Inferring cell-specific gene regulatory networks from single cell gene expression data

Ziqi Zhang[1] Jongseok Han[1] Le Song[2,3] Xiuwei Zhang[1*]

[1]Dept of Computational Science and Engineering,
Georgia Institute of Technology, Atlanta, GA 30332
[2]Biomap, [3] MBZUAI

Abstract

Single cell gene expression datasets have been used to uncover differences between single cells, leading to discoveries of new cell types and cell identities, which are usually defined by the transcriptome profiles of cells. Biological networks, in particular, gene regulatory networks (GRNs), can be viewed as another feature of the cells, that contributes to the uniqueness of each single cell. However, methods that reconstruct cell-specific GRNs are still missing.

We propose CeSpGRN (**Ce**ll **Sp**ecific **GRN**), which infers cell-specific GRNs from single cell gene expression data. CeSpGRN uses a Gaussian weighted kernel which allows the GRN of a given cell to be learned from the gene expression profile of this cell and cells that are upstream and downstream of this cell in the developmental process. CeSpGRN can be applied to infer cell-specific GRNs in cell populations of any trajectory or cluster structure, and it does not require time information of cells as additional input. We compared the performance of CeSpGRN and baseline methods on simulated datasets obtained under various settings. CeSpGRN showed superior performance in reconstructing the GRN for each cell, as well as in detecting the regulatory interactions that differ between cells. We also applied CeSpGRN to real datasets including THP-1 human myeloid monocytic leukemia cells and mouse embryonic stem cells, where CeSpGRN suggested a set of interactions between genes that rewire during the differentiation process in these cells. CeSpGRN is available at `https://github.com/PeterZZQ/CeSpGRN`.

*Corresponding Author. Email: xiuwei.zhang@gatech.edu

1 Introduction

Gene regulatory networks (GRNs) represent how genes regulate each other during biological processes. Inferring GRNs from gene expression data has been a long-standing and challenging problem. Single cell gene expression data (which can be obtained from various technologies including single cell RNA-sequencing (scRNA-Seq), multiplex PCR and image-based technologies like SeqFISH and MERFISH) data have been used to infer GRNs given its large number of samples, where each cell is used as a sample [1, 2, 3, 4, 5, 6, 7]. Although a number of methods have been developed, these methods aim to learn one GRN from the gene expression data of a number of single cells. However, it has been reported that GRNs are highly dynamic and the topology changes over a temporal process [8, 9, 10, 11, 12, 13]. Methods were developed to learn time-varying GRNs at specific time points using microarray or RNA-seq data [10, 9, 12, 13]. However, since the number of time points in these datasets is usually low (at the scale of dozens) and the data at each time point measures bulk-level gene expression of a set of cells, certain network rewiring events may not be detected.

When analyzing scRNA-seq data of cells in a dynamic process, each cell is considered to belong to a unique stage in the dynamic process, and its gene expression profile represents the cell state of the corresponding stage. Just like that each individual cell has unique transcriptome or proteome profiles, different cells can have different GRNs. In this paper, we propose CeSpGRN, which infers a GRN for every single cell. Obtaining cell-specific GRNs is of great significance as it allows researchers to investigate the changes in GRNs during a dynamic process or cross a spatial landscape.

Previously, Dai *et al* developed CSN (Cell Specific Network), which attempted to calculate pairwise gene-gene association in single cells using a statistical measurement [14]. In GRN inference all genes are considered at the same time thus the problem is more challenging than pairwise gene-gene association analysis. Typical GRN inference methods require a sufficient number of samples which makes inferring a cell-specific GRN for each cell seemingly impossible. We are inspired by an existing method, KELLER, which infers time-varying GRNs using a weighted kernel that allows one to “borrow” information from nearby time points [10]. In CeSpGRN, we assume that the GRNs of cells that undergo a developmental or differentiation process change smoothly along the cell trajectory. This assumption enables “information sharing” across different cells through a weighted kernel. However, there are substantial differences between CeSpGRN and KELLER in their methods: (1) KELLER uses Markov Random Field (MRF) to model a GRN, which takes binary gene expression data. In order to avoid the information loss that occurred during the binarization step and to better accommodate for the distribution of single cell gene expression data, we use a Gaussian Copula Graphical Model (GCGM) [15, 16] in CeSpGRN; (2) KELLER requires time information for every sample and constructs the kernel using the 1-dimensional time information for each sample. CeSpGRN does not require known time information of single cells, and constructs the kernel using high-dimensional gene expression profiles of the cells; (3) As KELLER designs the kernel using cells’ time information, it cannot distinguish cells which are at the same developmental time but with different fates. That is, a kernel using 1-dimensional time information can only work with datasets where cells form a linear trajectory. In contrast, the high-dimensional kernel used in CeSpGRN can work with cells forming any trajectory structures, as well as cells forming distinct clusters.

Gaussian Graphical Models (GGM) were used to represent biological networks or other networks [17, 18, 19]. GGMs were used in TREEGL [20] to learn multiple GRNs during a cell differentiation process which can be represented by a tree. TREEGL constrains the GRNs by a *total variation* (TV) penalty, which aims to minimize the differences between the learned GRNs. Compared to the weighted kernel approach adapted in KELLER and CeSpGRN, the TV penalty framework is computationally more expensive. This time complexity difference will be more pronounced in the case of inferring cell-specific GRNs given a large number of single cells. Moreover, TREEGL requires the tree as input, and in the case of scRNA-seq data, the tree which represents cell trajectories needs to be inferred. In contrast, CeSpGRN can deal with datasets where cells form trajectories of any graph structure (including tree structures), and it does not require the graph or tree information as input, as the graph structure is reflected in the kernel function constructed from high-dimensional gene expression data.

Recently, Gaussian Copula Graphical Models (GCGM) have been used to learn GRNs from single cell gene expression data and particularly scRNA-seq data [21, 22]. Compared to GGMs, GCGM account for the non-Gaussian nature of single cell gene expression data. However, existing methods using GCGMs cannot infer cell-specific networks but infer one GRN for a group of cells.

Through the use of GCGM model, CeSpGRN is not limited to gene expression data which are binary or Gaussian-distributed; and through the use of the high-dimensional weighted kernel, CeSpGRN can infer one GRN for each cell in datasets where cells can form any trajectory or cluster structures. We also extended CeSpGRN to a version named CeSpGRN-TF which can take advantage of prior information on which genes are transcription factors (TFs). We tested our methods on simulated datasets from two different simulators and two real datasets, and compared them with CSN and state-of-the-art GRN inference methods including GENIE3 [23] and SCODE [5].

2 Methods

In this section, we first introduce a scenario where a GGM or GCGM model is used to infer one GRN from n samples (that is, n cells in single cell gene expression data). Then we will introduce CeSpGRN including the design of the weighted kernel, and the use of GCGM, etc. Finally, we introduce an extended version of CeSpGRN that uses TF information, named CeSpGRN-TF.

2.1 Inferring GRNs using GGM and GCGM

If the gene expression data follow a multivariate Gaussian distribution, and the gene regulatory relationship is encoded in the sparse precision matrix (inverse covariance matrix) of the distribution, inferring a GRN from gene expression data is to infer the precision matrix of a Gaussian graphical model, which is a long-existing problem [17, 24, 25].

Denoting the gene expression data as a random vector $\mathbf{X} \in \mathbb{R}^g$, where g is the number of genes, and the adjacency matrix of the undirected GRN as $\mathbf{\Theta}$. Then $\mathbf{X}$ follows a multivariate Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}; \mathbf{\Theta}^{-1})$, and the gene expression data for a cell is a sample vector from that distribution. Denoting the gene expression data for cell i as $\mathbf{X}_i$, the precision matrix $\mathbf{\Theta}$ can be inferred using maximum likelihood estimation:

$$\begin{aligned}\hat{\mathbf{\Theta}} &= \arg\min_{\mathbf{\Theta}\in S^g_{++}} -\ell_{\mathbf{\Theta}}(\{\mathbf{X}_i\}_{i=1}^n) \\ &= \arg\min_{\mathbf{\Theta}\in S^g_{++}} -\frac{n}{2}\log|\mathbf{\Theta}| + \frac{n}{2}tr(\hat{\mathbf{\Sigma}}\mathbf{\Theta})\end{aligned} \tag{1}$$

S^g_{++} denotes the set of all symmetric positive definite matrices with dimension g, and $\hat{\mathbf{\Sigma}}$ is the empirical covariance matrix of genes estimated from sample data:

$$\hat{\mathbf{\Sigma}} = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{X}_i - \boldsymbol{\mu})(\mathbf{X}_i - \boldsymbol{\mu})^T, \quad \boldsymbol{\mu} \quad = \frac{1}{n}\sum_{i=1}^{n}\mathbf{X}_i \tag{2}$$

where n is the total number of cells.

Following [25], we incorporate ℓ_1 regularization on the inferred $\mathbf{\Theta}$ to account for the sparse property of GRN, and the optimization problem is now:

$$\begin{aligned}\hat{\mathbf{\Theta}} &= \arg\min_{\mathbf{\Theta}\in S^g_{++}} -\ell_{\mathbf{\Theta}}(\{\mathbf{X}_i\}_{i=1}^n) + \lambda\|\mathbf{\Theta}\|_1 \\ &= \arg\min_{\mathbf{\Theta}\in S^g_{++}} -\frac{n}{2}\log|\mathbf{\Theta}| + \frac{n}{2}tr(\hat{\mathbf{\Sigma}}\mathbf{\Theta}) + \lambda\|\mathbf{\Theta}\|_1\end{aligned} \tag{3}$$

λ is a hyper-parameter that regulates the sparsity of $\mathbf{\Theta}$. Multiple algorithms exist to solve this optimization problem, including Iterative Shrinkage Thresholding Algorithm (ISTA) [26], Alternating Direction Method of Multipliers (ADMM) [27] and Alternating Minimization (AM) [19].

In a GGM model, $\hat{\mathbf{\Sigma}}$ is calculated as the empirical covariance matrix from data, which assumes the data follows a Gaussian distribution. In a GCGM model, a *nonparametric covariance matrix* is calculated based on Kendall's τ correlation [15, 16] for data which do not follow a Gaussian distribution.

