translation process acts like a low pass filter. However, it can also be seen that strong microRNA-mRNA correlation leads to better noise reduction at lower frequencies which means that the sgFFL will be most effective in noise reduction at lower frequencies (demonstrated in supplementary methods). Figure 5.5 b) demonstrates this phenomenon on the same parameter values as those selected in the previous subsection.

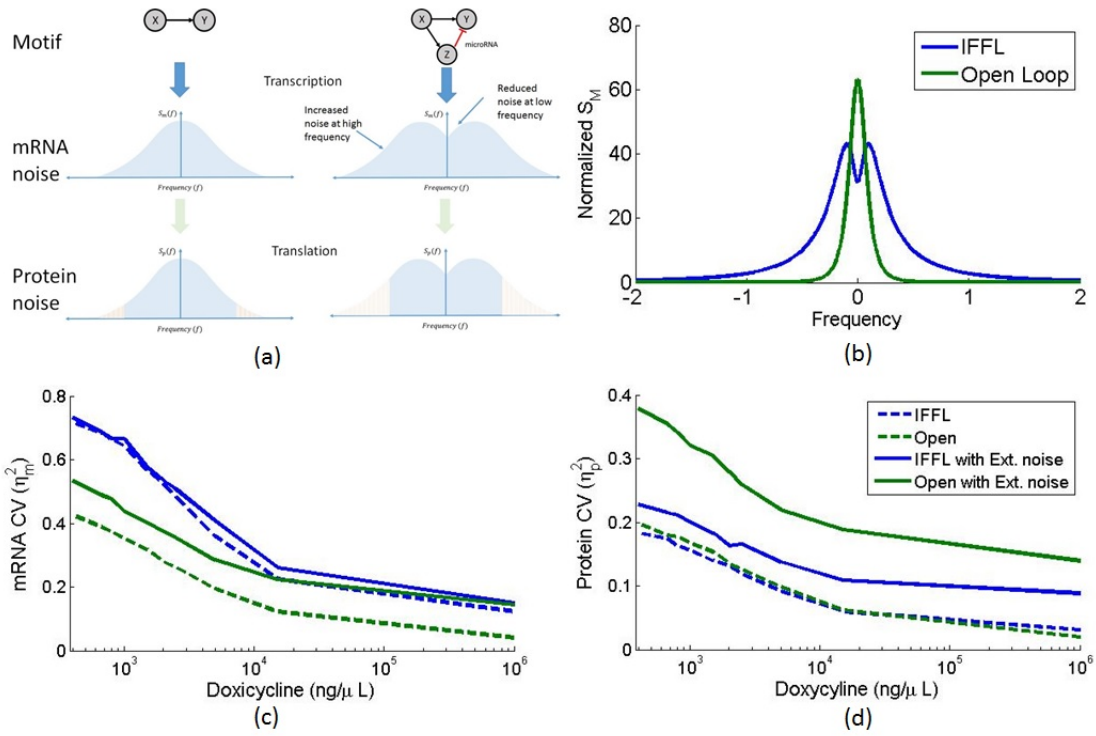


Figure 5.5: Effect of mRNA-microRNA interaction on mRNA noise spectrum. a) Illustration of the noise filtering phenomenon at transcriptional and translational levels. b) Simulation of mRNA noise spectrum $S_m(f)$ for IFFL and Open loop systems. It can be seen here that in the presence of microRNA mediated regulation, there is a sharp drop of noise at lower frequencies while there is an attenuation of noise at higher frequencies. This is caused by the fast time scale of mRNA-microRNA interaction. Since the translation process is much slower, it can act as a low pass filter and result in lower noise at the protein level compared to the open loop system. c-d) Effect of IFFL on mRNA and protein noise at different induction levels. It can be seen that the mRNA noise is increased in the IFFL system than the open loop system. It is also seen that the extrinsic noise is better rejected by the IFFL system than intrinsic noise.

Stochastic simulations with experimentally validated model verifies theoretic findings

According to our analysis from previous sections, optimal extrinsic noise rejection is achieved when the correlation ρ is equal to 1. This case corresponds to the sgFFL case. To test our

observations on real parameters, we use experimental data from [3] and perform parameter identification in MATLAB. We then ran two sets of stochastic simulations with the estimated parameters as explained in the methods section. Shown in Figures 5.5 c-d) are the mRNA and protein level plots of the coefficient of variances at different induction levels for both sets of simulations. We observe here that there is indeed an increase in noise at the mRNA level and reduction in noise at the protein level. The reason for this is the frequency shift occurring at the mRNA level, which has been explained in the previous subsection. It is also seen that the IFFL has little change between the two sets of simulations whereas the Open loop shows a large increase when Extrinsic noise is added. This demonstrates that the IFFL does indeed reject Extrinsic noise better than the Open loop system.

5.4 Conclusions

In this work we have outlined the noise reduction properties that result from microRNA based control of genes and show specifically that these are amplified in the case of IFFL systems. We studied the effect of the two different noise reduction mechanisms and and the effect of microRNA-mRNA interaction strength on each of these. While studying the relative contributions of the microRNA based regulation mechanisms, we also note that the effect of noise reduction is not identical at the mRNA and protein level. The noise reduction at protein level is shown to occur across a larger parameter regime than the mRNA level. This led to the finding that any microRNA based regulation results in lowering of mRNA noise at lower frequencies and increase of noise at higher frequencies. Since translation acts as a low pass filter, this gives an additional layer of control over total noise reduction. These results also provide important insights into how IFFLs can be tuned to maximize noise reduction, which might be of importance in Synthetic Biology.

We studied the noise components separately to get an understanding of the relative efficacy of such IFFL systems in reducing each component noise. Our observations seem to suggest that IFFLs are better at reducing extrinsic noise than intrinsic noise, which is consistent with experimental observations from [3]. This manifests in the adaption to

input concentrations by microRNA based IFFLs as noted in [3, 104]. Future work will be aimed at understanding how microRNA based network motifs can be tuned for optimal noise reduction. It is also of interest to study the factors that impact the level of translational inhibition in microRNA mediated repression.

Chapter 6

CONCLUSION AND FUTURE WORK

A key accomplishment of the work presented in this thesis has been the development of tools that enable the study of cellular dynamics and heterogeneity at a single cell resolution. The ultimate goal of this work is to enable synthetic biologists to engineer pathways in a more targeted manner. To this end, we have presented two tools, UNCURL(in Chapter 3) and PIPER (in Chapter 4), which address two very different challenges in the study of scRNA-Seq data.

UNCURL is a pre-processing tool for scRNA-Seq data which uses a sampling distribution aware method to remove the effects of sampling in the single cell sequencing pipeline. Additionally, UNCURL allows the integration of qualitative prior information into the data analysis pipeline through a framework called QualNorm. We demonstrate that pre-processing using UNCURL leads to better identification of cellular heterogeneity and cellular dynamics using common state-of-the-art downstream algorithms. We envision that this will provide a more accurate understanding of the various cell types in different tissues, as well as a more clear awareness of the differentiation processes that forms various cell types.

PIPER is a novel pipeline to identify key regulators of biological progressions from multi-stage scRNA-Seq data. PIPER first uses a scRNA-Seq specific state-of-the-art network inference technique to jointly infer the gene regulatory network structure at different stages (for which data is available). Second, it provides a novel approach to identify the amount of perturbation in each gene between consecutive stages. Finally, it utilizes this information, along with the estimated condition specific graphs, to infer potential key regulators at each stage of the biological progression. PIPER is a useful tool in identifying the genes that lead to disease progression or cell differentiation. This information can in turn result in more

targeted approaches to re-engineering pathways associated with various diseases.

We then took a diversion into the study of a particular biological network motif (the micro-RNA mediated incoherent feedforward loop) at the single cell level. We aimed to identify the effects of this particular network motif on the different sources of cellular variability (i.e. noise), namely 1) intrinsic and 2) extrinsic noise. We also studied the effect of two constitutive mechanisms: 1) micro-RNA mediated degradation and 2) translation inhibition on the enhancing/reducing the different types of noise. This study illuminates the complex mechanism of noise reduction that is often attributed to micro-RNA mediated IFFLs [3] and paves way for more model aware engineering of such systems.

6.1 Future work

While we have already discussed the possible future extensions of the individual approaches in their respective chapters, we now present a few promising directions for future work that are closely related to our prior work.

- **Interactive visualization and expert mediated state estimation:** A large part of the biological data analysis workflow is currently done manually by experts. Common tasks include splitting/merging clusters depending on biological relevance and identifying potential outlier points/clusters. This process is usually done iteratively as described in [9]. While UNCURL allows users to intgrate their qualitative priors into the state estimation process, this is currently done at the beginning of the state estimation process and is quite limited in the type of priors it can handle. We are currently working on an interactive visualization and iterative refinement tool along the lines of [116] to enable a more automated incorporation of expert knowledge into the state estimation process.

- **Fast hierarchical state estimation:** While UNCURL is demonstrated to have rapid run times compared to most common scRNA-Seq analysis algorithms, it scales linearly in the number of cell types in a dataset. Therefore, UNCURL can be slow for datasets

containing very large numbers of cell types. However, in such cases the cell types typically form a hierarchy. We speculate a significant reduction of run time is possible if state estimation is performed in an hierarchical manner. While this method relies on the accuracy of the state estimation at each layer of the hierarchy, future work will aim to develop methods which improve the speed and accuracy of the process.

BIBLIOGRAPHY

[1] C. Trapnell, D. Cacchiarelli, J. Grimsby, P. Pokharel, S. Li, *et al.*, "The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells," *Nature biotechnology*, vol. 32, no. 4, pp. 381–386, 2014.

[2] A. Raj and A. van Oudenaarden, "Nature, nurture, or chance: stochastic gene expression and its consequences," *Cell*, vol. 135, no. 2, pp. 216–226, 2008.

[3] T. J. Strovas, A. B. Rosenberg, B. E. Kuypers, R. A. Muscat, and G. Seelig, "A microrna-based single-gene circuit buffers protein synthesis rates against perturbations," *ACS Synthetic Biology*, 2014.

[4] N. K. Hanchate, K. Kondoh, Z. Lu, D. Kuang, X. Ye, *et al.*, "Single-cell transcriptomics reveals receptor transformations during olfactory neurogenesis," *Science*, p. aad2456, Nov. 2015.

[5] C. R. Blyth, "On simpson's paradox and the sure-thing principle," *Journal of the American Statistical Association*, vol. 67, no. 338, pp. 364–366, 1972.

[6] A. Zeisel, A. B. Muñoz-Manchado, S. Codeluppi, P. Lönnerberg, G. La Manno, *et al.*, "Cell types in the mouse cortex and hippocampus revealed by single-cell rna-seq," *Science*, vol. 347, no. 6226, pp. 1138–1142, 2015.

[7] L. Bonetta, "Whole-genome sequencing breaks the cost barrier," *Cell*, vol. 141, no. 6, pp. 917–919, 2010.

[8] A. M. Klein, L. Mazutis, I. Akartuna, N. Tallapragada, A. Veres, *et al.*, "Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells," *Cell*, vol. 161, no. 5, pp. 1187–1201, 2015.

[9] A. B. Rosenberg, C. Roco, R. A. Muscat, A. Kuchina, S. Mukherjee, *et al.*, "Scaling single cell transcriptomics through split pool barcoding," *BioRxiv*, p. 105163, 2017.

[10] N. K. Hanchate, K. Kondoh, Z. Lu, D. Kuang, X. Ye, *et al.*, "Single-cell transcriptomics reveals receptor transformations during olfactory neurogenesis," *Science*, vol. 350, no. 6265, pp. 1251–1255, 2015.

[11] P. V. Kharchenko, L. Silberstein, and D. T. Scadden, "Bayesian approach to single-cell differential expression analysis," *Nature methods*, vol. 11, no. 7, pp. 740–742, 2014.

[12] R. Satija, J. A. Farrell, D. Gennert, A. F. Schier, and A. Regev, "Spatial reconstruction of single-cell gene expression data," *Nature biotechnology*, vol. 33, no. 5, pp. 495–502, 2015.

[13] K. Achim, J.-B. Pettit, L. R. Saraiva, D. Gavriouchkina, T. Larsson, *et al.*, "High-throughput spatial mapping of single-cell rna-seq data to tissue of origin," *Nature biotechnology*, vol. 33, no. 5, pp. 503–509, 2015.