2.2 Inferring cell-specific GRNs (CeSpGRN)

The framework introduced in Section 2.1 can be used to infer one GRN from n cells. The problem we aim to address in this paper, which is to learn one GRN for each cell, defines a large parameter space that grows linearly with the number of cells. We incorporate additional constraints to help with finding solutions in such a space, which is that we assume the cells' GRNs change smoothly along the cell trajectory. When analyzing single cell gene expression data, the cell trajectories can be learned from the data through trajectory inference methods [28, 29, 30]. These methods use the principle that cells with similar gene expression profiles locate close to each other in the developmental trajectory. Therefore, assuming GRNs change smoothly along the cell trajectory which is obtained by gene expression similarity is the same as assuming cells with similar gene expression patterns have similar GRNs.

Based on this assumption, we can design a weighted kernel directly using the gene expression data, which is advantageous over first inferring the trajectory of cells then designing the weighted kernel using the inferred trajectory, since additional trajectory inference step can introduce inference errors. The weighted kernel allows us to "borrow" information from other cells when inferring GRN for a given cell: when inferring the GRN for cell i, which is $\mathbf{\Theta}_i$, we consider not only the likelihood of $\mathbf{X}_i$ against $\mathbf{\Theta}_i$, but also the likelihood of other cells' gene expression profile against $\mathbf{\Theta}_i$ weighted by a kernel function $\mathbf{K}()$. To formulate, when inferring the GRN for cell i, i.e. $\mathbf{\Theta}_i$, we define the following optimization problem:

$$\begin{aligned}\hat{\mathbf{\Theta}}_i &= \arg\min_{\mathbf{\Theta}_i \in S^g_{++}} -\sum_{j=1}^{n} \mathbf{K}_{ij}\ell_{\mathbf{\Theta}_i}(\{\mathbf{X}_j\}_{j=1}^n) + \lambda\|\mathbf{\Theta}_i\|_1 \\ &= \arg\min_{\mathbf{\Theta}_i \in S^g_{++}} \frac{n}{2}\sum_{j=1}^{n} \mathbf{K}_{ij}\left[-\log|\mathbf{\Theta}_i| + tr(\hat{\mathbf{\Sigma}}_j\mathbf{\Theta}_i)\right] + \lambda\|\mathbf{\Theta}_i\|_1\end{aligned} \tag{4}$$

where n is the total number of cells, $\mathbf{K}_{ij}$ is a kernel weight which account for the gene expression similarity between cells i and j, and $\hat{\mathbf{\Sigma}}_j$ is the nonparametric covariance matrix of genes in cells centered at cell j. Next, we describe how to calculate the kernel weight $\mathbf{K}_{ij}$ for all cell pairs and how to estimate $\hat{\mathbf{\Sigma}}_j$ for any cell j.

The kernel weight $\mathbf{K}_{ij}$ should reflect the transcriptome similarity between two cells i and j. We adapt the idea from manifold learning on scRNA-seq datasets [31, 32] and calculate $\mathbf{K}_{ij}$ for every pair of cells using the following steps: (1) Perform Principal Component Analysis (PCA) on the scRNA-seq data matrix, and calculate pairwise Euclidean distance between cells using their low-dimensional PCA representations; (2) Construct a k-nearest neighbor (kNN) graph between cells using the pairwise Euclidean distance; (3) Approximate the manifold distance between every two cells using the geodesic distance [33] between them on the kNN graph. (4) Denote the geodesic distance between cells i and j by D_{ij}, the kernel weight $\mathbf{K}_{ij}$ is then calculated using a Gaussian kernel function: $\mathbf{K}_{ij} = \exp(-D_{ij}^2/(2\sigma^2))$. σ is the bandwidth hyper-parameter that accounts for the differences between cell GRNs. Larger σ means that GRN is changing more slowly between cells. The number of neighbors in the kNN graph, k, is set to be the minimum number larger than 5 that makes the graph connected. We select this k value because a k value that is too large makes the estimated manifold distance vulnerable to short-circuit errors, and a k value that is too small tends to fragment the data manifold into disconnect regions [34].

In traditional GCGM models, the nonparametric covariance matrix is calculated from a group of samples. In CeSpGRN, we need to calculate $\hat{\mathbf{\Sigma}}_j$ for any cell j. We can again borrow information from cells around cell j in the gene expression space using the same kernel function $\mathbf{K}()$, and modify the original method in [16] to estimate $\hat{\mathbf{\Sigma}}_j$:

$$\tau_j(m,n) = \frac{\sum_{\substack{k,k'\in N(j)\\ k\neq k'}} w_{kk'}^{mn}\text{sign}((\mathbf{X}_k^m - \mathbf{X}_{k'}^m)(\mathbf{X}_k^n - \mathbf{X}_{k'}^n))}{|N(j)|\cdot(|N(j)|-1)}$$
$$\hat{\mathbf{\Sigma}}_j(m,n) = \begin{cases} \sin\frac{\pi}{2}\tau_j(m,n) & \text{if } m\neq n \\ 1 & \text{if } m = n \end{cases} \tag{5}$$

where $w_{kk'}^{mn} = \mathbf{K}_{jk}\mathbf{K}_{jk'}b_k^m b_{k'}^m b_k^n b_{k'}^n$, and $b_k^m, b_{k'}^m, b_k^n, b_{k'}^n$ are binary annotation for the zero values. $b_k^m = 1$ if the expression of gene m in cell k, $\mathbf{X}_k^m$, is not zero, and $b_k^m = 0$ if $\mathbf{X}_k^m = 0$. The estimated $\hat{\mathbf{\Sigma}}_j$ is not guaranteed to be positive definite, and we project $\hat{\mathbf{\Sigma}}_j$ into the positive definite sets by replacing its non-positive eigenvalues with a small positive value ϵ. $N(j)$ is the set of neighboring cells set of cell j. The size of neighborhood $|N(j)|$ is a hyper-parameter in the formula, with a similar role as that of the bandwidth parameter σ.

Having calculated kernel weight $\mathbf{K}_{ij}$ and $\hat{\mathbf{\Sigma}}_j$, we now can proceed to solve the optimization problem (Eq. 4). We use the ADMM algorithm to perform the optimization, as: (1) ADMM preserves the positive definiteness of $\mathbf{\Theta}_i$ for each iteration as long as the initial $\mathbf{\Theta}_i$ is positive definite (proof in Note S1); (2) ADMM quickly converges to a reasonably good suboptimum [27]; (3) The convergence condition and hyper-parameter choices for the algorithm are well-studied [27, 35]. In order to apply ADMM, we transform the original optimization problem into the following problem:

$$\hat{\mathbf{\Theta}}_i = \arg\min_{\mathbf{\Theta}_i,\mathbf{Z}\in S_{++}^g} \frac{n}{2}\sum_{j=1}^{n}\mathbf{K}_{ij}\left[-\log|\mathbf{\Theta}_i| + tr(\hat{\mathbf{\Sigma}}_j\mathbf{\Theta}_i)\right] + \lambda\|\mathbf{Z}\|_1$$
$$s.t. \quad \mathbf{\Theta}_i = \mathbf{Z} \tag{6}$$

the new optimization problem in Eq. 6 can be solved with `Algorithm 1` (detailed derivation in Note S2, pseudo-code in Note S3). Parameter ρ in the algorithm affects the convergence speed of the algorithm, and is set to be 1.7, following [35].

2.3 Inferring cell-specific GRN using known TF information (CeSpGRN-TF)

As the inference of GRNs has been a hard problem, prior information can be incorporated to help with the inference. For example, the TF information, *i.e.*, which genes are TFs and which genes are target genes, is known in some scenarios. We here present an extended version of CeSpGRN, CeSpGRN-TF, to take advantage of the TF information.

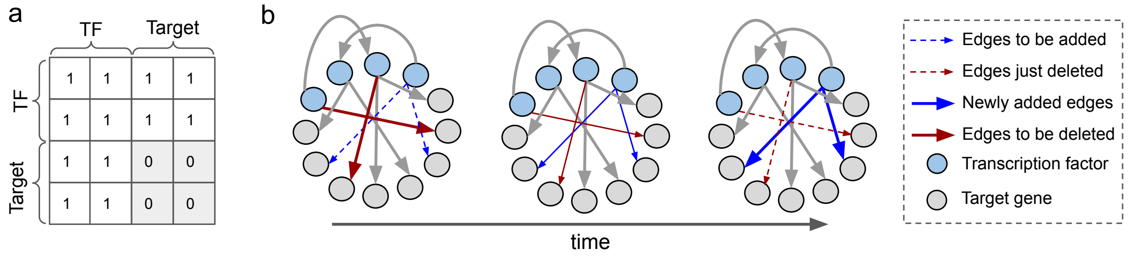


Figure 1: (a) An example binary mask with 2 TFs and 2 target genes. (b) Toy example showing the change of GRN within one simulation interval ($T = 3$).

We first transform the TF information into a binary masking matrix $\mathbf{M}$ (an example is shown in Fig. 1a), which masks out impossible interactions (interactions between non-TFs (target genes)). In $\mathbf{M}$, 0 means an interaction is impossible at the corresponding entry, and 1 otherwise. Incorporating the masking matrix $\mathbf{M}$ into the regularization term, we can construct the optimization problem of CeSpGRN-TF:

$$\begin{aligned}\hat{\boldsymbol{\Theta}}_i = \arg \min_{\boldsymbol{\Theta}_i \in S^g_{++}} \frac{n}{2} \sum_{j=1}^{n} \mathbf{K}_{ij} \left[-\log|\boldsymbol{\Theta}_i| + tr(\hat{\boldsymbol{\Sigma}}_j \boldsymbol{\Theta}_i) \right] \\ + \lambda \|\boldsymbol{\Theta}_i\|_1 + \beta \|\overline{\mathbf{M}} \odot \boldsymbol{\Theta}_i\|_F^2\end{aligned} \tag{7}$$

where β is a parameter that controls the strength of this constraint, and $\overline{\mathbf{M}}$ denotes the element-wise reversion ($0 \to 1$, $1 \to 0$) of $\mathbf{M}$. To apply ADMM for optimization, we transform the objective function into the following form and solve it using `Algorithm 2` (pseudo-code see Note S3).

$$\begin{aligned}\hat{\boldsymbol{\Theta}}_i = \arg \min_{\boldsymbol{\Theta}_i, \mathbf{Z} \in S^g_{++}} \frac{n}{2} \sum_{j=1}^{n} \mathbf{K}_{ij} \left[-\log|\boldsymbol{\Theta}_i| + tr(\hat{\boldsymbol{\Sigma}}_j \boldsymbol{\Theta}_i) \right] \\ + \lambda \|\mathbf{Z}\|_1 + \beta \|\bar{\mathbf{M}} \odot \mathbf{Z}\|_F^2, \quad s.t.\ \boldsymbol{\Theta}_i = \mathbf{Z}\end{aligned} \tag{8}$$

3 Results

Data. We tested the performance of our method on simulated data generated by two distinct simulators, and two real single cell gene expression datasets. The simulators are designed to generate single cell gene expression data where cells evolve along a developmental trajectory. We first generated cell-specific ground truth GRNs for the cells, then generated gene expression data for each cell given its GRN. For the latter step, the first simulator generates multivariant Gaussian (gene expression) data, and the second simulator was modified from BoolODE, a simulator used in [1] to benchmark various methods which infer GRNs from scRNA-seq data. The BoolODE simulator generates gene expression data using differential equations with a nonlinear hill function. The test on the first simulator is used to validate the method, and the test on the second simulator is used to show the applicability and accuracy of the methods on real life single cell gene expression datasets. In the tests, We consider two types of cell trajectories, one is "linear" where cells form a linear trajectory and the other is "bifurcating" where cells differentiate into two different cell fates.

Baseline methods for comparison with CeSpGRN. We used CSN [14], GENIE3 [23] and SCODE [5] as baseline methods. GENIE3 and SCODE were reported among top-performing GRN inference methods on scRNA-seq data [1]. GENIE3 is a regression-based method and SCODE is an ordinary differential equation-based method that requires time-series single cell gene expression data. We provide the ground truth simulation time as input to SCODE. GENIE3 and SCODE were designed to infer a static GRN from all the cells. In order to use them to obtain cell-specific GRNs to compare with our method, we designed their "dynamic" version, termed GENIE3-Dyn and SCODE-Dyn respectively. The dynamic versions work in the following manner: along the cell trajectory, we divided cells into segments of 100 cells. Then we ran GENIE3 or SCODE on each segment, and the inferred cell-specific GRN for a cell is the GRN of its corresponding segment. When using the original versions of GENIE3 and SCODE as baseline methods, their output cell-specific GRNs are the same for all cells, which is the static GRN inferred from all cells.

Evaluation metrics. To evaluate the quality of inferred cell-specific GRNs, we used Pearson correlation and Spearman correlation between each pair of ground truth and inferred adjacency matrices of GRNs in the same cell, as both the ground truth and estimated GRNs are weighted graphs. By replacing the nonzero values by 1s, we can binarize the ground truth networks and calculate typically used measures AUPRC (area under the precision-recall curve) and *early precision* scores. The early precision score is the fraction of true positives in the top-r edges ($r = \min(|E_{pred}|, |E_{true}|)$, where $|E_{pred}|$ and $|E_{true}|$ are the number of edges in respectively the predicted network and the true network) [1]. We also evaluated how well the methods detect the differences (*i.e.* the edges that change) between the GRNs of different cells. To evaluate the detection of changing edges, we also used the same four measures, Pearson correlation, Spearman correlation, AUPRC, and early precision scores.

3.1 Testing on multivariate Gaussian data

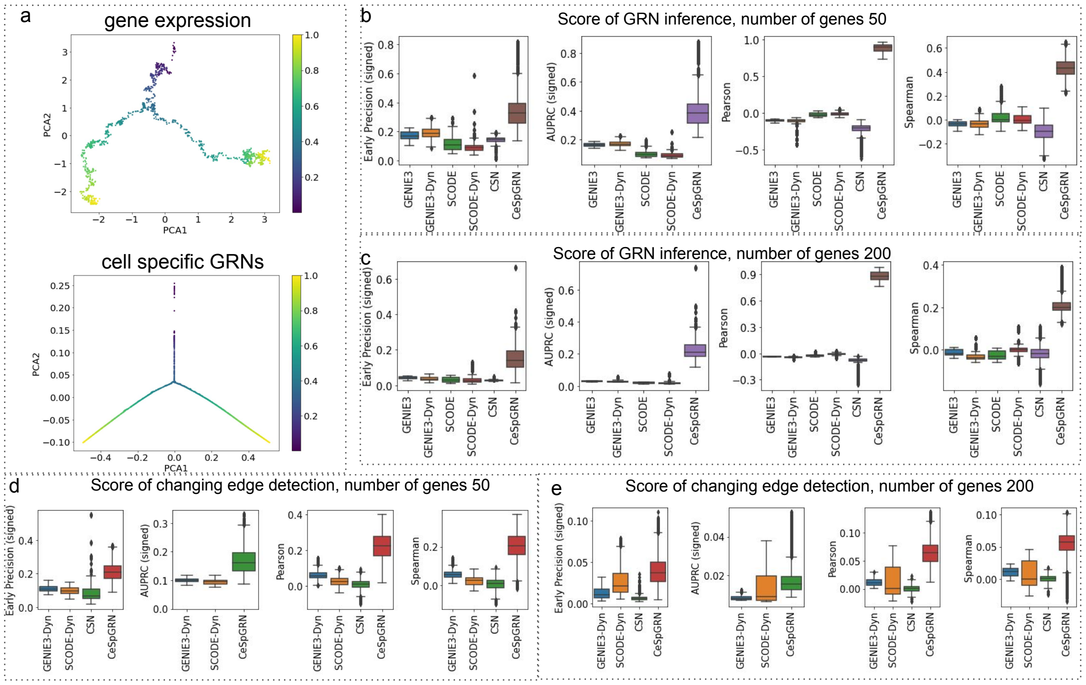


Figure 2: (a) PCA visualization of simulated gene expression data (upper plot) and corresponding CeSpGRN inferred GRNs (lower plot). Each point corresponds to a cell and is colored by cell simulation time. (b-c) Accuracy of cell-specific GRN inference by CeSpGRN and baseline methods on time-evolving Gaussian data. (d-e) Accuracy of detection of changing edges by CeSpGRN and baseline methods. Interval $T = 5$.

Data Simulation. First, we chose a trajectory structure (linear or bifurcating) and generated ground-truth GRNs for cells evolving along the trajectory, where each GRN corresponds to a cell. Each cell corresponds to a time point. An *interval* is defined to be T consecutive time steps. Initial GRNs are sampled from GRNs used in [36]. At the beginning of each interval, we randomly select m new edges to add and m edges to delete, and include a checking procedure that ensures each gene has at least one regulator and there is no regulation between two non-TF genes. Within an interval, there are no edge addition or deletion events, but there are changes in edge weights (with w_e/T difference at each time step, where w_e denote the maximum weight of the edge in the interval). The edge additions or deletions are accomplished at the end of each interval (Fig. 1b).

After the ground truth GRNs for all cells were generated, we generated the gene expression data for each cell using the procedure in Note S4. The generated gene expression data is visualized in 2-dimensional space using PCA in Fig. 2a (upper plot).

Experimental Design. We generated datasets with number of cells $n = 1000$, number of genes $g = 50, 200$, number of changing edges $m = 2$ and interval length $T = 5, 25$. For each simulation setting, we generated multiple datasets with different random seeds. Different interval lengths correspond to different graph changing speed.

Cell-specific GRN inference accuracy. We ran CeSpGRN with a wide range of hyper-parameters: bandwidth $\sigma = 0.1, 1.0, 10.0$, neighborhood size $|N(i)| = 15, 30, 100$, and the sparsity coefficient $\lambda = 0.001, 0.01, 0.1$. We visualized one simulated dataset along with its corresponding CeSpGRN inferred GRNs using PCA in Fig. 2a, and the result shows that the inferred GRNs evolve along a continuous bifurcating trajectory consistent with that of the gene expression data.

As there are edges with positive and negative weights in the GRNs, we distinguish the positive and negative edges when calculating the AUPRC and early precision scores: we calculate AUPRC and early precision scores for positive edges only and then do so for negative edges, and use the average of the scores for positive and negative edges, termed "AUPRC (signed)" and "early precision (signed)" scores respectively. The AUPRC scores for CSN are not available because CSN outputs binary networks which do not result in precision-recall curves. It is worth noting that GENIE3 and CSN are not able to distinguish edges with positive weights from those with negative weights, and they only return non-negative values for the edges. When calculating the evaluation scores for GENIE3 and CSN, we first take the absolute value of edges in the ground truth network and then conduct comparisons.

To account for the effects of hyper-parameters, we included results from all combinations of hyper-parameter values in the boxplots of CeSpGRN shown in Figs. 2b-e. One can observe that under a wide range of parameter settings and evaluation metrics, CeSpGRN infers more accurate cell-specific GRNs compared to baseline methods (Figs. 2b,c).

Detection of GRN edges that change between cells. We further evaluated the detection of changing edges by different methods (Figs. 2d,e). We measured the changes between pairs of inferred graphs, and compared them with the changing edges in the two corresponding ground truth graphs. Denote the ground truth GRNs for cells i and j by $\mathbf{G}_i$ and $\mathbf{G}_j$. We compared "inferred graph difference" $\hat{\mathbf{\Theta}}_i - \hat{\mathbf{\Theta}}_j$ with "true graph difference" $\mathbf{G}_i - \mathbf{G}_j$ using the same four metrics: early precision (signed) score, AUPRC (signed) score, Pearson and Spearman correlations. CeSpGRN captures the differences between cells' GRNs better than baseline methods that can infer cell-specific networks including GENIE3-Dyn, SCODE-Dyn, and CSN (Figs. 2d,e). The boxplots of CeSpGRN again include results from all parameter configurations and datasets. In Fig. 3 we visualize the edges that change from a cell i at simulation time 0.50 to a cell j at 0.54 (all cells' simulation time are in the range $[0, 1]$). A large portion of the changing edges are correctly detected by CeSpGRN.