[14] A. Wagner, A. Regev, and N. Yosef, "Revealing the vectors of cellular identity with single-cell genomics," *Nature biotechnology*, vol. 34, no. 11, pp. 1145–1160, 2016.

[15] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*, vol. 1. Springer series in statistics Springer, Berlin, 2001.

[16] D. Donoho and V. Stodden, "When does non-negative matrix factorization give a correct decomposition into parts?," in *Advances in neural information processing systems*, p. None, 2003.

[17] Y.-X. Wang and Y.-J. Zhang, "Nonnegative matrix factorization: A comprehensive review," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 6, pp. 1336–1353, 2013.

[18] R. Graham and P. Winkler, "On isometric embeddings of graphs," *Transactions of the American mathematical Society*, vol. 288, no. 2, pp. 527–536, 1985.

[19] L. Van Der Maaten, E. Postma, and J. Van den Herik, "Dimensionality reduction: a comparative," *J Mach Learn Res*, vol. 10, pp. 66–71, 2009.

[20] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[21] J. D. Welch, A. J. Hartemink, and J. F. Prins, "Slicer: inferring branched, nonlinear cellular trajectories from single cell rna-seq data," *Genome biology*, vol. 17, no. 1, p. 1, 2016.

[22] Y. R. Wang and H. Huang, "Review on statistical methods for gene network reconstruction using expression data," *Journal of theoretical biology*, vol. 362, pp. 53–61, 2014.

[23] R. Gill, S. Datta, and S. Datta, "A statistical framework for differential network analysis from microarray data," *BMC bioinformatics*, vol. 11, no. 1, p. 1, 2010.

[24] A. P. Parikh, W. Wu, R. E. Curtis, and E. P. Xing, "Treegl: reverse engineering tree-evolving gene networks underlying developing biological lineages," *Bioinformatics*, vol. 27, no. 13, pp. i196–i204, 2011.

[25] T. Ideker and N. J. Krogan, "Differential network biology," *Molecular systems biology*, vol. 8, no. 1, p. 565, 2012.

[26] P. Danaher, P. Wang, and D. M. Witten, "The joint graphical lasso for inverse covariance estimation across multiple classes," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 76, no. 2, pp. 373–397, 2014.

[27] P. S. Swain, M. B. Elowitz, and E. D. Siggia, "Intrinsic and extrinsic contributions to stochasticity in gene expression," *Proceedings of the National Academy of Sciences*, vol. 99, no. 20, pp. 12795–12800, 2002.

[28] G. W. Oehlert, "A note on the delta method," *The American Statistician*, vol. 46, no. 1, pp. 27–29, 1992.

[29] A. M. Klein, L. Mazutis, I. Akartuna, N. Tallapragada, A. Veres, *et al.*, "Droplet Barcoding for Single-Cell Transcriptomics Applied to Embryonic Stem Cells," *Cell*, vol. 161, pp. 1187–1201, May 2015.

[30] G. X. Y. Zheng, J. M. Terry, P. Belgrader, P. Ryvkin, Z. W. Bent, *et al.*, "Massively parallel digital transcriptional profiling of single cells," *Nature Communications*, vol. 8, p. 14049, Jan. 2017.

[31] D. Grun and A. van Oudenaarden, "Design and Analysis of Single-Cell Sequencing Experiments," *Cell*, vol. 163, pp. 799–810, Nov. 2015.

[32] A. Zeisel, A. B. Muoz-Manchado, S. Codeluppi, P. Lnnerberg, G. L. Manno, *et al.*, "Cell types in the mouse cortex and hippocampus revealed by single-cell RNA-seq," *Science*, vol. 347, pp. 1138–1142, Mar. 2015.

[33] M. Baron, A. Veres, S. L. Wolock, A. L. Faust, R. Gaujoux, *et al.*, "A Single-Cell Transcriptomic Map of the Human and Mouse Pancreas Reveals Inter- and Intra-cell Population Structure," *Cell Systems*, vol. 3, pp. 346–360.e4, Oct. 2016.

[34] C. Trapnell, D. Cacchiarelli, J. Grimsby, P. Pokharel, S. Li, *et al.*, "The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells," *Nature Biotechnology*, vol. 32, pp. 381–386, Apr. 2014.

[35] J. Shin, D. A. Berg, Y. Zhu, J. Y. Shin, J. Song, *et al.*, "Single-Cell RNA-Seq with Waterfall Reveals Molecular Cascades underlying Adult Neurogenesis," *Cell Stem Cell*, vol. 17, pp. 360–372, Sept. 2015.

[36] J. D. Welch, A. J. Hartemink, and J. F. Prins, "SLICER: inferring branched, nonlinear cellular trajectories from single cell RNA-seq data," *Genome Biology*, vol. 17, p. 106, 2016.

[37] M. Setty, M. D. Tadmor, S. Reich-Zeliger, O. Angel, T. M. Salame, *et al.*, "Wishbone identifies bifurcating developmental trajectories from single-cell data," *Nature Biotechnology*, vol. 34, pp. 637–645, June 2016.

[38] R. Satija, J. A. Farrell, D. Gennert, A. F. Schier, and A. Regev, "Spatial reconstruction of single-cell gene expression data," *Nature Biotechnology*, vol. 33, p. 495, May 2015.

[39] K. B. Halpern, R. Shenhav, O. Matcovitch-Natan, B. Tth, D. Lemze, *et al.*, "Single-cell spatial reconstruction reveals global division of labour in the mammalian liver," *Nature*, vol. 542, p. 352, Feb. 2017.

[40] C. R. Blyth, "On Simpson's Paradox and the Sure-Thing Principle," *Journal of the American Statistical Association*, vol. 67, pp. 364–366, June 1972.

[41] C. Trapnell, "Defining cell types and states with single-cell genomics," *Genome Research*, vol. 25, pp. 1491–1498, Oct. 2015.

[42] A. Wagner, A. Regev, and N. Yosef, "Revealing the vectors of cellular identity with single-cell genomics," *Nature Biotechnology*, vol. 34, pp. 1145–1160, Nov. 2016.

[43] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1988.

[44] C. Ding and X. He, "K-means Clustering via Principal Component Analysis," in *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, (New York, NY, USA), pp. 29–, ACM, 2004.

[45] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 2, pp. 433–459, July 2010.

[46] S. T. Roweis and L. K. Saul, "Nonlinear Dimensionality Reduction by Locally Linear Embedding," *Science*, vol. 290, pp. 2323–2326, Dec. 2000.

[47] L. v. d. Maaten and G. Hinton, "Visualizing Data using t-SNE," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[48] D. Grun, L. Kester, and A. v. Oudenaarden, "Validation of noise models for single-cell transcriptomics," *Nature Methods*, vol. 11, p. 637, June 2014.

[49] D. v. Dijk, J. Nainys, R. Sharma, P. Kathail, A. J. Carr, *et al.*, "MAGIC: A diffusion-based imputation method reveals gene-gene interactions in single-cell RNA-sequencing data," *bioRxiv*, p. 111591, Feb. 2017.

[50] B. Wang, J. Zhu, E. Pierson, D. Ramazzotti, and S. Batzoglou, "Visualization and analysis of single-cell RNA-seq data by kernel-based similarity learning," *Nature Methods*, vol. 14, pp. 414–416, Mar. 2017.

[51] E. Pierson and C. Yau, "ZIFA: Dimensionality reduction for zero-inflated single-cell gene expression analysis," *Genome Biology*, vol. 16, p. 241, Nov. 2015.

[52] D. D. Lee and H. S. Seung, "Algorithms for Non-negative Matrix Factorization," in *Advances in Neural Information Processing Systems 13* (T. K. Leen, T. G. Dietterich, and V. Tresp, eds.), pp. 556–562, MIT Press, 2001.

[53] J.-P. Brunet, P. Tamayo, T. R. Golub, and J. P. Mesirov, "Metagenes and molecular pattern discovery using matrix factorization," *Proceedings of the national academy of sciences*, vol. 101, no. 12, pp. 4164–4169, 2004.

[54] C. Shao and T. Hofer, "Robust classification of single-cell transcriptome data by non-negative matrix factorization," *Bioinformatics*, vol. 33, pp. 235–242, Jan. 2017.

[55] S. Anders and W. Huber, "Differential expression analysis for sequence count data," *Genome Biology*, vol. 11, p. R106, Oct. 2010.

[56] H. H. Bauschke, J. Bolte, and M. Teboulle, "A Descent Lemma Beyond Lipschitz Gradient Continuity: First-Order Methods Revisited and Applications," *Mathematics of Operations Research*, vol. 42, pp. 330–348, Nov. 2016.

[57] W. Perkins, M. Tygert, and R. Ward, "Computing the confidence levels for a root-mean-square test of goodness-of-fit," *Applied Mathematics and Computation*, vol. 217, pp. 9072–9084, July 2011.

[58] A. N. Langville, C. D. Meyer, R. Albright, J. Cox, and D. Duling, "Initializations for the nonnegative matrix factorization," in *Proceedings of the twelfth ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 23–26, Citeseer, 2006.

[59] C. Boutsidis and E. Gallopoulos, "SVD based initialization: A head start for nonnegative matrix factorization," *Pattern Recognition*, vol. 41, pp. 1350–1362, Apr. 2008.

[60] D. Arthur and S. Vassilvitskii, "K-means++: The Advantages of Careful Seeding," in *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, (Philadelphia, PA, USA), pp. 1027–1035, Society for Industrial and Applied Mathematics, 2007.

[61] K. Gu, H. K. T. Ng, M. L. Tang, and W. R. Schucany, "Testing the ratio of two poisson rates," *Biometrical Journal*, vol. 50, no. 2, pp. 283–298, 2008.

[62] D. Usoskin, A. Furlan, S. Islam, H. Abdo, P. Lnnerberg, *et al.*, "Unbiased classification of sensory neuron types by large-scale single-cell RNA sequencing," *Nature Neuroscience*, vol. 18, no. 1, pp. 145–153, 2015.

[63] B. Tasic, V. Menon, T. N. Nguyen, T. K. Kim, T. Jarsky, *et al.*, "Adult mouse cortical cell taxonomy revealed by single cell transcriptomics," *Nature Neuroscience*, vol. 19, p. nn.4216, Jan. 2016.

[64] X. Qiu, Q. Mao, Y. Tang, L. Wang, R. Chawla, *et al.*, "Reversed graph embedding resolves complex single-cell trajectories," *Nature Methods*, vol. 14, p. 979, Aug. 2017.

[65] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, *et al.*, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, p. 28252830, Oct. 2011.

[66] 10XGenomics, "1m_neurons - Datasets - Single Cell Gene Expression - Official 10x Genomics Support," 2017.

[67] M. Barron and J. Li, "Identifying and removing the cell-cycle effect from single-cell RNA-sequencing data," *Scientific Reports*, vol. 6, p. 33892, 2016.

[68] B. Treutlein, D. G. Brownfield, A. R. Wu, N. F. Neff, G. L. Mantalas, *et al.*, "Reconstructing lineage hierarchies of the distal lung epithelium using single-cell rna-seq," *Nature*, vol. 509, no. 7500, pp. 371–375, 2014.

[69] S. Mukherjee, Y. Zhang, S. Kannan, and G. Seelig, "Prior knowledge and sampling model informed learning with single cell rna-seq data," *bioRxiv*, p. 142398, 2017.

[70] A. A. Margolin, I. Nemenman, K. Basso, C. Wiggins, G. Stolovitzky, *et al.*, "Aracne: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context," *BMC bioinformatics*, vol. 7, no. Suppl 1, p. S7, 2006.

[71] O. Alarcón Heras and A. Gordi Margalef, "Partial correlation: Network analysis," 2014.

[72] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, 2008.

[73] Q. Tang, S. Sun, and J. Xu, "Learning scale-free networks by dynamic node specific degree prior," in *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pp. 2247–2255, 2015.

[74] S. Sun, H. Wang, and J. Xu, "Inferring block structure of graphical models in exponential families.," in *AISTATS*, 2015.

[75] M. Grechkin, M. Fazel, D. Witten, and S.-I. Lee, "Pathway graphical lasso," in *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[76] K. Mohan, P. London, M. Fazel, D. Witten, and S.-I. Lee, "Node-based learning of multiple gaussian graphical models," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 445–488, 2014.