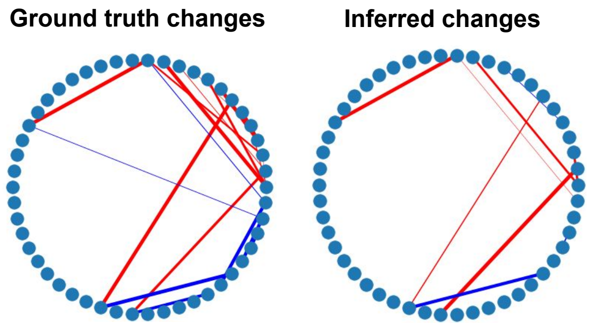


Figure 3: The edges that change from cell i to cell j. The left graph shows the ground truth changing edges, and the right shows the changing edges that CeSpGRN correctly detected. Red represents edges that are lost, and blue the edges that are gained. Width of edges represents weights.

Hyper-parameter effects and choices. In the results above we showed that CeSpGRN has superior performance even when we consider all parameter configurations together. Here we examine how the performance of CeSpGRN is affected by the hyper-parameters and provide guidance on parameter choices. We now show the performance of CeSpGRN in different parameter settings (Figs. S1a,b). First, we observe that bandwidth σ does not affect the results much and CeSpGRN is robust to σ. The performance tends to decrease when neighborhood size $|N(i)|$ increases. This is because the cell-specific graphs differ between cells, and using a large neighborhood assumes little difference between GRNs in a large neighborhood which is not true. We suggest setting σ to be between 1 and 10, and neighborhood size between 15 and 30. λ controls the sparsity of the inferred graph, which can affect the overall inference accuracy. It should be adjusted specifically for different testing scenarios. $\lambda = 0.1$ performs the best in our current test.

Overall, CeSpGRN obtained much better results than baseline methods in this set of experiments. Ce-

SpGRN benefits from the modeling of the continuous changes in GRNs and may have also benefited from the Gaussian distribution of the generated data. Next, we test CeSpGRN with non-Gaussian datasets.

3.2 Testing on simulated data from BoolODE model

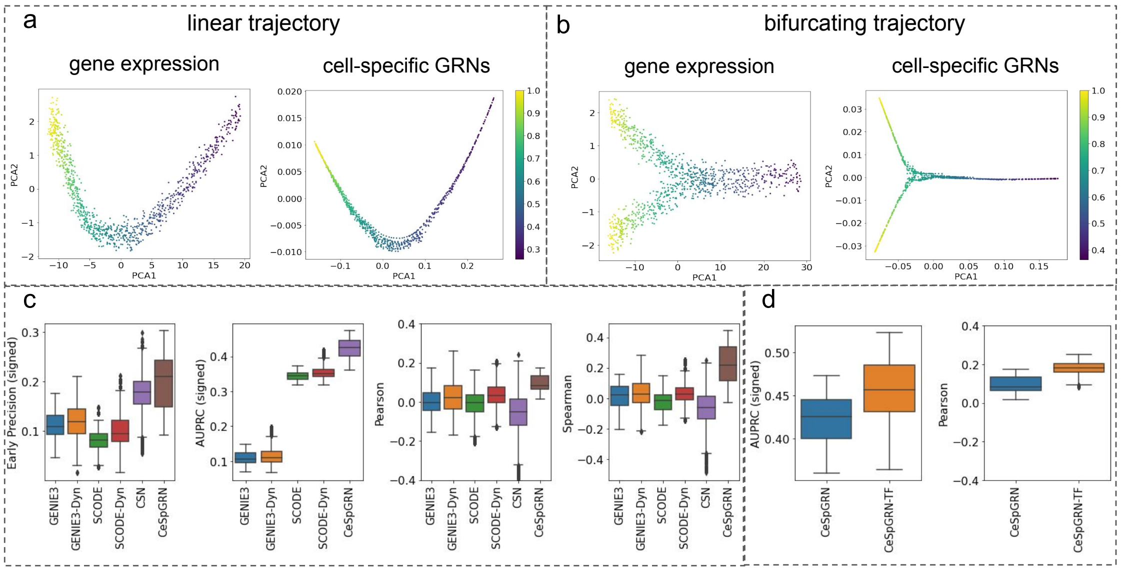


Figure 4: (a) PCA visualization of simulated gene expression data with linear and bifurcating trajectories. Each point corresponds to a cell and is colored by cell simulation time. (b) PCA visualization of CeSpGRN inferred GRNs. Each point corresponds to the GRN of a cell, and the color of points represents cell simulation time. (c-d) GRN inference accuracy scores of CeSpGRN, CeSpGRN-TF and other baseline methods.

Data simulation. There exist simulators that generate scRNA-seq data with an input GRN such as BoolODE [1], DynGen [37] and Sergio [36], but they do not allow each cell to use a distinct GRN. So we modified the BoolODE simulator such that each cell's gene expression data is generated from its cell-specific GRN. We start by generating the ground-truth GRNs for cells using the same procedure as used in the multivariate Gaussian simulator (Section 3.1). The initial GRNs are sampled from GRNs used in [36]. After having the ground-truth cell-specific GRNs, we generate single cell gene expression data using modified BoolODE, described as follows:

In BoolODE, the gene expression dynamics are modeled by stochastic differential equations. For a gene A, the change of its expression over time can be modeled as Eq. 9.

$$\frac{dA}{dt} = m_A \cdot f(B, C, D, \cdots; G(t)) - \ell_A \cdot A + dW \tag{9}$$

where $B, C, D, \cdots$ are the expressions of other genes, $G(t)$ is the GRN at the current time step, and dW is the random noise term. m_A and ℓ_A are mRNA synthesis and degradation rates, which are set to be constant. The regulating function $f(B, C, D, \cdots; G(t))$ is a hill function. When there is only one gene B that is regulating A in $G(t)$, the regulating function can be written as $f(B; G(t)) = \frac{\alpha_0 + \alpha_1 \left(\frac{B}{K_B}\right)^{N_B}}{1 + \left(\frac{B}{K_B}\right)^{N_B}}$. BoolODE constrains α_0 and α_1 to be binary values, but here we extend them to real values between 0 and 1 in order to model weighted GRNs. The new α_1 denote the normalized regulation strength between genes A and B, which corresponds to the weight of the edge between A and B in $G(t)$. And $\alpha_0 = 1 - \alpha_1$. K_B and N_B are hill threshold and hill coefficient, which are constant values. The function can be extended to the case of multiple regulators (see Note S5 for more details). Then the gene expression dynamics are generated from the differential equation (Eq. 9) using Euler method, and $G(t)$ is updated for each time step t. Cells are randomly sampled from the gene expression dynamics. The simulation details and simulation parameter settings are in Note S5.

Experimental Design. We used this simulator to generate 5 datasets with linear trajectories and 5 with bifurcating trajectories, with different random seeds. Each dataset has 1000 cells and 20 genes. Samples of the simulated datasets are visualized in Figs. 4a (linear trajectory) and b (bifurcating trajectory).

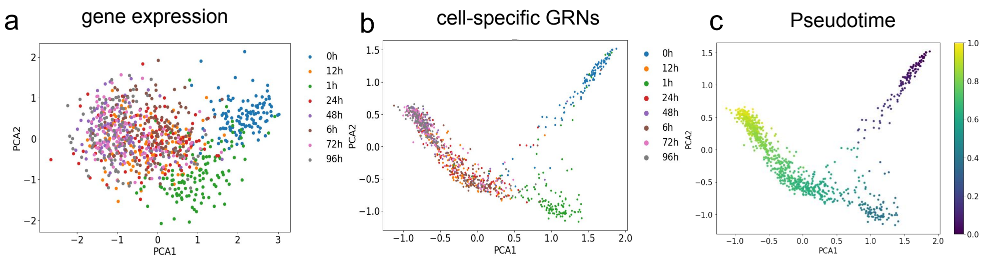
Cell-specific GRN inference accuracy. We tested CeSpGRN and the baseline methods on these simulated datasets. The expected trajectories can be clearly detected from CeSpGRN inferred GRNs (Figs. 4a,b). CeSpGRN has better inference accuracy of cell-specific GRNs compared to other methods under all four evaluation metrics (Fig. 4c). The hyper-parameters we used for the test are $\sigma = 10, |N(i)| = 30, \lambda = 0.001$, guided by our analysis in Section 3.1.

With this set of data, we also tested CeSpGRN-TF by incorporating the TF information for improved performance. Fig. 4d shows that incorporating TF information further improves the CeSpGRN performance.

3.3 Testing on human THP-1 cells dataset

We applied CeSpGRN to a THP-1 human myeloid monocytic leukemia cells dataset [38]. Kouno *et al* measured the gene expression levels of 45 TFs in 960 single cells gathered across 8 different time points using multiplex PCR. The cells are from the differentiation process of monocytic leukemia cells to macrophages. Fig. 5a shows the 2-dimensional visualization of the dataset with PCA.

We ran CeSpGRN to obtain cell-specific GRNs for this dataset. Ground truth cell-specific GRNs for real data are not available, but there exists an experimentally measured GRN for these 45 TFs for monocytic THP-1 cells [39], which we call a perturbation matrix. Because this GRN is not cell-specific but is corresponding to a population of cells, we obtained a population-level GRN from the CeSpGRN results to compare with the perturbation matrix. We summed up the adjacency matrices of the CeSpGRN inferred GRNs across all cells to obtain one network, and calculated the AUPRC ratio and the early precision ratio (EP ratio) against the perturbation matrix. The AUPRC ratio and EP ratio are ratios of the AUPRC score or EP score and the corresponding score of a random predictor [1]. The scores are calculated separately for positive and negative regulations and averaged into AUPRC ratio (signed) and EP ratio (signed) similar to Section 3.1. We used GENIE3, SCODE, and CSN to infer a population-level network as baseline for comparison (for CSN, we calculate the population-level network by summing up all cell-specific networks similar to CeSpGRN). GENIE3 and CSN cannot infer signed edges, so we compare GENIE3 and CSN results with the absolute values of the perturbation matrix. The results are summarized in Table 1. CeSpGRN infers GRNs with higher accuracy scores compared to the baseline methods.



d

	total	gain-ing	losing
up	23	2	3
mid	11	3	3
down	11	4	1

Figure 5: (a) PCA visualization of gene expression data. Cells are colored by the labeled time they are associated to. (b-c) PCA visualization of CeSpGRN inferred GRNs. GRNs are colored according to (b) the cells' labeled time (c) inferred pseudotime. (d) Numbers of genes in categories "upstream", "midstream" and "downstream" in different sets of genes.