[77] G. Allen, Z. Liu, *et al.*, "A local poisson graphical model for inferring networks from sequencing data," *NanoBioscience, IEEE Transactions on*, vol. 12, no. 3, pp. 189–198, 2013.

[78] M. Grechkin, B. A. Logsdon, A. J. Gentles, and S.-I. Lee, "Identifying network perturbation in cancer," *bioRxiv*, p. 040394, 2016.

[79] T. M. Kodinariya and P. R. Makwana, "Review on determining number of cluster in k-means clustering," *International Journal*, vol. 1, no. 6, pp. 90–95, 2013.

[80] L. Li, D. Alderson, J. C. Doyle, and W. Willinger, "Towards a theory of scale-free graphs: Definition, properties, and implications," *Internet Mathematics*, vol. 2, no. 4, pp. 431–523, 2005.

[81] A. R. Borneman, J. A. Leigh-Bell, H. Yu, P. Bertone, M. Gerstein, *et al.*, "Target hub proteins serve as master regulators of development in yeast," *Genes & development*, vol. 20, no. 4, pp. 435–448, 2006.

[82] L. M. Przybyla and J. Voldman, "Attenuation of extrinsic signaling reveals the importance of matrix remodeling on maintenance of embryonic stem cell self-renewal," *Proceedings of the National Academy of Sciences*, vol. 109, no. 3, pp. 835–840, 2012.

[83] Y. Tay, J. Zhang, A. M. Thomson, B. Lim, and I. Rigoutsos, "Micrornas to nanog, oct4 and sox2 coding regions modulate embryonic stem cell differentiation," *Nature*, vol. 455, no. 7216, pp. 1124–1128, 2008.

[84] H. Niwa, J.-i. Miyazaki, and A. G. Smith, "Quantitative expression of oct-3/4 defines differentiation, dedifferentiation or self-renewal of es cells," *Nature genetics*, vol. 24, no. 4, pp. 372–376, 2000.

[85] H. Xu, Y.-S. Ang, A. Sevilla, I. R. Lemischka, and A. Ma'ayan, "Construction and validation of a regulatory network for pluripotency and self-renewal of mouse embryonic stem cells," *PLoS Comput Biol*, vol. 10, no. 8, p. e1003777, 2014.

[86] S. Masui, Y. Nakatake, Y. Toyooka, D. Shimosato, R. Yagi, *et al.*, "Pluripotency governed by sox2 via regulation of oct3/4 expression in mouse embryonic stem cells," *Nature cell biology*, vol. 9, no. 6, pp. 625–635, 2007.

[87] Y.-H. Loh, Q. Wu, J.-L. Chew, V. B. Vega, W. Zhang, *et al.*, "The oct4 and nanog transcription network regulates pluripotency in mouse embryonic stem cells," *Nature genetics*, vol. 38, no. 4, pp. 431–440, 2006.

[88] J. L. Kopp, B. D. Ormsbee, M. Desler, and A. Rizzino, "Small increases in the level of sox2 trigger the differentiation of mouse embryonic stem cells," *Stem cells*, vol. 26, no. 4, pp. 903–911, 2008.

[89] X. Chen, "A microrna as a translational repressor of apetala2 in arabidopsis flower development," *Science*, vol. 303, no. 5666, pp. 2022–2025, 2004.

[90] A. Marson, S. S. Levine, M. F. Cole, G. M. Frampton, T. Brambrink, *et al.*, "Connecting microrna genes to the core transcriptional regulatory circuitry of embryonic stem cells," *Cell*, vol. 134, no. 3, pp. 521–533, 2008.

[91] R. Garzon, G. Marcucci, and C. M. Croce, "Targeting micrornas in cancer: rationale, strategies and challenges," *Nature reviews Drug discovery*, vol. 9, no. 10, pp. 775–789, 2010.

[92] S. Volinia, G. A. Calin, C.-G. Liu, S. Ambs, A. Cimmino, *et al.*, "A microrna expression signature of human solid tumors defines cancer gene targets," *Proceedings of the National academy of Sciences of the United States of America*, vol. 103, no. 7, pp. 2257–2261, 2006.

[93] S. J. Gosline, A. M. Gurtan, C. K. JnBaptiste, A. Bosson, P. Milani, *et al.*, "Elucidating microrna regulatory networks using transcriptional, post-transcriptional, and histone modification measurements," *Cell reports*, vol. 14, no. 2, pp. 310–319, 2016.

[94] A. Manuscript, "Processes," vol. 149, no. 3, pp. 515–524, 2013.

[95] J. M. Schmiedel, S. L. Klemm, Y. Zheng, A. Sahay, N. Blüthgen, *et al.*, "Microrna control of protein expression noise," *Science*, vol. 348, no. 6230, pp. 128–132, 2015.

[96] M. Osella, C. Bosia, D. Corá, and M. Caselle, "The role of incoherent microRNA-mediated feedforward loops in noise buffering," *PLoS Computational Biology*, vol. 7, no. 3, 2011.

[97] X. Li, J. J. Cassidy, C. A. Reinke, S. Fischboeck, and W. Richard, "NIH Public Access," vol. 137, no. 2, pp. 273–282, 2010.

[98] A. M. Gurtan and P. A. Sharp, "The role of mirnas in regulating gene expression networks," *Journal of molecular biology*, vol. 425, no. 19, pp. 3582–3600, 2013.

[99] E. J. Lee, M. Baek, Y. Gusev, D. J. Brackett, G. J. Nuovo, *et al.*, "Systematic evaluation of microrna processing patterns in tissues, cell lines, and tumors," *Rna*, vol. 14, no. 1, pp. 35–42, 2008.

[100] M. Selbach, B. Schwanhäusser, N. Thierfelder, Z. Fang, R. Khanin, *et al.*, "Widespread changes in protein synthesis induced by microrna," *nature*, vol. 455, no. 7209, pp. 58–63, 2008.

[101] J. Tsang, J. Zhu, and A. van Oudenaarden, "MicroRNA-Mediated Feedback and Feedforward Loops Are Recurrent Network Motifs in Mammals," *Molecular Cell*, vol. 26, no. 5, pp. 753–767, 2007.

[102] R. Shalgi, D. Lieber, M. Oren, and Y. Pilpel, "Global and local architecture of the mammalian microrna–transcription factor regulatory network," *PLoS Comput Biol*, vol. 3, no. 7, p. e131, 2007.

[103] N. J. Martinez and A. J. M. Walhout, "The interplay between transcription factors and microRNAs in genome-scale regulatory networks.," *BioEssays : news and reviews in molecular, cellular and developmental biology*, vol. 31, no. 4, pp. 435–445, 2009.

[104] L. Bleris, Z. Xie, D. Glass, A. Adadey, E. Sontag, *et al.*, "Synthetic incoherent feedforward circuits show adaptation to the amount of their genetic template," *Molecular Systems Biology*, vol. 7, no. 1, 2011.

[105] S. Legewie, D. Dienst, A. Wilde, H. Herzel, and I. M. Axmann, "Small rnas establish delays and temporal thresholds in gene expression," *Biophysical journal*, vol. 95, no. 7, pp. 3232–3238, 2008.

[106] E. Levine, Z. Zhang, T. Kuhlman, and T. Hwa, "Quantitative characteristics of gene regulation by small rna," *PLoS biology*, vol. 5, no. 9, p. e229, 2007.

[107] P. Mehta, S. Goyal, and N. S. Wingreen, "A quantitative comparison of srna-based and protein-based gene regulation," *Molecular systems biology*, vol. 4, no. 1, 2008.

[108] R. J. Bloom, S. M. Winkler, and C. D. Smolke, "A quantitative framework for the forward design of synthetic mirna circuits," *Nature methods*, 2014.

[109] N. G. Van Kampen, *Stochastic processes in physics and chemistry*, vol. 1. Elsevier, 1992.

[110] J. Paulsson, "Models of stochastic gene expression," *Physics of life reviews*, vol. 2, no. 2, pp. 157–175, 2005.

[111] M. Scott and B. Ingalls, "Using the linear noise approximation to characterize molecular noise in reaction pathways," in *Proceedings of the AIChE Conference on Foundations of Systems Biology in Engineering (FOSBE), Santa Barbara, California*, Citeseer, 2005.

[112] D. T. Gillespie, "Exact stochastic simulation of coupled chemical reactions," *The journal of physical chemistry*, vol. 81, no. 25, pp. 2340–2361, 1977.

[113] A. Singh and J. P. Hespanha, "Optimal feedback strength for noise suppression in autoregulatory gene networks," *Biophysical journal*, vol. 96, no. 10, pp. 4013–4023, 2009.

[114] P. S. Swain, "Efficient attenuation of stochasticity in gene expression through post-transcriptional control," *Journal of molecular biology*, vol. 344, no. 4, pp. 965–976, 2004.

[115] L.-G. Alberto, "Probability and random processes for electrical engineering," *Reading, Massachusetts: Addison-Wesely Publishing Company, Inc*, 1994.

[116] J. Choo, C. Lee, C. K. Reddy, and H. Park, "Utopian: User-driven topic modeling based on interactive nonnegative matrix factorization," *IEEE transactions on visualization and computer graphics*, vol. 19, no. 12, pp. 1992–2001, 2013.

[117] F. Buettner, K. N. Natarajan, F. P. Casale, V. Proserpio, A. Scialdone, *et al.*, "Computational analysis of cell-to-cell heterogeneity in single-cell RNA-sequencing data reveals hidden subpopulations of cells," *Nature Biotechnology*, vol. 33, pp. 155–160, Feb. 2015.

[118] A. A. Kolodziejczyk, J. K. Kim, J. C. H. Tsang, T. Ilicic, J. Henriksson, *et al.*, "Single Cell RNA-Sequencing of Pluripotent States Unlocks Modular Transcriptional Variation," *Cell Stem Cell*, vol. 17, pp. 471–485, Oct. 2015.

[119] A. A. Pollen, T. J. Nowakowski, J. Shuga, X. Wang, A. A. Leyrat, *et al.*, "Low-coverage single-cell mRNA sequencing reveals cellular heterogeneity and activated signaling pathways in developing cerebral cortex," *Nature Biotechnology*, vol. 32, pp. 1053–1058, Oct. 2014.

[120] G. Heimberg, R. Bhatnagar, H. El-Samad, and M. Thomson, "Low dimensionality in gene expression data enables the accurate extraction of transcriptional programs from shallow sequencing," *Cell systems*, vol. 2, no. 4, pp. 239–250, 2016.

[121] M. Schmidt, G. Fung, and R. Rosales, "Optimization methods for l1-regularization," *University of British Columbia, Technical Report TR-2009*, vol. 19, 2009.

[122] K. Basso, A. A. Margolin, G. Stolovitzky, U. Klein, R. Dalla-Favera, *et al.*, "Reverse engineering of regulatory networks in human b cells," *Nature genetics*, vol. 37, no. 4, pp. 382–390, 2005.

[123] Z. Zhu, T. Zheng, C. G. Lee, R. J. Homer, and J. A. Elias, "Tetracycline-controlled transcriptional regulation systems: advances and application in transgenic animal modeling," in *Seminars in cell & developmental biology*, vol. 13, pp. 121–128, Elsevier, 2002.

Appendix A

SUPPLEMENTARY MATERIALS FOR CHAPTER 3

A.1 Datasets and pre-processing

A.1.1 Dataset Pre-processing

All methods were run on the same subset of genes, selected by first binning the genes into five bins by mean expression value, and then taking the top 20% of genes in each bin by variance. For most datasets, the NMI did not change substantially for any of the methods when larger sets of genes were used.

A.1.2 Computational Experiment Details

tSNE used the implementation in scikit-learn 0.19. tSNE and ZIFA used 2 output dimensions. tSNE was run after first taking the truncated SVD with 50 dimensions on the log-transformed and column-normalized data, using the scikit-learn randomized TSVD. SIMLR, Magic, and UNCURL were all run using default settings. Large-scale SIMLR (Python implementation) was used for all datasets with more than 1000 cells, with 500 PCA components, on log-transformed data. UNCURL was run using 8 threads for all datasets with less than 1000 cells and 32 threads for all other datasets.

All computational experiments were done on a cluster with 64 dual-core AMD Opteron 6380 processors and 512GB of memory.

A.1.3 Dataset Descriptions

Dataset	Distribution	Cells	K	Description
10x_pooled_full	Poisson	73233	8	This dataset consists of 8 FACS sorted cell types from [30]: CD19+ b cells, CD14+ monocytes, CD34+, CD56+ NK, CD4+/CD45RO+ memory t, CD8+/CD45RA+ naive cytotoxic, CD4+/CD45RA+/CD25- naive t, and CD4+/CD25 regulatory t
10x_PBMC	Poisson	68579	10	68k PBMC dataset from [30], ground truth labels inferred by Spearman correlation with mean expression values of cell-sorted data
Zeisel	Poisson	3005	9	[32]
Zeisel_sub	Poisson	3005	9	[32], with UMI counts downsampled by 99%
10x_1.3M	Poisson	1306127	10*	`https://support.10xgenomics.com/single-cell-gene-expression/datasets/1.3.0/1M_neurons`
Tasic	Poisson	1629	49	[63] - cells from adult mouse visual cortex
mESC	LogNorm	182	3	[117], used as tests dataset in [50]
Kolod	LogNorm	704	3	[118], used as test dataset in [50]
Pollen	LogNorm	249	11	[119], used as test dataset in [50]
Usoskin	Poisson	622	4	[62], used as test dataset in [50]

All the 10x datasets were downloaded from https://support.10xgenomics.com/single-cell-gene-expression/datasets.

*The 10x_1.3M dataset does not have a provided K. We chose K=10 as a balance between runtime performance and cluster resolution.

A.2 Additional Algorithmic Details

A.2.1 State estimation

Algorithm 1 State estimation with the Probabilistic Convex Mixture Model

function ESTIMATE-STATE($X, k, maxiters, \epsilon$)

$W \leftarrow$ Init-W(X, k)

$M \leftarrow$ Init-M(X, W, k)

for $iter \leftarrow 1...maxiters$ **do**

Update W to minimize $-\log P(X \mid M, W)$, subject to W nonnegative

Update M to minimize $-\log P(X \mid M, W)$, subject to M nonnegative

if M and W changed less than ϵ this iteration **then**

return M, W normalized

end if

end for

return M, W normalized

end function

By default, Init-W(X, k) simply performs a k-means clustering on D reduced with truncated SVD, and assigns 0.75 to the highest cluster for each cell and 0.25 divided among the remaining clusters. Init-M(X, W, k) sets each column of M to be the mean of all cells assigned to that cluster.

W and M can also be initialized manually, or with the output of qualNorm or a different clustering/dimensionality reduction algorithm.

A.2.2 Sparse NoLips Algorithm

Algorithm 2 One round of NoLips optimization for W

function NoLips-Estimate-W(X, M, W, k, ϵ)

  for $c \leftarrow 1...numcells$ **do**

    $\lambda \leftarrow 1/(2\sum_j X[j, c])$

    $ci \leftarrow [0]^k$

    for $g \leftarrow 1...numgenes$ **do**

      $mw \leftarrow X[g, c]/(M[g, :] * W[:, c])$

      for $j \leftarrow 1...k$ **do**

        $ci[j] \leftarrow ci[j] + mwM[g, j]$

      end for

    end for

    for $j \leftarrow 1...k$ **do**

      $W \leftarrow \max(0, W[j, c]/(1 + \lambda W[j, c](\sum_l M[l, j] - ci[j])))$

    end for

  end for

  return W

end function

The optimization step for M proceeds identically, with X^T, W^T, and M^T as the parameters to the above algorithm.

A.2.3 Identifying and ranking cluster specific genes

In Figure 6 of the main text, we utilized the ranked list of cluster specific genes. While a commonly used approach used for cluster specific gene identification is 'one-vs-all' differential gene expression analysis, this method suffers from a few defects making it unsuitable for identification of over-expressed cell type specific genes from multiple clusters. The main

problem with the one-vs-all approach is that it treats all other clusters equally, when focusing on one cluster. This can lead to overlap among significant gene sets between different clusters. Moreover, differential gene expression analysis is also sampling model dependent hence would not work well for our pre-processed data. Hence, we devise the following test statistic to compare how over-expressed a gene is on average in one cluster compared to all other clusters:

$$CScore[G, C] = \frac{E[X_{j,i \in S_C}] + \epsilon}{\max_{k' \in [K], k' \neq C} E[X_{j,i \in S_{k'}}] + \epsilon}$$

Here G is the gene, C is the cluster of interest, S_k is a set of all cells belonging to cluster k and ϵ is a user defined small number ($\epsilon = 10^{-3}$ was used in this work). The test statistic measures the ratio of average expression between the cluster of interest and the highest among all other clusters. Hence, if $CScore \geq 1$, it represents a gene that has a higher expression in the cluster of interest than all other clusters. Since the statistic basically measures fold change, it can be used to rank the cluster specific genes.

A.2.4 Heatmap of cluster specific gene expression

Clusters are identified by performing $arg - max$ on the W matrix estimated by UNCURL. The top 10 genes of each cluster are identified by sorting on the basic of the $CScore$. We then group the genes and cells by cluster. Finally, within each cluster (say cluster i), the cells are sorted based on the values of the ith column of W. Each gene's expression is then normalized by it's maximum value. A heatmap is then plotted on this new matrix.

A.3 Synthetic data generation for lineage inference

A.3.1 Generating simulation data along a linear trajectory

To generate sampled data along a linear trajectory, we first select the extreme means M_1 and M_2. We then generate the true state matrix T such that the ith column is equal to:

$$T_i = rM_1 + (1 - r)M_2$$

Where $r \in [0, 1]$ is a variable that controls the mixing between the two means. The r's are chosen sequentially to obtain a complete lineage between the two simulated cell types. The observed data matrix (D) is then generated by the following way:

$$D_{i,j} = r, \text{where } r \sim Poiss(T_{i,j})$$

The normalization of the read count (to yield a standard read depth for each cell) is then performed using the uniform sampling method described in [120].

A.3.2 Generating simulation data along a branched trajectory

To generate sampled data along a branched trajectory, we first select the three extreme means M_1, M_2 and M_3. The centroid M_c is then found as follows:

$$M_c = \frac{M_1 + M_2 + M_3}{3}$$

Having generated this, we then proceed to generate linear trajectories between the pairs (M_1, M_c), (M_2, M_c) and (M_3, M_c). There are then combined into a true state matrix T. We then generate the observed data matrix in the same way as the linear case.

A.4 Additional results

A.4.1 Clustering NMI results

Dataset	tSNE + kmeans	SIMLR + kmeans	ZIFA + kmeans	Magic + tSNE + kmeans	UNCURL + argmax	UNCURL + tSNE + kmeans
10x_pooled_full	0.65	0.59	N/A	0.54	**0.83**	0.56
10x_PBMC	0.42	0.42	N/A	0.31	**0.57**	0.38
Zeisel	0.72	**0.77**	0.42	0.53	0.73	0.67
Zeisel_sub	0.55	0.58	0.43	0.37	0.64	0.61
Tasic	0.78	0.71	0.48	0.64	**0.82**	0.79
mESC	0.38	**0.81**	0.41	0.3	0.51	0.54
Kolod	0.99	0.99	0.55	**1.0**	0.95	**1.0**
Pollen	**0.95**	**0.95**	0.81	0.88	**0.95**	**0.95**
Usoskin	0.79	0.74	0.58	0.79	0.87	**0.98**

Dataset	kmeans	PCA + kmeans	Magic + kmeans	Magic + PCA + kmeans	UNCURL + kmeans	UNCURL + PCA + kmeans
10x_pooled_full	0.34	0.59	0.71	0.71	**0.85**	0.7
10x_PBMC	0.4	0.25	0.07	0.07	**0.57**	0.5
Zeisel	0.59	0.44	0.49	0.48	0.74	0.55
Zeisel_sub	0.43	0.43	0.42	0.37	**0.68**	0.5
Tasic	0.72	0.6	0.64	0.64	0.81	0.73
mESC	0.4	0.32	0.29	0.3	0.62	0.56
Kolod	0.92	0.65	**1.0**	**1.0**	0.97	0.97
Pollen	**0.95**	0.74	0.91	0.9	**0.95**	0.87
Usoskin	0.44	0.28	0.89	0.89	0.87	0.8

A.4.2 Comparison of semi-supervision on 10x pooled data

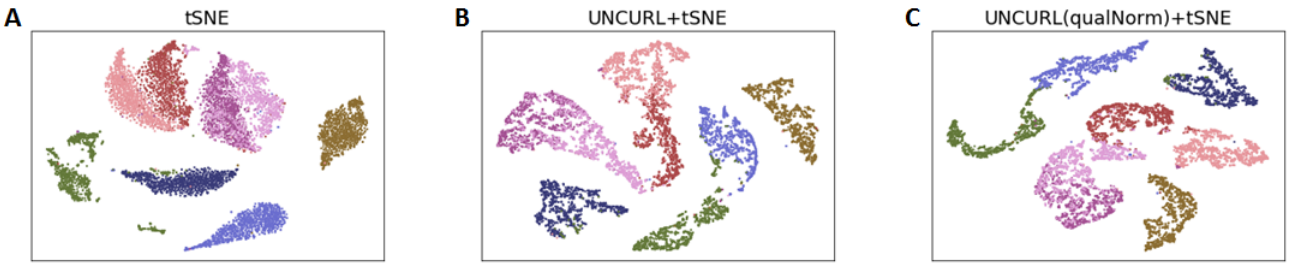


Figure A.1: Effect on tSNE based visualization for 10x pooled dataset using different initialization strategies. A) tSNE without pre-processing. B) Unsupervised UNCURL + tSNE. C) QualNorm semi-supervised UNCURL + tSNE.

A.4.3 Comparison of lineage inference on synthetic lineages

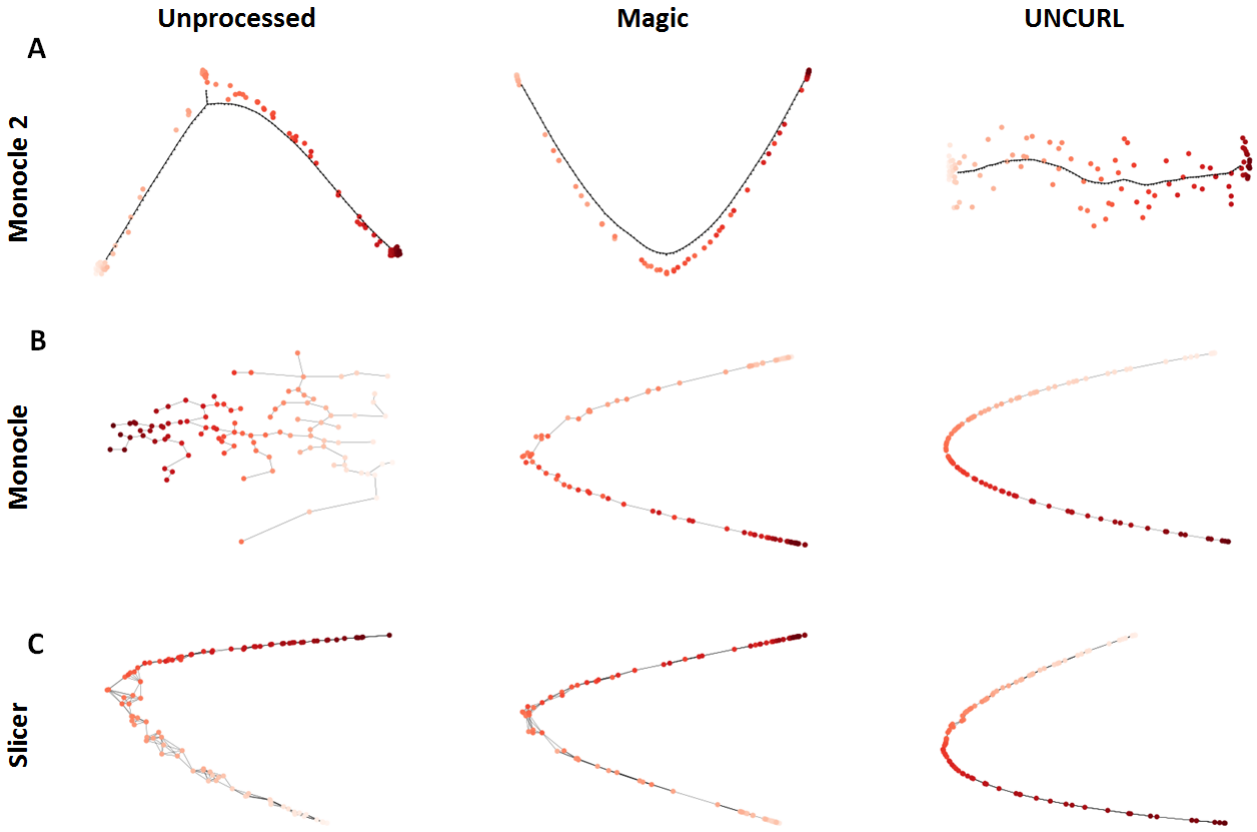


Figure A.2: Comparison of lineage estimation on a synthetic linear lineage containing 100 cells. Comparison of different algorithms and different three different pre-processing methods namely A) Unprocessed, B) Magic pre-processed, C) UNCURL pre-processed. Cells are colored by true progress along the lineage.