We investigated the heterogeneity of the inferred cell-specific GRNs by CeSpGRN. First, to check whether the cell-specific GRNs, as a feature of the cells, change along the expected differentiation trajectory, we projected the adjacency matrices to 2-dimensional space with PCA and visualized them in Fig. 5b. The inferred networks form a differentiation trajectory, where cells are correctly ordered according to their timestamps. This trajectory appears to be less noisy than the trajectory formed by the gene expression data in Fig. 5a.

Obtaining the cell-specific GRNs allows us to examine the dynamic rewiring of the THP-1 regulatory network along the cell differentiation process. First, we applied a trajectory inference method, Slingshot [29], to the cells using the inferred cell-specific GRNs as features, in order to infer the pseudotime of cells (Fig. 5c). Then we detected regulatory edges that change weights significantly along the pseudotime by a *generalized additive model* as performed in [40]. Out of all edges detected, we removed the ones which are not in the experimentally measured perturbation matrix. The resulting list of changing edges is included in Table S1.

We then analyzed the genes associated with these changing edges. In particular, we took the set of genes which gain interactions through time, and the set of genes which lose interactions through time. The set of genes gaining interactions are: *BCL6*, *ETS1*, *PPARD*, *PPARG*,*SNAI1*,*SNAI3*, *SP3*, *SPIB*, *TCF3*, and the genes that lose interactions are: *FOSB*, *MYEF2*, *TCFL5*, *NFATC2*, *NFATC1*, *UHRF1*, *JUN*. In [38] the authors classified the 45 genes into upstream, midstream, and downstream genes. This classification is based on the location of the genes in the directed GRN measured experimentally at the population-level [39]. The table in Fig. 5d shows the number of genes in these three categories from (1) all the 45 genes (column "total"); (2) genes gaining interactions (column "gaining"); and (3) genes losing interactions (column "losing"). The gene names in each category are listed in Table. S2. It can be observed that genes that gain interactions are mostly mid- or down-stream genes, and genes that lose interactions are mostly mid- or up-stream genes. This indicates that in the cell-specific GRNs corresponding to early time points, upstream interactions in the population-level GRN are more active than downstream interactions, and in the GRNs of late time points, downstream interactions are more active.

	AUPRC ratio (signed)	EP ratio (signed)
CeSpGRN	1.205	3.287
GENIE3	1.177	1.176
CSN	NA	1.219
SCODE	1.258	1.759

Table 1: Accuracy of population-level GRN inferred by CeSpGRN, SCODE, CSN, and GENIE3 on the THP-1 monocytic cells.

3.4 Testing on mouse embryonic stem cell dataset

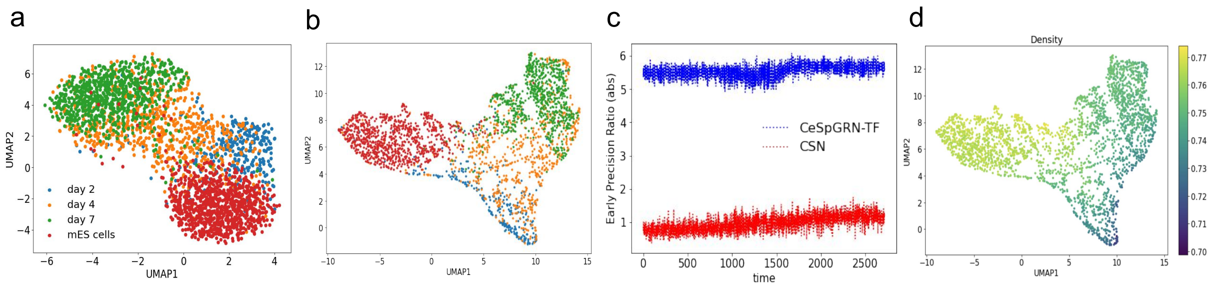


Figure 6: (a) UMAP visualization of gene expression data. Cells are colored by the cluster they are associated to. (b) UMAP visualization of cell-specific GRNs inferred by CeSpGRN-TF. Dots are colored according to the cells' labels. Legend is the same as the one used in (a). (c) Early Precision Ratio curve of CeSpGRN-TF and CSN, where cells are ordered along x-axis according to the inferred pseudotime. (d) UMAP visualization of CeSpGRN-TF inferred GRNs, where dots are colored with the GRN network density.

We further applied CeSpGRN to a mouse embryonic stem cells (mESC) dataset [41]. In this dataset, 2717 mESC were sequenced using scRNA-Seq technology. The dataset includes cells from the following developmental stages: "before leukemia inhibitory factor (LIF) withdrawal", "2 days after LIF withdrawal", "4 days after LIF withdrawal", and "7 day after LIF withdrawal". These four stages correspond to four cell types respectively labeled as "mES cells", "day 2", "day 4" and "day 7". Since LIF inhibits stem cell differentiation, the dataset records the various stages of mESC differentiation.

We preprocessed the dataset and selected 96 key genes which are highly variable genes and are also included in the key regulatory gene list reported in [42]. Among the 96 genes, 10 are known to be TFs (*Pou5f1*, *Nr5a2*, *Sox2*, *Sall4*, *Otx2*, *Esrrb*, *Stat3*, *Tcf7*, *Nanog*, *Etv5*) and the remaining are target genes [42]. Since we have known TF information, we inferred cell-specific GRNs using CeSpGRN-TF. The PCA visualizations of the scRNA-seq data and inferred cell-specific GRNs show a high density of cells in the "mES cells" and "day 7" cell types (Figs. S2a,b), so we used UMAP instead of PCA where overlapping cells are distinguished (Figs. 6a,b). Figs. 6a,b show that the inferred GRNs form a similar differentiation trajectory as the scRNA-seq counts.

We also ran CSN on this dataset, to output one network for each single cell but the network is constructed from pairwise relationships between genes. In [42], the authors compiled a population-level GRN in mESCs from multiple sources of information including gene expression data, binding site information and protein-protein interactions from databases. We used this network as the ground-truth population-level GRN, and compared the cell-specific networks inferred respectively by CeSpGRN-TF and CSN with it. Since AUPRC ratio cannot be used for CSN, only early precision ratios are calculated by comparing each inferred cell-specific GRN and the ground truth GRN. Fig. 6c shows the results where cells are ordered along pseudotime obtained from applying trajectory inference method on the scRNA-seq dataset.

The cell-specific GRNs inferred by CeSpGRN-TF allow us to analyze the dynamics of GRNs in this dataset. First we calculated the *density* of each inferred GRN. The density of a graph is the ratio between the number of edges with strength above a threshold and the number of all possible edges (which is g^2). We observed that the density of GRNs decreases along the differentiation process (Fig. 6d). This can indicate that at the beginning of the differentiation process there are more regulatory activities, and as differentiation progresses, less active regulatory interactions are needed.

We further selected the edges connected to three central regulators, *Pou5f1* (its protein is *Oct4*), *Sox2*, and *Nanog* [42], and ranked the edges according to the variance of their edge weights along the differentiation process (the full ranking list in Fig. S2c). CeSpGRN-TF detected top-ranking edges including *Pou5f1*-*Nanog*, *Esrrb*-*Nanog*, *Sox2*-*Otx2*, and *Sox2*-*Tcf7*. Those edges were discussed to be the key regulatory interactions for mESC differentiation in literature [42, 43, 44]. This can suggest that key regulatory interactions need not be active throughout the differentiation process but their strengths can be low at certain stages.

3.5 Running time

	number of genes: 50	number of genes: 200
running time (sec)	182.35	3654.61

Table 2: Running time of CeSpGRN under datasets with different sizes

CeSpGRN learns a high dimensional tensor of the shape (n, g, g), where n is the number of cells, g is the number of genes, and each matrix along the first dimension corresponds to the adjacency matrix of a cell-specific GRN. We accelerated the computation using GPU, and further parallelized the inference by optimizing one mini-batch of tensor a time instead of looping through the cells one by one. We generated 6 datasets with different random seeds, 3 with number of genes respectively equals to 50 and 200 to measure the running time (Table 2). We used `NVIDIA Tesla V100-32GB SXM2` on XSEDE [45] for the running time test.

4 Discussion and Future Work

We proposed CeSpGRN, which infers weighted, signed, and undirected cell-specific GRNs from single cell gene expression data. The GCGM model we used allows inference of both positive and negative regulatory edges, which is not the case for methods including GENIE3 and CSN. We have generated datasets with both linear and bifurcating continuous trajectories to test the performance of CeSpGRN. In principle, CeSpGRN works with datasets which form any trajectory structure, as well as datasets which form discrete clusters. This is because the kernel function we designed is based on pairwise manifold distance between cells, which makes CeSpGRN much more applicable than a broad category of methods which require time-series data. The use of Gaussian Copula Graphical Models allows CeSpGRN to work with data with a distribution which can be transformed into multivariate Gaussian through univariate monotone functions. We have demonstrated the application of CeSpGRN on simulated data from different simulators and real data from different technologies (multiplex PCR for the human THP-1 dataset and scRNA-seq for the mouse ESC dataset).

As the first method proposed to infer cell-specific GRN (CSN infers cell-specific pairwise gene-gene association), CeSpGRN can be further improved in the following directions. First, if there is prior information on which regulatory interactions do not change across cells, this information can be incorporated to improve the accuracy of GRN inference and changing edge detection. Second, CeSpGRN models the GRNs as undirected graphs, and future work can allow for modeling GRNs as directed graphs. In addition, with the advances of single cell multi-omics technologies, modeling multi-modality data into the network inference framework is also a promising future step.