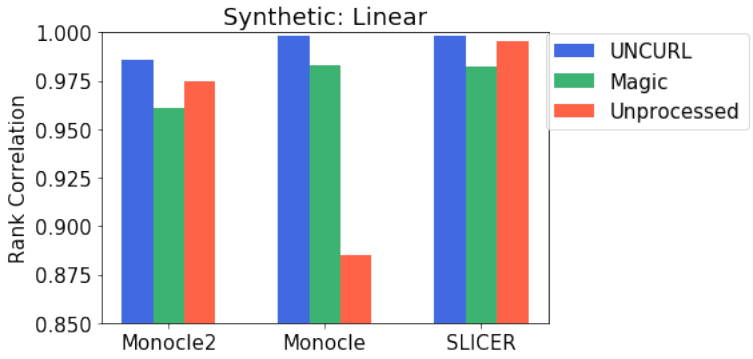


Figure A.3: Rank correlation (with true pseudotime) of the pseudotime estimated by different lineage estimation algorithms using different pre-processing methods.

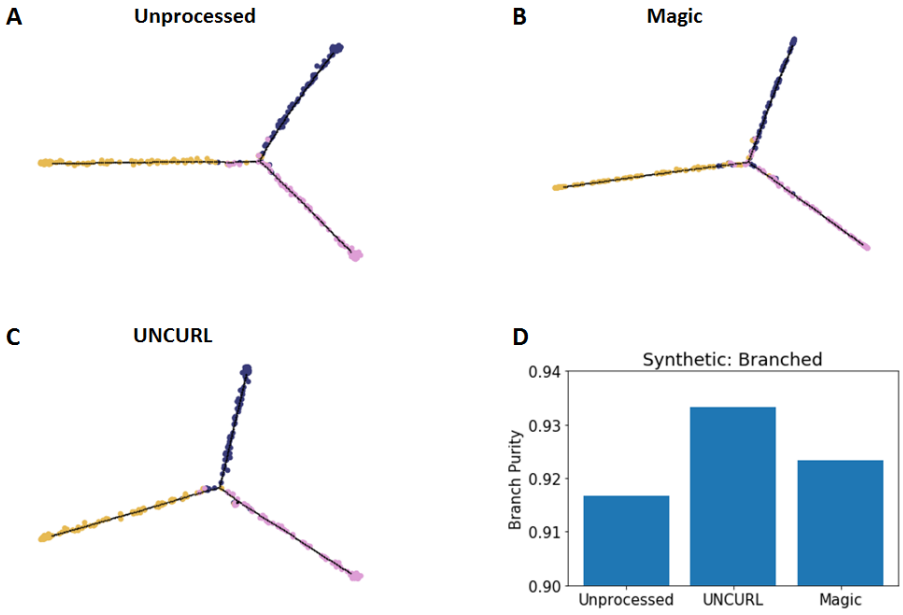


Figure A.4: Estimated lineage along a branched trajectory (containing 300 cells, 100 per branch) using Monocle2 after different pre-processing methods. A) Unprocessed, B) UNCURL pre-processed, C) Magic pre-processed. D) Comparison of branch purity measured using identified branches and actual branches.

A.4.4 Unsupervised UNCURL lineage for Hanchate et. al.

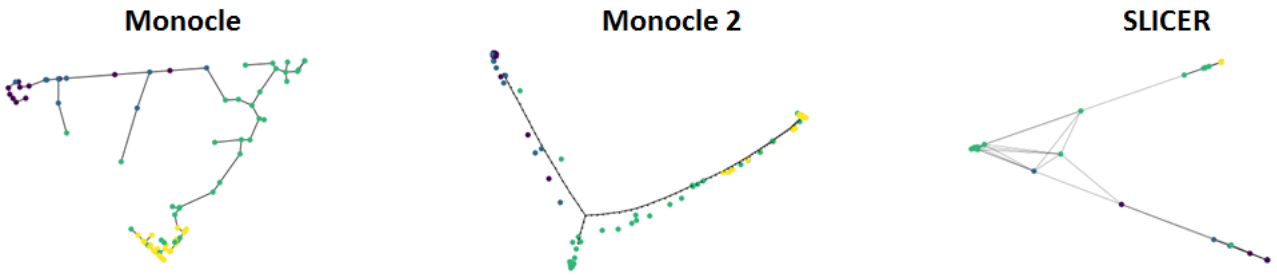


Figure A.5: Unsupervised lineage estimation after UNCURL pre-processing using different methods on the dataset from [4]

A.4.5 Timing comparison of various clustering methods

Dataset	Distribution	Cells	K	tSNE + kmeans	SIMLR	ZIFA + kmeans	Magic + tSNE + kmeans	UNCURL
10x_pooled	Poisson	73233	8	3615	2504	N/A	23188	**591**
10x_PBMC	Poisson	68579	10	4051	2733	N/A	9670	**490**
Zeisel	Poisson	3005	9	120	184	3040	93	**62**
Zeisel_sub	Poisson	3005	9	134	149	12778	96	**36.5**
10x_1.3M	Poisson	1.3 mil	10	87981.53	109887.2	N/A	N/A	**36869**
Tasic	Poisson	1629	49	**24**	27	2734	56	144
mESC	LogNorm	182	3	2	6.73	287	3	**0.5**
Kolod	LogNorm	704	3	9	84.75	2401	13	**2**
Pollen	LogNorm	249	11	2.7	10.5	964	6	**2.2**
Usoskin	Poisson	622	4	**8.3**	49.29	4603	21	19

A.4.6 Parallelizability of UNCURL

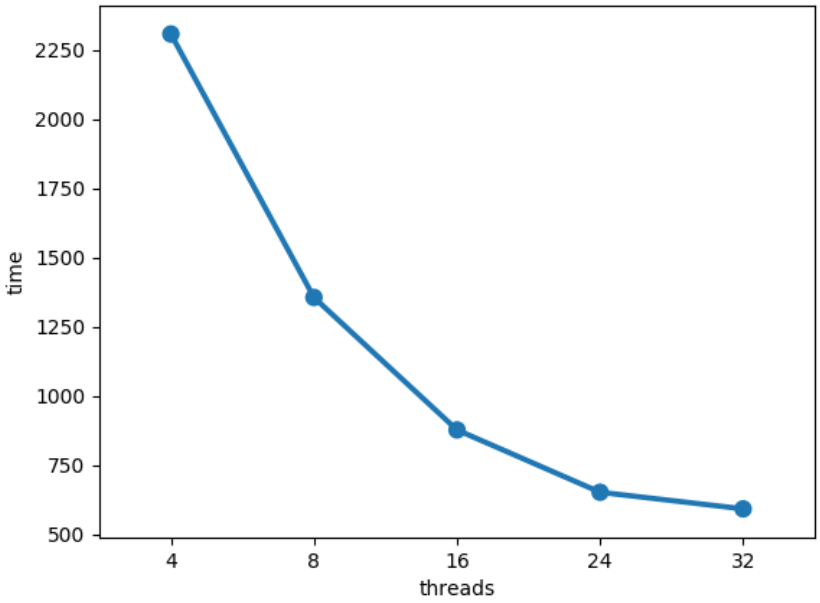


Figure A.6: Comparison of run times for UNCURL with different number of computing threads for the 10x-pooled dataset. It is seen that the run times show an almost linear decrease till about 24 cores after which the performance saturates.

A.4.7 Robustness to choice of K

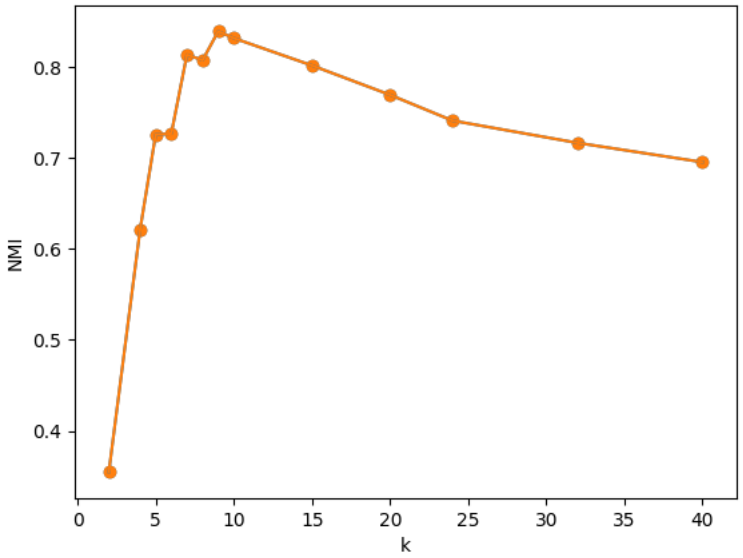


Figure A.7: Robustness of NMI to choice of K. For the 10x-pooled dataset, we compared the NMI of arg-max(W) for different choices of K. The true K for this dataset is 8. We see that while the NMI of UNCURL peaks around the true value of K, increasing it further leads to only a slight loss in precision. Thereby showing that UNCURL is quite robust to the choice of K.

A.4.8 Cluster specific gene heatmap after Magic pre-processing for 20k subset 10x 1.3 million dataset

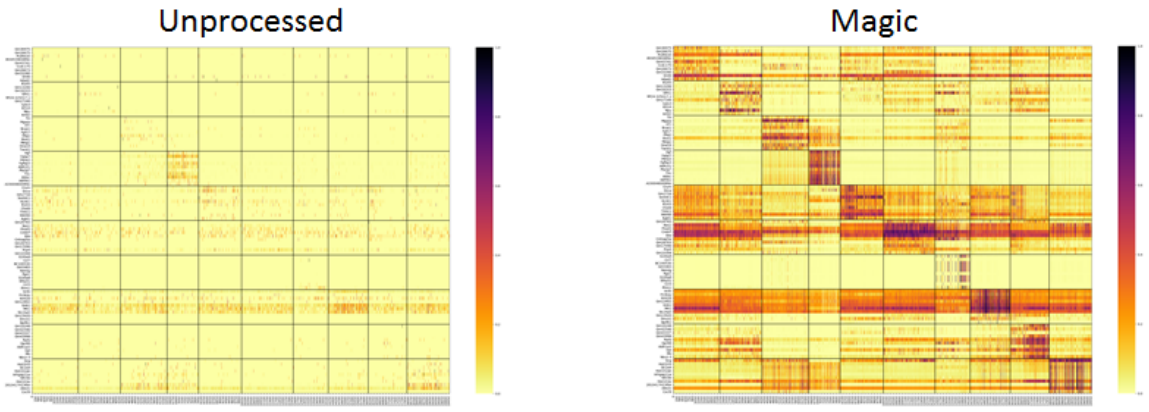


Figure A.8: Clustered heatmaps showing the top cluster specific genes identified by Magic before and after pre-processing. Cells sorted by decreasing W for each cluster.