Acknowledgements

This work was supported in part by the US National Science Foundation DBI-2019771 and National Institutes of Health grant R35GM143070. This work used the Extreme Science and Engineering Discovery Environment [45] (XSEDE, allocation ID TG-CIS210004), which is supported by National Science Foundation grant number ACI-1548562.

References

[1] Pratapa, A., Jalihal, A. P., Law, J. N., Bharadwaj, A. & Murali, T. Benchmarking algorithms for gene regulatory network inference from single-cell transcriptomic data. *Nature methods* **17**, 147–154 (2020).

[2] Chen, S. & Mar, J. C. Evaluating methods of inferring gene regulatory networks highlights their lack of performance for single cell gene expression data. *BMC Bioinformatics* **19**, 232 (2018).

[3] Aibar, S. *et al.* SCENIC: single-cell regulatory network inference and clustering. *Nat. Methods* **14**, 1083–1086 (2017).

[4] Chan, T. E., Stumpf, M. P. H. & Babtie, A. C. Gene regulatory network inference from Single-Cell data using multivariate information measures. *Cell Syst* **5**, 251–267.e3 (2017).

[5] Matsumoto, H. *et al.* SCODE: an efficient regulatory network inference algorithm from single-cell RNA-Seq during differentiation. *Bioinformatics* **33**, 2314–2321 (2017).

[6] Sanchez-Castillo, M., Blanco, D., Tienda-Luna, I. M., Carrion, M. C. & Huang, Y. A bayesian framework for the inference of gene regulatory networks from time and pseudo-time series data. *Bioinformatics* **34**, 964–970 (2018).

[7] Papili Gao, N., Ud-Dean, S. M. M., Gandrillon, O. & Gunawan, R. SINCERITIES: inferring gene regulatory networks from time-stamped single cell transcriptional expression profiles. *Bioinformatics* **34**, 258–266 (2018).

[8] Luscombe, N. M. *et al.* Genomic analysis of regulatory network dynamics reveals large topological changes. *Nature* **431**, 308–312 (2004).

[9] Ahmed, A. & Xing, E. P. Recovering time-varying networks of dependencies in social and biological studies. *Proc. Natl. Acad. Sci. U. S. A.* **106**, 11878–11883 (2009).

[10] Song, L., Kolar, M. & Xing, E. P. KELLER: estimating time-varying interactions between genes. *Bioinformatics* **25**, i128–36 (2009).

[11] Przytycka, T. M., Singh, M. & Slonim, D. K. Toward the dynamic interactome: it's about time. *Briefings in Bioinformatics* **11**, 15–29 (2010).

[12] Kim, H. J. *et al.* Transcriptional network dynamics during the progression of pluripotency revealed by integrative statistical learning. *Nucleic Acids Res.* **48**, 1828–1842 (2020).

[13] Kouno, T. *et al.* Temporal dynamics and transcriptional control using single-cell gene expression analysis. *Genome Biol.* **14**, R118 (2013).

[14] Dai, H., Li, L., Zeng, T. & Chen, L. Cell-specific network constructed by single-cell RNA sequencing data. *Nucleic Acids Res.* **47**, e62 (2019).

[15] Liu, H., Han, F., Yuan, M., Lafferty, J. & Wasserman, L. High-dimensional semiparametric Gaussian copula graphical models. *The Annals of Statistics* **40**, 2293–2326 (2012).

[16] Wang, H., Fazayeli, F., Chatterjee, S. & Banerjee, A. Gaussian copula precision estimation with missing values. In *Artificial Intelligence and Statistics*, 978–986 (PMLR, 2014).

[17] Dobra, A. *et al.* Sparse graphical models for exploring gene expression data. *J. Multivar. Anal.* **90**, 196–212 (2004).

[18] Wang, T. *et al.* FastGGM: An efficient algorithm for the inference of Gaussian graphical model in biological networks. *PLoS Comput. Biol.* **12**, e1004755 (2016).

[19] Shrivastava, H. *et al.* GLAD: Learning sparse graph recovery. In *International Conference on Learning Representations* (2020). URL `https://openreview.net/forum?id=BkxpMTEtPB`.

[20] Parikh, A. P., Wu, W., Curtis, R. E. & Xing, E. P. TREEGL: reverse engineering tree-evolving gene networks underlying developing biological lineages. *Bioinformatics* **27**, i196–i204 (2011).

[21] Wu, N., Yin, F., Ou-Yang, L., Zhu, Z. & Xie, W. Joint learning of multiple gene networks from single-cell gene expression data. *Computational and structural biotechnology journal* **18**, 2583–2595 (2020).

[22] Dong, M., He, Y., Jiang, Y. & Zou, F. Joint gene network construction by single-cell RNA sequencing data. *bioRxiv* (2021).

[23] Huynh-Thu, V. A., Irrthum, A., Wehenkel, L. & Geurts, P. Inferring regulatory networks from expression data using tree-based methods. *PloS one* **5**, e12776 (2010).

[24] Friedman, J., Hastie, T. & Tibshirani, R. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics* **9**, 432–441 (2008).

[25] Banerjee, O., El Ghaoui, L. & d'Aspremont, A. Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data. *J. Mach. Learn. Res.* **9**, 485–516 (2008).

[26] Guillot, D., Rajaratnam, B., Rolfs, B. T., Maleki, A. & Wong, I. Iterative thresholding algorithm for sparse inverse covariance estimation. *arXiv preprint arXiv:1211.2532* (2012).

[27] Boyd, S., Parikh, N. & Chu, E. *Distributed optimization and statistical learning via the alternating direction method of multipliers* (Now Publishers Inc, 2011).

[28] Trapnell, C. *et al.* The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells. *Nature biotechnology* **32**, 381–386 (2014).

[29] Street, K. *et al.* Slingshot: cell lineage and pseudotime inference for single-cell transcriptomics. *BMC genomics* **19**, 1–16 (2018).

[30] Haghverdi, L., Büttner, M., Wolf, F. A., Buettner, F. & Theis, F. J. Diffusion pseudotime robustly reconstructs lineage branching. *Nature methods* **13**, 845–848 (2016).

[31] Zhang, Z., Yang, C. & Zhang, X. Learning latent embedding of multi-modal single cell data and cross-modality relationship simultaneously. *bioRxiv* (2021).

[32] Moon, K. R. *et al.* Visualizing structure and transitions in high-dimensional biological data. *Nature biotechnology* **37**, 1482–1492 (2019).

[33] Tenenbaum, J. B., De Silva, V. & Langford, J. C. A global geometric framework for nonlinear dimensionality reduction. *science* **290**, 2319–2323 (2000).

[34] Balasubramanian, M., Schwartz, E. L., Tenenbaum, J. B., de Silva, V. & Langford, J. C. The isomap algorithm and topological stability. *Science* **295**, 7–7 (2002).

[35] Nishihara, R., Lessard, L., Recht, B., Packard, A. & Jordan, M. A general analysis of the convergence of ADMM. In *International Conference on Machine Learning*, 343–352 (PMLR, 2015).

[36] Dibaeinia, P. & Sinha, S. Sergio: a single-cell expression simulator guided by gene regulatory networks. *Cell Systems* **11**, 252–271 (2020).

[37] Cannoodt, R., Saelens, W., Deconinck, L. & Saeys, Y. Spearheading future omics analyses using dyngen, a multi-modal simulator of single cells. *Nature Communications* **12**, 1–9 (2021).

[38] Kouno, T. *et al.* Temporal dynamics and transcriptional control using single-cell gene expression analysis. *Genome biology* **14**, 1–12 (2013).

[39] Tomaru, Y. *et al.* Regulatory interdependence of myeloid transcription factors revealed by Matrix RNAi analysis. *Genome biology* **10**, 1–13 (2009).

[40] Zhang, Z. & Zhang, X. Inference of high-resolution trajectories in single-cell RNA-seq data by using RNA velocity. *Cell Reports Methods* **1**, 100095 (2021).

[41] Klein, A. M. *et al.* Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells. *Cell* **161**, 1187–1201 (2015).

[42] Zhou, Q., Chipperfield, H., Melton, D. A. & Wong, W. H. A gene regulatory network in mouse embryonic stem cells. *Proceedings of the National Academy of Sciences* **104**, 16438–16443 (2007).

[43] Sevilla, A. *et al.* An esrrb and nanog cell fate regulatory module controlled by feed forward loop interactions. *Frontiers in cell and developmental biology* **9**, 502 (2021).

[44] Loh, Y.-H. *et al.* The oct4 and nanog transcription network regulates pluripotency in mouse embryonic stem cells. *Nature genetics* **38**, 431–440 (2006).

[45] Towns, J. *et al.* XSEDE: Accelerating scientific discovery. *Computing in Science Engineering* **16**, 62–74 (2014).

Supplementary Notes, Figures and Tables

Note S1: Algorithms 1 and 2 preserve the positive definiteness of the inferred matrix

When we initialize $\mathbf{Z}^0$ to be symmetric positive definite, then according to algorithm 1 and 2, we will always have $\mathbf{Z}^t$ to be symmetric. Then $\frac{1}{2}(\mathbf{U} - \mathbf{Z}^t + \bar{\mathbf{\Sigma}}_i)$ is also symmetric, and can be eigenvalue decomposed as $\mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$, where $\mathbf{V}^T\mathbf{V} = \mathbf{V}\mathbf{V}^T = \mathbf{I}$. Rewriting the updating rule of $\mathbf{\Theta}_i^{t+1}$ in Algorithm 1 and 2 as

$$\begin{aligned}\mathbf{\Theta}_i^{t+1} &= \sqrt{\frac{1}{\rho}\mathbf{I} + \frac{1}{4}(\mathbf{U}^t - \mathbf{Z}^t + \bar{\mathbf{\Sigma}}_i)^T(\mathbf{U}^t - \mathbf{Z}^t + \bar{\mathbf{\Sigma}}_i)} - \frac{1}{2}(\mathbf{U}^t - \mathbf{Z}^t + \bar{\mathbf{\Sigma}}_i)\\ &= \sqrt{\frac{1}{\rho}\mathbf{I} + \mathbf{V}\mathbf{\Lambda}^2\mathbf{V}^T} - \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T\\ &= \mathbf{V}\left(\left(\mathbf{\Lambda}^2 + \frac{1}{\rho}\mathbf{I}\right)^{\frac{1}{2}} - \mathbf{\Lambda}\right)\mathbf{V}^T\end{aligned} \tag{1}$$

Since $\left(\left(\mathbf{\Lambda}^2 + \frac{1}{\rho}\mathbf{I}\right)^{\frac{1}{2}} - \mathbf{\Lambda}\right)$ has minimum eigenvalue greater than 0, $\mathbf{\Theta}_i^{t+1}$ is always positive definite. Both algorithm 1 and 2 are able to preserve the symmetric positive definiteness of $\mathbf{\Theta}_i^{t+1}$.

Note S2: Derivation of Algorithms 1 and 2

We can solve the optimization problem in Eq. 6 following the iterative updating rule as below:

$$\begin{aligned}\mathbf{\Theta}_i^{t+1} &= \arg\min_{\mathbf{\Theta}_i} - \log|\mathbf{\Theta}_i| + tr\left(\sum_{j=1}^{n}(\mathbf{K}(i,j)\hat{\mathbf{\Sigma}}_j)\cdot\mathbf{\Theta}_i\right) + \frac{\rho}{2}\|\mathbf{\Theta}_i - \mathbf{Z}^t + \mathbf{U}^t\|_F^2\\ \mathbf{Z}^{t+1} &= \arg\min_{\mathbf{Z}} \frac{\rho}{2}\|\mathbf{\Theta}_i^{t+1} - \mathbf{Z} + \mathbf{U}^t\|_F^2 + \lambda\|\mathbf{Z}\|_1\\ \mathbf{U}^{t+1} &= \mathbf{U}^t + \mathbf{\Theta}_i^{t+1} - \mathbf{Z}^{t+1}\end{aligned} \tag{2}$$

Then, we calculate the $\mathbf{\Theta}_i^{t+1}$ by setting the gradient to 0:

$$\begin{aligned}&-\frac{1}{\rho}\left(\mathbf{I} - \sum_{j=1}^{n}(\mathbf{K}(i,j)\hat{\mathbf{\Sigma}}_j)\cdot\mathbf{\Theta}_i^{t+1}\right) + \mathbf{\Theta}_i^{(t+1)T}\mathbf{\Theta}_i^{t+1} - \mathbf{Z}^t\mathbf{\Theta}_i^{t+1} + \mathbf{U}^t\mathbf{\Theta}_i^{t+1} = 0\\ &(\mathbf{\Theta}_i^{t+1} + \frac{1}{2}(\mathbf{U}^t - \mathbf{Z}^t + \bar{\mathbf{\Sigma}}_i))^2 = \frac{1}{\rho}\mathbf{I} + \frac{1}{4}(\mathbf{U}^t - \mathbf{Z}^t + \bar{\mathbf{\Sigma}}_i)^T(\mathbf{U}^t - \mathbf{Z}^t + \bar{\mathbf{\Sigma}}_i)\end{aligned} \tag{3}$$

And

$$\begin{aligned}\mathbf{\Theta}_i^{t+1} &= \sqrt{\frac{1}{\rho}\mathbf{I} + \frac{1}{4}(\mathbf{U}^t - \mathbf{Z}^t + \bar{\mathbf{\Sigma}}_i)^T(\mathbf{U}^t - \mathbf{Z}^t + \bar{\mathbf{\Sigma}}_i)} \\ &- \frac{1}{2}(\mathbf{U}^t - \mathbf{Z}^t + \bar{\mathbf{\Sigma}}_i) \\ \bar{\mathbf{\Sigma}}_i &= \frac{1}{\rho}\sum_{j=1}^{n}\mathbf{K}(i,j)\hat{\mathbf{\Sigma}}_j\end{aligned} \tag{4}$$

We can also calculate $\mathbf{Z}^{t+1}$ by setting the gradient of $\frac{\rho}{2}\|\mathbf{\Theta}_i^{t+1} - \mathbf{Z} + \mathbf{U}^t\|_F^2 + \lambda\|\mathbf{Z}\|_1$ to 0, which corresponds to soft-thresholding formula $sign(\mathbf{\Theta}_i^{t+1} + \mathbf{U}^t)\left(|\mathbf{\Theta}_i^{t+1} + \mathbf{U}^t| - \frac{\lambda}{\rho}\right)_+$.

We can derive the updating rule of algorithm 2 following similar procedure as above.

Note S3: Pseudo-code of CeSpGRN and CeSpGRN-TF

Algorithm 1 Infer cell-specific GRN

```
1: function CeSpGRN($\mathbf{K}$, $\{\hat{\mathbf{\Sigma}}_i\}_{i=1}^n$, $t_{max}$) // $t_{max}$ is the max number of iterations
2:     $\mathbf{\Theta} = \{\}$ // store GRN for all cells
3:     for i in $1, 2, \cdots, n$ do
4:         Initialize $\bar{\mathbf{\Sigma}}_i = \frac{1}{\rho}\sum_{j=1}^n \mathbf{K}_{ij} \cdot \hat{\mathbf{\Sigma}}_j$;   $\mathbf{U}^1 = \mathbf{0}$
5:         Initialize $\mathbf{Z}^1 = \text{diag}(\bar{\mathbf{\Sigma}}_i)$ // make sure symmetric positive definite
6:         for t in $1, 2, \cdots, t_{max}$ do
7:             $\mathbf{\Theta}_i^{t+1} = \sqrt{\frac{1}{\rho}\mathbf{I} + \frac{1}{4}(\mathbf{U}^t - \mathbf{Z}^t + \bar{\mathbf{\Sigma}}_i)^T(\mathbf{U}^t - \mathbf{Z}^t + \bar{\mathbf{\Sigma}}_i)} - \frac{1}{2}(\mathbf{U}^t - \mathbf{Z}^t + \bar{\mathbf{\Sigma}}_i)$
8:             $\mathbf{Z}^{t+1} = sign(\mathbf{\Theta}_i^{t+1} + \mathbf{U}^t)\left(|\mathbf{\Theta}_i^{t+1} + \mathbf{U}^t| - \frac{\lambda}{\rho}\right)_+$
9:             $\mathbf{U}^{t+1} = \mathbf{U}^t + \mathbf{\Theta}_i^{t+1} - \mathbf{Z}^{t+1}$
10:        $\mathbf{\Theta} = \mathbf{\Theta} \cup \mathbf{Z}^{t+1}$
11:    return $\mathbf{\Theta}$
```

Algorithm 2 Infer cell-specific GRN with TF information

```
1: function CeSpGRN-TF($\mathbf{K}$, $\{\hat{\mathbf{\Sigma}}_i\}_{i=1}^n$, $\mathbf{M}$, $t_{max}$) // $t_{max}$ is the max number of iterations
2:     $\mathbf{\Theta} = \{\}$ // store GRN for all cells
3:     for i in $1, 2, \cdots, n$ do
4:         Initialize $\bar{\mathbf{\Sigma}}_i = \frac{1}{\rho}\sum_{j=1}^n \mathbf{K}_{ij} \cdot \hat{\mathbf{\Sigma}}_j$;   $\mathbf{U}^1 = \mathbf{0}$
5:         Initialize $\mathbf{Z}^1 = \text{diag}(\bar{\mathbf{\Sigma}}_i)$ // ensure symmetric positive definite
6:         for t in $1, 2, \cdots, t_{max}$ do
7:             $\mathbf{\Theta}_i^{t+1} = \sqrt{\frac{1}{\rho}\mathbf{I} + \frac{1}{4}(\mathbf{U}^t - \mathbf{Z}^t + \bar{\mathbf{\Sigma}}_i)^T(\mathbf{U}^t - \mathbf{Z}^t + \bar{\mathbf{\Sigma}}_i)} - \frac{1}{2}(\mathbf{U}^t - \mathbf{Z}^t + \bar{\mathbf{\Sigma}}_i)$
8:             $\mathbf{Z}^{t+1} = sign(\mathbf{\Theta}_i^{t+1} + \mathbf{U}^t)\left(\frac{\rho|\mathbf{\Theta}_i^{t+1} + \mathbf{U}^t| - \lambda}{\rho + 2\beta\mathbf{M}^2}\right)_+$
9:             $\mathbf{U}^{t+1} = \mathbf{U}^t + \mathbf{\Theta}_i^{t+1} - \mathbf{Z}^{t+1}$
10:        $\mathbf{\Theta} = \mathbf{\Theta} \cup \mathbf{Z}^{t+1}$
11:    return $\mathbf{\Theta}$
```

Note S4: Simulating multivariate Gaussian gene expression data

(1) Calculate the covariance matrix of each graph. A positive bias is added on the diagonal values in the adjacency matrix of the graph to make sure that the adjacency matrix is positive definite, following [1, 3].

Then the covariance matrix is calculated as the inverse of the adjacency matrix. (2) Gene expression data is sampled from multivariate Gaussian distribution defined by the covariance matrix. After we obtain the gene expression data $\mathbf{X}_t$ for cell t, we set $\mathbf{X}_t$ to be the mean used in the multivariate Gaussian for generating $\mathbf{X}_{t+1}$. We generated single cell gene expression data in this manner following a predefined bifurcating cell trajectory.