A.4.9 Cluster specific gene plots for 10x 1.3 million dataset

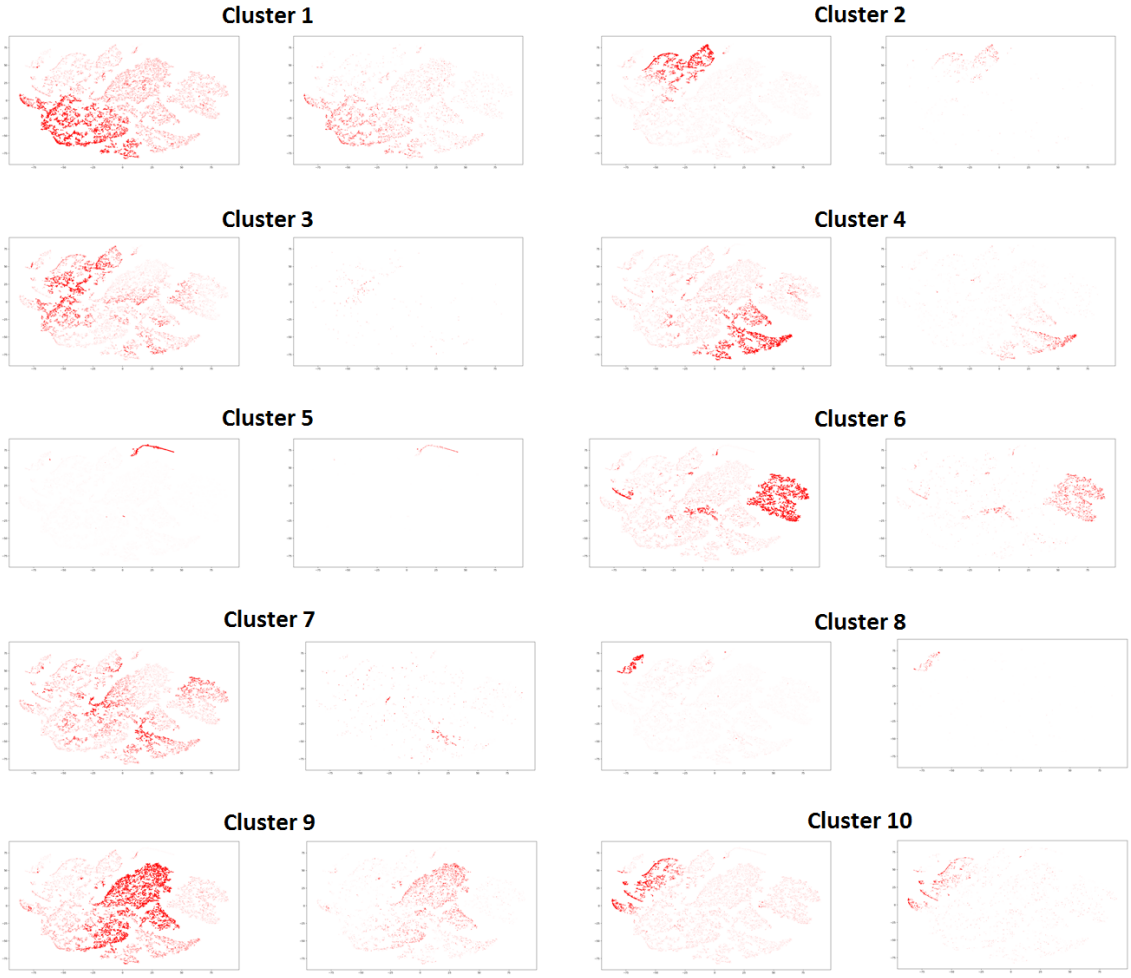


Figure A.9: Average expression (with and without pre-processing) of the top cluster specific genes overlaid on the UNCURL processed tSNE plot.

A.4.10 Top cluster specific genes for 10x 1.3 million dataset

Cluster number	Genes
1	Dnah3, RP23-32A8.6, 1700013G24Rik, Gm11973, 1700101O22Rik, Gm20069, Gm11817, 1700042O10Rik, Tex21, Gm28856
2	Kcnk18, Paqr6, Rsph4a, 2210011K15Rik, Opn5, Gm42956, Tnn, Map3k19, Tctex1d1, Wnt3a
3	Krt18, Synpo2l, Gm43419, Gm13583, Prdm13, Hoxc6, Odf3l1, Ttll10, Pebp4, Gm26516
4	Myh2, Gm14637, Slc6a5, Dsg1a, Hormad1, Gys2, RP23-274K7.1, Rxfp3, Fam163a, Tbata
5	Gm5483, BC100530, Stfa2l1, S100a9, Ngp, Wfdc21, Retnlg, Gm5416, Epb42, Stfa2
6	Cntnap3, Isl1, Gm43998, Cacng3, Htr3a, Nxph2, Gm16315, Erbb4, Cntnap5c, Neb
7	Glycam1, Clca3a2, Ntn5, RP23-344G21.5, Trhr2, Gm8239, Cnn1, Slc9c1, Gm15294, Gm31728
8	Ccl7, Bcl3, Spint1, Il21r, Il1a, Ms4a7, Ptgs2, Tecrl, Srpx2, Gm5127
9	Crhr2, 1700080N15Rik, C8b, 4921521D15Rik, Baat, Arhgap33os, Macrod2os1, Smco1, Ly6g6f, Zdhhc23
10	Agbl1, Gm6116, Ttc39d, Dmrtc2, Gm20711, Btla, Cxcl9, 4933427J07Rik, Tbx5, Fbxw19

Appendix B

SUPPLEMENTARY MATERIALS FOR CHAPTER 4

B.1 Comparing network structure inference by PIPER and GGM for synthetic data

Although it has been demonstrated previously in [77] that Local Poisson graphical models (LPGMs) outperform GGMs on simple synthetic Poisson datasets, we compare the two techniques on networks of different sizes and sample sizes to identify if LPGMs are indeed the best candidate for us. This is an important step because the accuracy of the inferred structure inference is an important step in identification of upstream genes.

Poisson data was generated using the method described in [77] for generating data with a Poisson prior for networks having different number of nodes (n = 25, 50, 100) with the sample size held constant at 300. Similarly we generate data for different sample sizes (p = 50, 100, 250, 500) with the number of genes held constant at 25. In both cases we can see in Figure B.1 that the networks estimated by PIPER outperform GGMs in all graph and sample sizes. The L1 regularized optimization problem for network estimation was implemented using the MATLAB L1-optimization tool developed in [121], while the GGM inference was done using publicly available MATLAB codes.

B.2 List of predicted key regulators for Zeisel dataset

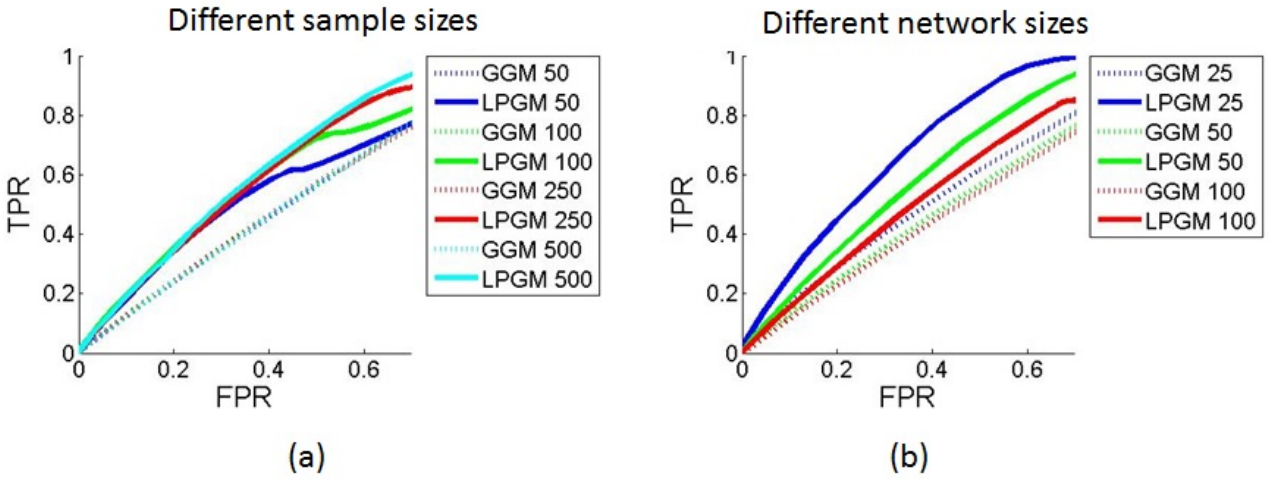


Figure B.1: Comparison of PIPER and GGM for synthetic count-valued data. a-b) PIPER outperforms Gaussian Graphical Models for synthetic datasets generated with Poisson prior for different sample sizes and graph sizes.

Table B.1: Prediced key regulators on Zeisel dataset

Prior evidence	**Genes**
Yes	Acsl3,Cers2,Ckb,Cpd,Dbi,Fkbp1a,Gnas,Grb14,Gsn,Jph4,Klhl2,Lgi3, Myrf, Nfasc,Olig1,Omg,Rplp1,Tcf4,Mbp
No	1500004A13Rik,1810037I17Rik,2810468N07Rik,Acot7,Arpp21,Atp2a2, Cldn11,Cox7b,Cspg5,Ctps,Dbndd2,Eif4a2,Epn2,Ermn,Fah,Fnbp1,Fth1, Gamt,Gm13363,Golga7,Hmgcs1,Hsd17b7,Kndc1,Mag,Mal,Nme1,Opalin, Osbpl1a,P4hb,Pdlim2,Plekha1,Plp1,Ptgds,Qdpr,Rnf7,Slc48a1,Sptbn1, Strbp,Sv2a,Tmem141,Ugt8a,Usmg5,Vmp1,Ywhaq

Table B.2: Detailed description of known genes with prior evidence

Genes	**Comment**
Acsl3	Regulated by mTOR pathway during olig differentiation
Cers2	Cers2 lacking mice have myelin defects
Ckb	Promotes astrocyte differentiation
Cpd	Plays role in rats
Dbi	Key regulators in astrocytes
Fkbp1a	Homolog known to play important role in differentiation
Gnas	Important gene in differentiation of other types
Grb14	Directly downstream Tcf4. Known transcription factor
Gsn	Early marker of oligodendrocyte
Jph4	Part of important family of regulators
Klhl2	Shown to play role in human oligodendrocyte development
Lgi3	Family of proteins playing role in neuronal development
Myrf	Key regulators in oligodendrocyte differentiation
Nfasc	Key regulator of neuronal progenitor differentiaiton in rats
Olig1	Key regulators in oligodendrocyte differentiation
Omg	Key regulator of neurogenesis
Rplp1	Shown to play role in neuronal development
Tcf4	Shown to play role in key neuronal development
Mbp	Differentiation regulator for oligodendrocytes

Appendix C

SUPPLEMENTARY MATERIALS FOR CHAPTER 5

C.1 Mathematical Model of Single Gene Feedforward Loop

A miRNA based sgFFL system in general has a gene with an intronic miRNA which targets the 3'UTR region of the same gene. The data used for parameter estimation of a model sgFFL system in this work is obtained from the the experimental work done in [3]. The system comprises of a CMV promoter with Tet recognition sites and a mCherry gene with intronic miRNA 124a. The Tet controlled CMV promoter is activated in the presence of the inducer Doxicycline. The 3'UTR of the mCherry has one or more target sites corresponding to mir-124a. The open loop cell line lacks the target sites in the 3'UTR of mCherry making it free from interactions with mir-124a. Figure C.1.

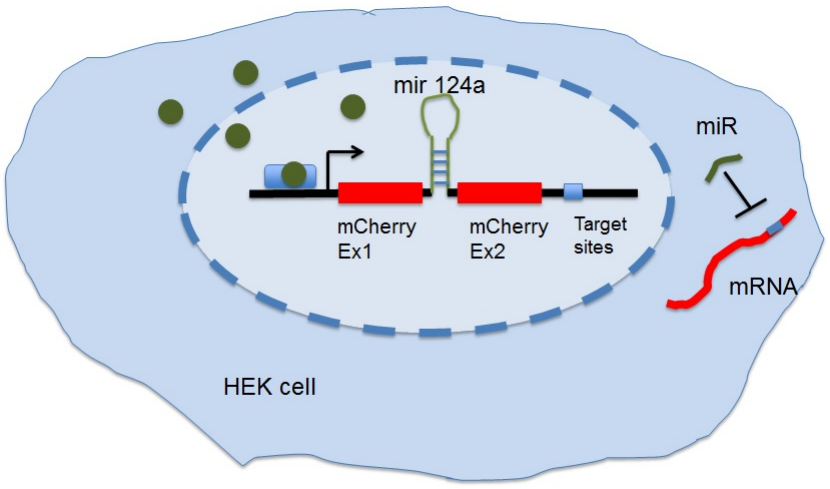
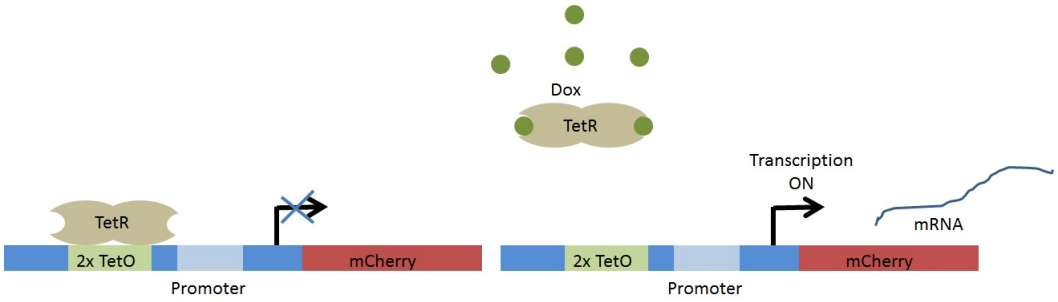


Figure C.1: Construct of the sgFFL system and the effect of Doxicyline induction

Based on the construct description in the previous paragraph, a mathematical model for



(a) (b)

Figure C.2: Doxycycline induction of Hybrid Tet system. a) In the absence of Doxycycline, the TetR is bound to the TetO sites and it blocks transcription b) In the presence of Doxycycline the TetR is displaced leading to transcription

the system can be arrived at using some standard modeling assumptions.

C.1.1 Doxycycline Induction

The system studied in this work used a Hybrid-Tet system which is activated by Doxycyline. This is a very well characterized system and has been studied by various groups in the past [122, 123]. The primary mechanism of operation for this system is demonstrated in Figure C.2. In the absence of the Doxycycline, the TetR binds to the Tet operator site and there is no transcription. When Doxycycline in introduced, it binds to the TetR dimer and cause a change in its conformation which prevents it from binding to the 2xTetO site and inititate transcription of mCherry. The detailed mechanism can be summarized as

$$nI + T_{on} \underset{k_r}{\overset{k_f}{\rightleftharpoons}} T_{off}$$

$$mT_{on} + G_{on} \underset{\lambda^+}{\overset{\lambda^-}{\rightleftharpoons}} G_{off}$$

Here, G_{off} is the gene with just the Tet R bound to it, while G_{on} is the gene with the activated gene with Doxicycline bound to the TetR, T_{off} is the inactive Tet R, T_{on} is the active Tet R and I is Doxicycline. The Tet R forms a dimer molecule and has binding sites for

two Doxycycline molecules. However, it has been seen that even binding of one Doxycyline molecule can induce a change in conformation, causing it to bind to the Tet Operator site. In our simple system we use a 2xTetO site, so either one or two Tet R's can bind to the operator region. Assuming the total number of genes in a cell are constant, the chemical kinetics for G_{on} can be written as

$$\frac{dg}{dt} = \lambda^{+}(g_T - g) - \lambda^{-} g T_{on}^{m} \tag{C.1}$$

Where, g is the number of active genes at any time, g_T is the total number of genes (1 in our case) and T_{on} is the number of active Tet R molecules inside the cell. It has been shown in [122] that the steady state value of T_{on} is

$$T_{on}^{\star} = \frac{T_{tot}}{1 + k_e I^n}$$

Here, I is the number of inducer molecules in a cell and $k_e = \frac{k_f}{k_r}$. We ignore the degradation of doxicycline and the dynamics of TetR for the purposes of this work. However, for practical ranges of induction levels $k_e I^n >> 1$, so $T_{on}^{\star}$ can be approximated to $T_{on}^{\star} = \frac{T_{tot}}{k_e I^n}$.

C.1.2 Micro RNA based regulation

As stated in previous sections, miRNA based regulation proceeds primarily through two groups of steps namely miRNA mediated degradation and translational inhibition. There are many models for miRNA based degradation that have been proposed as in [105–107]. Some of the newer work [108] have developed more detailed models. For the purposes of this work we rely on the existing models and make minor adjustments based on the available experimental data. We assume that the mRNA (M) and microRNA (S) form a complex C irreversibly which then degrades to give back the microRNA irreversably.

$$M + S \xrightarrow{\gamma_s} C \xrightarrow{k_c} S$$

This leads to the following model

$$\frac{dm}{dt} = \alpha_m g - \beta_m m - \gamma_s ms \tag{C.2}$$
$$\frac{ds}{dt} = \alpha_s g - \beta_s s - \gamma_s ms + k_c c \tag{C.3}$$
$$\frac{dc}{dt} = \gamma_s ms - k_c c \tag{C.4}$$
$$\frac{dp}{dt} = \alpha_p (m + k_1 c) - \beta_p p \tag{C.5}$$

Here, m is the no. of mRNA transcripts, s is the number of mature miRNAs, p is the number of proteins, c is the miRNA-mRNA complex, k_1 is a fractional number between 0 and 1 signifying how much slower the translation is for the miRNA-mRNA complex in comparison to just the mRNA. α_m, α_s, α_p are the production rates of mRNA, miRNA and protein while β_m, β_s, β_p are their corresponding natural degradation rates. γ_s is the miRNA mediated degradation rate.

We then assume that the mRNA-miRNA complex formation is fast and so is under quasi steady state. Hence, we get $c = \frac{\gamma_s ms}{k_c} = k_2 ms$. We base this on the experimental observations that the miRNA plot strongly resembles an exponential curve. We also introduce the concept of translational inhibition into the model. For the purposes of simplicity we assume that the effect of translational inhibition is to slow down the rate of translation. Upon applying the quasi steady state assumption and simplifying, this leads to the following model

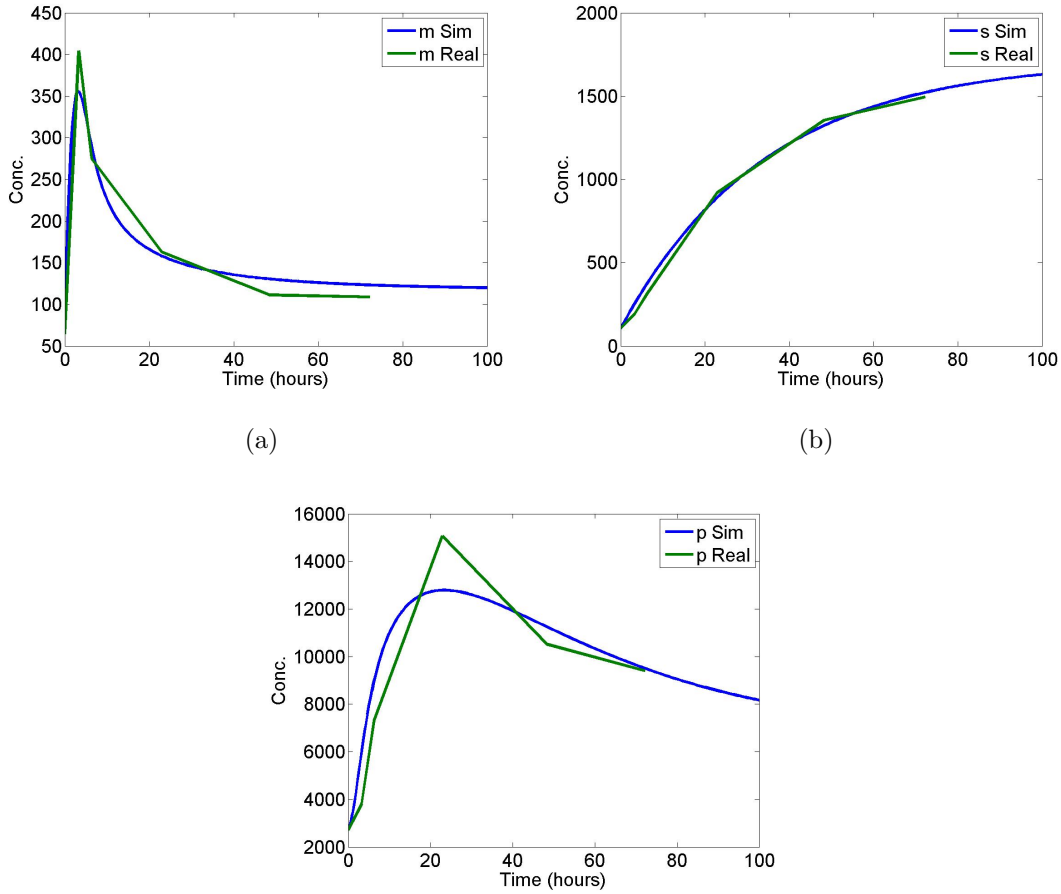
$$\frac{dm}{dt} = \alpha_m g - \beta_m m - \gamma_s ms \tag{C.6}$$
$$\frac{ds}{dt} = \alpha_s g - \beta_s s \tag{C.7}$$
$$\frac{dp}{dt} = \alpha_p m(1 + k_1 k_2 s) - \beta_p p \tag{C.8}$$

For the open loop system the values of γ_s, k_1 and k_2 are all zero.

C.2 Parameter Estimation

With the previously stated model, we performed parameter estimation with time course dataset available (for both sgFFL and open loop cell lines) for different induction levels. The data available was FISH (Fluorescence In Situ Hybridization) data for mRNA reads for each cell, qPCR data for miRNA and Flow Cytometry data for mCherry expression levels. This data has been taken from [3]. The parameter values were allowed to vary between realistic parameter ranges. For the purposes of parameter estimation we take $\alpha_M = \alpha_m g^{ss}$, $\alpha_S = \alpha_s g^{ss}$ where g^{ss} is the steady state value of g. Assuming that the switching of states of the gene occurs at a much faster rate, this would be a reasonable assumption. Figure C.3 shows the fit obtained by the estimated parameters at an induction level of 1 $\mu g/ml$. Since α_M and α_S are related by virtue of being transcribed together, we assume their ratio to be constant across all induction levels. Thus by letting all other parameters remain constant and just varying α_M, we obtain plot a of Figure C.4. Two different hill functions of the form $\frac{aI^{nm}}{b+I^{nm}}$ and $\frac{aI}{b+I}$ were fitted using the matlab curve fitting toolbox to the estimated α_M vs Dox data. It was observed that the second hill function (with n =1) led to better fits indicating lack of cooperativity in the induction. The parameters a and b here stand for $a = \alpha_m$ and $b = \frac{\lambda^- T_{tot}^m}{\lambda^+ k_e^m}$ respectively. Here the induction level is also adjusted to a per cell value based on the approximate volume of a HEK-293 cell. A similar analysis is performed on the open loop cell line as well.

The different identified parameter values are seen in Table C.1.



(a) (b)

(c)

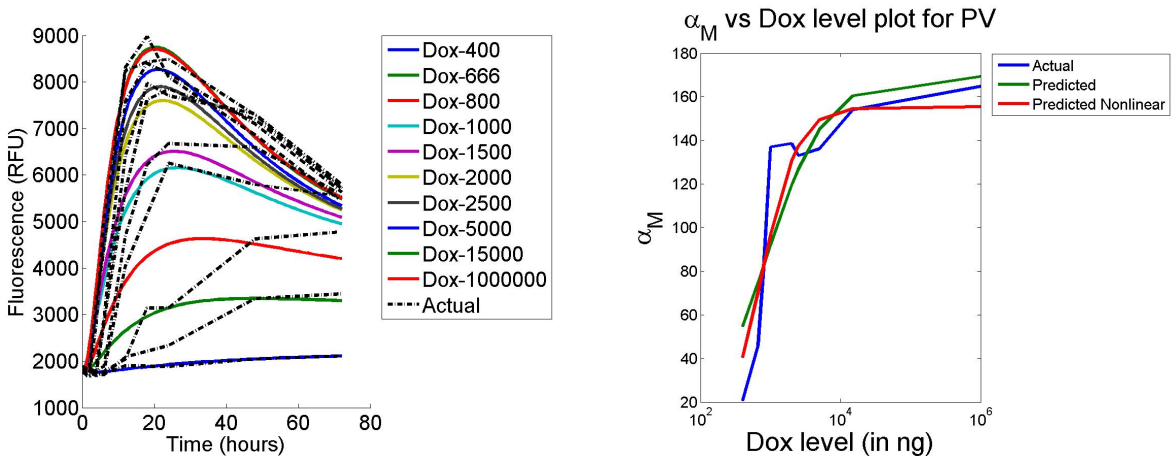
Figure C.3: Plots obtained from parameter estimation. a-c) Fits at 1 $\mu g/ml$ concentration of doxicycline using identified values of parameters for a) mRNA b) microRNA and c) protein.