Note S5: Simulating scRNA-Seq data from modified BoolODE model

Extending the regulation function. We explain the extension using an example where gene A is controlled by P, Q, R according to the GRN $G(t)$. Denoting

$$\begin{aligned} H_P &= \left(\frac{P}{K_P}\right)^{N_P} \\ H_Q &= \left(\frac{Q}{K_Q}\right)^{N_Q} \\ H_R &= \left(\frac{R}{K_R}\right)^{N_R} \end{aligned} \tag{5}$$

where N_P, N_Q, N_R are gene-specific hill threshold, and K_P, K_Q, K_R are gene-specific hill coefficient. Then the extended regulation function is

$$f(P,Q,R) = \frac{\alpha_0 + \alpha_P H_P + \alpha_Q H_Q + \alpha_{PQ} H_P H_Q + \alpha_{PR} H_P H_R + \alpha_{QR} H_Q H_R + \alpha_{PQR} H_P H_Q H_R}{1 + H_P + H_Q + H_P H_Q + H_P H_R + H_Q H_R + H_P H_Q H_R} \tag{6}$$

In the original BoolODE simulator, α are binary values denoting different binding cases, i.e. P, Q, R not binding; P binding, Q, R not binding; Q binding, P, R not binding; etc. The value of α can be calculated through the boolean regulatory formula, by give the corresponding regulator 1 if binding happens, and 0 if binding not happens. For example, if the regulating relationship is denoted as $A = (P \vee Q) \wedge R$, then α_0, denoting the case where all three bindings not happen, can be calculated as $\alpha_0 = (0 \vee 0) \wedge 1 = 0$.

To make the formula account for the regulation strength, the straightforward way is to change α from boolean to soft value. We make the inhibitor has α value close to 0, the activator has α value close to 1. And we make the combinatorial regulation to the sum or product of individual regulation value (replace $\vee$ with $+$, $\wedge$ with $\cdot$).

Generating datasets with different trajectory backbones. The simulation procedure above is only able to generate single-cell dataset with linear trajectory structure. In order to make the simulator to generate the datasets with different trajectory structures like bifurcating structure, we create anchor time points along the simulation time. When the evolved GRN hits a anchor time point, it will branch into two or more evolving directions from the anchor time point, which will guide the gene expression dynamics to branch into multiple directions and thus create datasets with different trajectory structures.

Simulation parameters. The simulation parameters are selected based on BoolODE [2]. Hill threshold is set to be 10 for all genes, hill coefficient is set to be 10 for all genes, gene transcription rate is set to be 200, and degradation rate is set to be 10. Simulation time step is set to be 0.001, and the number of changing edges for each interval is set to be 4.

Supplementary Figures and tables

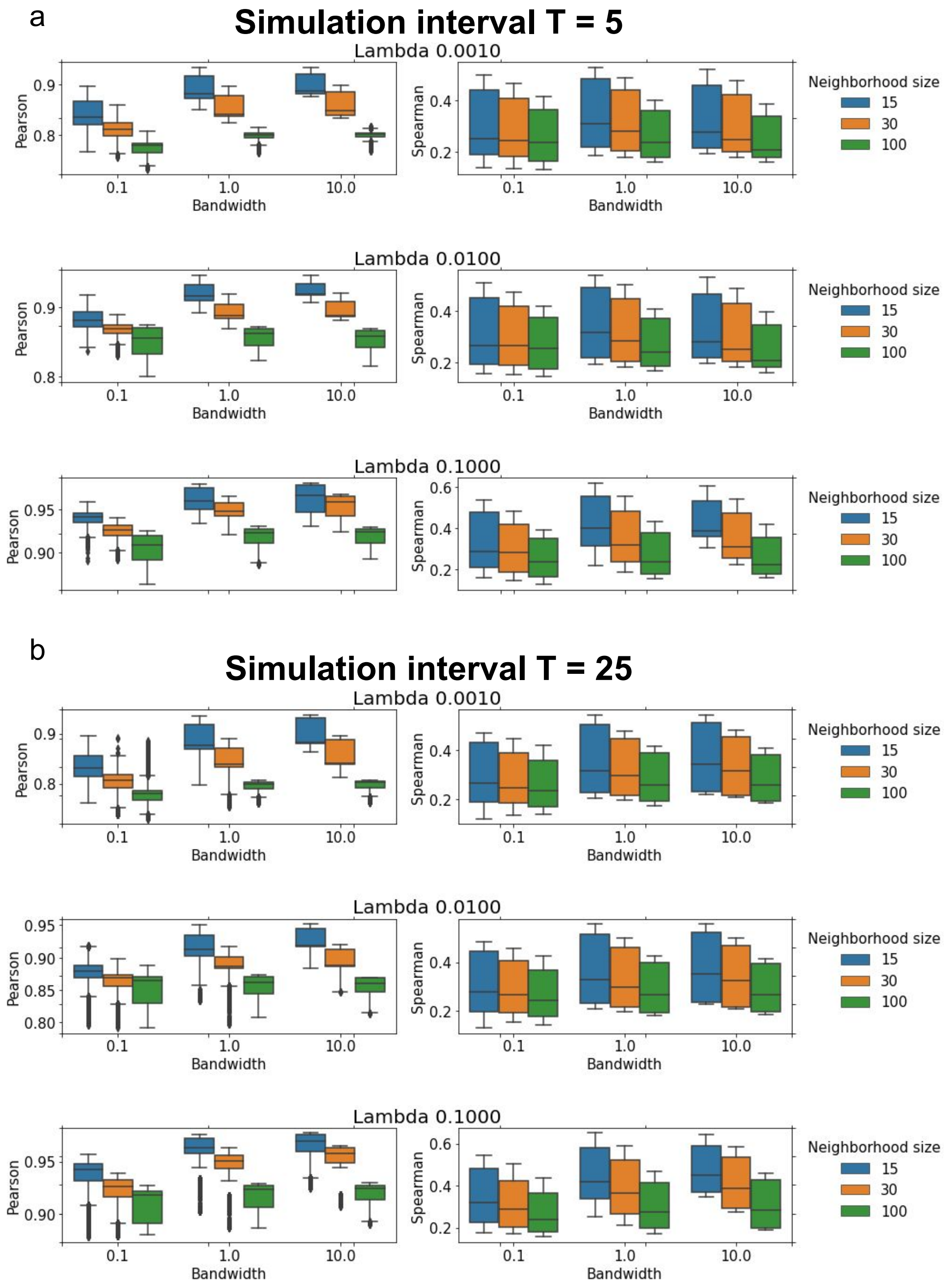
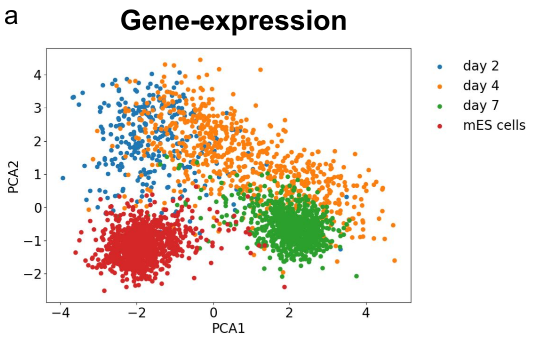
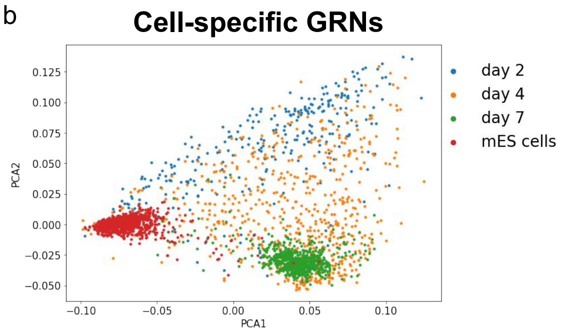


Figure S1: (a-b) Performance of CeSpGRN with different hyper-parameters. Datasets with both fast changing graphs (upper) and slow changing graphs (lower) are used.

a



b



c

Pou5f1

Genes	Variance
Sox2	5.14e-5
Esrrb	3.13e-5
Nanog	2.19e-5
Stat3	1.88e-5
Etv5	1.34e-5

Sox2

Genes	Variance
Pou5f1	5.14e-5
Tcf7	2.93e-5
Sall4	2.90e-5
Nr5a2	6.58e-6
Otx2	4.64e-6

Nanog

Genes	Variance
Otx2	1.15e-4
Esrrb	7.62e-4
Nr5a2	3.92e-5
Pou5f1	2.19e-5
Etv5	3.44e-6

Figure S2: (a) PCA visualization of gene expression, where each cells correspond to one dot, and cells are colored by the cluster label. (b) PCA visualization of cell-specific GRNs, where each GRN correspond to one dot, and GRNs are colored by the cluster label. (c) Ranking of the variances of edges that connect to *Pou5f1*, *Sox2*, and *Nanog*.

Gained through time	Lost through time
BCL6-TCF3	EGR2-FOSB
EGR2-SNAI1	EGR2-HOXA13
ETS1-PPARD	EGR2-MAFB
HOXA13-SPIB	EGR2-NFATC1
MAFB-PPARD	FOSB-MAFB
MAFB-PPARG	JUN-MAFB
MAFB-SNAI3	MAFB-TCFL5
MYB-SP3	MYEF2-UHRF1
	MYB-NFATC2

Table S1: List of differentially rewired edges in human THP-1 cells dataset.

	gaining	**losing**
Upstream	SP3	NFATC1
	TCF3	UHRF1
		JUN
Midstream	PPARD	FOSB
	SNAI1	MYEF2
	SPIB	TCFL5
Downstream	SNAI3	NFATC2
	PPARG	
	ETS1	
	BCL6	

Table S2: The table of the genes that are gaining interactions and losing interactions in human THP-1 cells dataset. Genes are categorized into Upstream, Midstream and Downstream genes.

References

[1] Dominique Guillot, Bala Rajaratnam, Benjamin T Rolfs, Arian Maleki, and Ian Wong. Iterative thresholding algorithm for sparse inverse covariance estimation. *arXiv preprint arXiv:1211.2532*, 2012.

[2] Aditya Pratapa, Amogh P Jalihal, Jeffrey N Law, Aditya Bharadwaj, and TM Murali. Benchmarking algorithms for gene regulatory network inference from single-cell transcriptomic data. *Nature methods*, 17(2):147–154, 2020.

[3] Harsh Shrivastava, Xinshi Chen, Binghong Chen, Guanghui Lan, Srinivas Aluru, Han Liu, and Le Song. GLAD: Learning sparse graph recovery. In *International Conference on Learning Representations*, 2020.