C.3 Closed form calculation of Intrinsic Noise

As explained in chapter 5, the Intrinsic noise calculation requires the solution of the equation

$$\frac{d\sigma}{dt} = A\sigma + \sigma A^T + B \tag{C.9}$$

where σ is the matrix of covariances, A is the Jacobian matrix for the dynamics of averages and B is diffusion matrix that depends on the size of the random events.



(a) (b)

Figure C.4: a) Protein level fits at different induction concentrations. b) Comparison of different hill function models the estimated α_M vs Dox curve

C.3.1 For mRNA level

For calculating the intrinsic noise at mRNA level,the Jacobian and diffusion matrices were calculated to be

$$A = \begin{bmatrix} -k_1 & 0 & 0 & 0 \\ 0 & -k_2 & 0 & 0 \\ \alpha_m & 0 & -\Gamma_m & -\gamma_s m^\star \\ 0 & \alpha_s & 0 & -\beta_s \end{bmatrix} \tag{C.10}$$

$$B = \begin{bmatrix} \lambda_1^+(1-g_1^\star)+\lambda_1^- g_1^\star & 0 & 0 & 0 \\ 0 & \lambda_2^+(1-g_2^\star)+\lambda_2^- g_2^\star & 0 & 0 \\ 0 & 0 & \alpha_m g_1^\star + \Gamma_m m^\star & 0 \\ 0 & 0 & 0 & \alpha_s g_1^\star + \beta_s s^\star \end{bmatrix} \tag{C.11}$$

These were then used to calculate the individual covariances of the different species. Of particular interest are C_M, C_{MS}, C_{MG_1} and C_{MG_2}, which we later use to calculate the steady state autocovariance function at the mRNA level. The simplified expressions were calculated

Parameter	**Value**
α_m	193.1286
β_m	0.25
α_s	46.9534
β_s	0.026366
γ_s	0.0010476
α_p	2.9081
β_p	0.061122
k_1	0.0093367
k_2	1e-05
a_{lin}	169.4
b_{lin}	5.036e+04
a_{NL}	157
b_{NL}	6.997e+07
n	1.685

Table C.1: Table of estimated parameter values

to be

$$C_M = \eta_m^2 m^{\star^2} \tag{C.12}$$

$$C_{MG_1} = m^\star \eta_{g_1}^2 \frac{\Gamma_m}{\Gamma_m + K_1} - \frac{1}{g_2^\star} \frac{\beta_s}{\beta_s + K_1} \frac{\gamma_s \sigma_{g_1 g_2}^2 m^\star s^\star}{\Gamma_m + K_1} \tag{C.13}$$

$$C_{MG_2} = \frac{1}{g_1^\star} \frac{\Gamma_m}{\Gamma_m + K_2} \sigma_{g_1 g_2}^2 m^\star - \frac{\beta_s}{\beta_s + K_2} \frac{\gamma_s \eta_{g_2}^2 m^\star s^\star}{\Gamma_m + K_2} \tag{C.14}$$

$$C_{MS} = \frac{m^\star s^\star \sigma_{g_1 g_2}^2}{g_1^\star g_2^\star} \frac{\Gamma_m}{\Gamma_m + K_2} (1 + \frac{K}{\beta_s + \Gamma_m}) - \frac{\gamma_s m^\star s^\star}{\beta_s + \Gamma_m} (1 + \eta_{g_2}^2 s^\star \frac{\beta_s}{\beta_s + K_2} (1 + \frac{\beta_s}{\Gamma_m + K_2})) \tag{C.15}$$

where the expression for η_m^2 can be found in chapter 5.

C.3.2 For protein level

The protein level calculation assumed that that the effect of correlation between the two genes can be captured by the correlation between the mRNA and the micro RNA. With this assumption, the Jacobian and diffusion matrices were obtained as

$$A = \begin{bmatrix} -\Gamma_m & -\gamma_s m^\star & \alpha_p \\ 0 & -\beta_s & 0 \\ A_p & \alpha_p K m^\star & -\beta_p \end{bmatrix} \tag{C.16}$$

$$B = \begin{bmatrix} \alpha_m + \Gamma_m m^\star & 0 & 0 \\ 0 & \alpha_s + \beta_s s^\star & 0 \\ 0 & 0 & A_p m^\star + \beta_p p \end{bmatrix} \tag{C.17}$$

where all the terms are the same as defined in chapter 5.

C.4 Intrinsic noise plot parameters

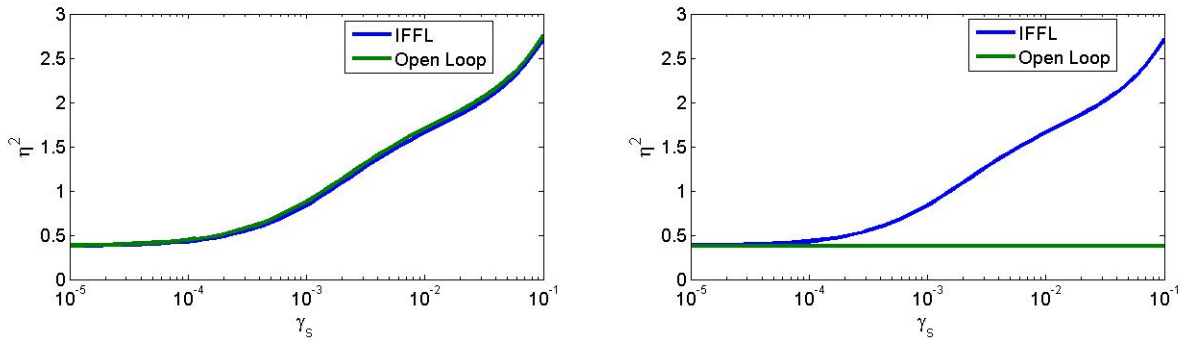
The parameter values were randomly chosen between an acceptable parameter regime. The parameter values used for simulation can be seen in Table C.2.

C.5 Intrinsic Noise plots with identified parameters

Figure C.5 shows the variance of mRNA noise for the identified parameters at identical and different means. Figure C.6 shows the filtering property demonstrated on the identified parameter set at low induction level.

	Parameter
α_s	77.3893
α_s	3556.63
β_m	0.10949
β_s	0.091792
$g_1^\star$	0.15315
$g_2^\star$	0.37144
k_1	0.17534
k_2	0.39569
γ_s	0.0099696
$\sigma^2_{g_1 g_2}$	0.18781

Table C.2: Table of random parameter values used for intrinsic noise simulations



(a) (b)

Figure C.5: Coefficient of Variation at the mRNA level plotted against different values of micro RNA - mRNA binding strength. a) Means of IFFL and open loop systems are identical b) Means of IFFL and open loop systems are different

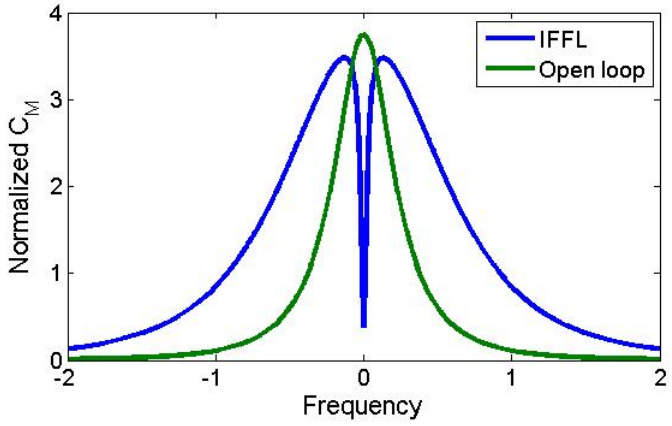


Figure C.6: Spectrum of mRNA steady state autocovariance function

C.6 Condition for buffering in IFFLs

C.6.1 Buffering in sgFFLs at mRNA level

Theorem 1:- *The mRNA count of the sgFFL system represented by the equations C.6, C.7, C.8 has lower sensitivity to changes in induction level than the open loop system for all parameter values.*

Proof :- First we observe that the change in induction level is analogous to change in α_M. Thus calculating the sensitivity with respect to α_M is equivalent to calculating the sensitivity with respect to the induction level changes. As shown before in [110] we calculate the steady state mRNA level for the open loop system to be

$$m^{\star}_{open} = \frac{\alpha_M}{\beta_m} \tag{C.18}$$

Notice that the steady state value of the active genes is captured inside the α_M term which is consistent for both open loop as well as sgFFL cell lines. Now as explained before, change in induction level is only likely to change α_M, so we are just concerned with the sensitivity

with respect to it. The sensitivity analysis reveals :-

$$S(m^{\star}_{open}, \alpha_M) = \frac{\alpha_M}{m^{\star}_{open}} |\frac{\partial m^{\star}_{open}}{\partial \alpha_M}| = 1 \tag{C.19}$$

For the sgFFL system let us recollect that $\frac{\alpha_S}{\alpha_M} = p$, where p is a constant. It is seen that the steady state mRNA level for the IFFL system is given by :-

$$m^{\star}_{sgFFL} = \frac{\alpha_M}{\beta_m + \gamma_s s^{\star}} = \frac{\alpha_M}{\beta_m + \gamma_s \frac{p\alpha_M}{\beta_s}} \tag{C.20}$$

Now the sensitivity analysis with respect to α_M reveals :-

$$S(m^{\star}_{sgFFL}, \alpha_M) = \frac{\alpha_M}{m^{\star}_{sgFFL}} |\frac{\partial m^{\star}_{sgFFL}}{\partial \alpha_M}| = \frac{1}{1 + p\frac{\alpha_M}{\beta_s \beta_m}} \tag{C.21}$$

Now since p, α_M, β_m and β_s are all positive quantities, this expression is always less than $S(m^{\star}_{sgFFL}, \alpha_M)$.

C.6.2 Buffering in sgFFLs at protein level

Theorem 2:- *The protein count of the sgFFL system represented by the equations C.6, C.7, C.8 has lower sensitivity to changes in induction level than the open loop system if* $\gamma_s > k_1 k_2 \beta_m$.

Proof :- The steady state protein count for the open loop system is given by

$$p^{\star}_{open} = \frac{\alpha_M \alpha_p}{\beta_m \beta_p} \tag{C.22}$$

Now the sensitivity analysis of equation C.22 with respect to α_M reveals :-

$$S(p^{\star}_{open}, \alpha_M) = \frac{\alpha_M}{p^{\star}_{open}} |\frac{\partial p^{\star}_{open}}{\partial \alpha_M}| = 1 \tag{C.23}$$

For the purposes of this proof let us take $k = k_1 k_2$. Then the steady state protein count of the sgFFL system is given by

$$p^{\star}_{sgFFL} = \frac{\alpha_M \alpha_p (1 + k s^{\star})}{\beta_p (\beta_m + \gamma_s s^{\star})} \tag{C.24}$$

Now on performing sensitivity analysis on equation C.24 we get

$$S(p^{\star}_{sgFFL}, \alpha_M) = \frac{\alpha_M}{p^{\star}_{sgFFL}} \left| \frac{\partial p^{\star}_{sgFFL}}{\partial \alpha_M} \right| = \frac{k\gamma_s s^{\star 2} + \beta_m + 2\beta_m k s^{\star}}{k\gamma_s s^{\star 2} + \beta_m + \beta_m k s^{\star} + \gamma_s s^{\star}} \tag{C.25}$$

It can be seen in equation C.25 that the expression evaluates to less than 1 ($S(p^{\star}_{open}, \alpha_M)$) only when $\gamma_s > k_1 k_2 \beta_m$. Now recollecting that k_1 is the fraction of translational coefficient of the miRNA-mRNA complex to the unbound mRNA, it follows that the lower this fraction, the better is the buffering. This seems to indicate that the stronger the translational inhibition, more is the buffering obtained from the sgFFL system. From our identified parameter values, $k_1 k_2 << \gamma_s$ for all data sets, so this seems to be true in general.