From Noise to Knowledge: Probabilistic Diffusion-Based Neural Inference of Gene Regulatory Networks

Hao Zhu[1[0000−0002−3386−6076]] and Donna K. Slonim[1[0000−0003−3357−437X]]

Tufts University, Medford MA 02155, USA
{hao.zhu,donna.slonim}@tufts.edu

Abstract. Understanding gene regulatory networks (GRNs) is crucial for unraveling cellular mechanisms and enhancing therapeutic interventions. Original methods for GRN inference from bulk expression data often struggled with the high dimensionality and inherent noise in the data. The prevalence of dropout events and background noise inherent in single-cell RNA sequencing has only increased the challenges. Here we introduce RegDiffusion, a novel neural network structure inspired by Denoising Diffusion Probabilistic Models but focusing on the regulatory effects among feature variables. Unlike other GRN methods for single-cell RNA-sequencing data, RegDiffusion introduces Gaussian noise to the input data following a diffusion schedule. It is subsequently trained to predict the added noise using a neural network with a parameterized adjacency matrix. We show that using this process, GRNs can be effectively learned with a surprisingly simple model architecture. In our benchmark experiments, RegDiffusion demonstrates superior performance compared to several baseline methods on multiple datasets. This work not only introduces a fresh perspective on GRN inference but also highlights the promising capacity of diffusion-based models in the area of single-cell analysis. The RegDiffusion software is available at https://github.com/TuftsBCB/RegDiffusion.

Keywords: Regulatory genomics · single-cell analysis · diffusion models.

1 Introduction

Gene regulatory networks (GRNs) describe the underlying regulatory network governing transcription and control over various cellular function [2,15,27]. Understanding how these networks work can shed light on processes as diverse as pathology and normal development. Moreover, understanding gene regulation from a systems perspective may help identify key points of regulation that could be amenable to therapeutic modulation [3].

Despite a growing understanding of the binding sites of many regulatory proteins, GRNs are often inferred primarily from transcriptomic data [22,41]. Such data sets offer unique functional *in vivo* snapshots of how genes are expressed in particular cells or under various conditions. Previously, such inference relied on microarray or bulk RNA sequencing data sets [22,34,9]. More recently, the growing accessibility of single cell RNA-sequencing data (scRNA-seq) has facilitated a higher-resolution view of expression in particular cells and cell states [25]. However, for several reasons, inferring GRNs from single cell data is still challenging [28]. First, GRN inference methods struggle with the dimensionality of the data. For data sets consisting of thousands of genes, there are potentially millions of edges to consider. Many algorithms do not scale well at this level. Second, the noise inherent in single-cell data brings further challenges to GRN inference. Among the recognized patterns of noise in single-cell data is "dropout" [7], where transcripts, disproportionately those with low or moderate expression levels, are lost from the expression counts. Other work has shown that even non-zero counts can be noisy due to technological background noise [13].

Over the years, numerous methods, ranging from Bayesian Networks (BNs), mutual information (MI), to tree-based methods, have been proposed for GRN inference. BN-based methods, such as G1DBN [17] and ebdbNet [29], model GRNs as causal inference problems in a Directed acyclic graph (DAG). MI-based methods, such as ARACNE [20], CLR [4], MRNET [23], and PIDC [1], measure the statistical dependence between two random variables and sort the edges based on these dependencies. Tree-based methods, such as GENIE3 [12], dynGENIE3 [11], and GRNBoost2 [24], rely on variable importance, a metric to rank variables while creating trees, to sort corresponding edges.

Recent advances in deep learning have offered new approaches to GRN inference. A recently published approach known as DeepSEM [35] learns the adjacency matrix by reconstructing the expression data through

a modified Variational Auto-Encoder (VAE). The authors found that DeepSEM runs efficiently and outperforms many commonly-used GRN inference methods on multiple benchmarks. Its network structure uses a parameterized adjacency matrix; an encoder, which transforms gene expression data to latent variables; and a decoder, which reconstructs the expression data from latent variables. The normality of the latent variables is enforced using the Kullback–Leibler (KL) divergence.

Yet, noise in single cell data still presents challenges. Previously, we introduced the idea of dropout augmentation (DA) and presented a novel method called DAZZLE [42]. Instead of trying to eliminate all the inflated zeros in single-cell data, DAZZLE demonstrates the advantage of training the model with added zeros and explicitly predicting the partially-augmented noise. The DA idea establishes a bridge between the "dropout" events in single-cell data and the deep learning regularization method also called "dropout" [38]. Since the augmented zeros simulate dropout events, they increase model robustness and improve the benchmark performance of GRN inference. Encouraged by our positive findings, here we further explore the value of noise injection for GRN inference methods.

One emerging methodology that has shown potentential in various domains, including image processing and time series analysis, is that of diffusion probabilistic models [10,36,30]. By simulating the process of diffusion, or the spread of information, diffusion models are likelihood-based methods that aim to restore data from Gaussian corruption by an iterative process. In Denoising Diffusion Probabilistic Models (DDPMs) [10], a probabilistic diffusion model consists of a non-parameterized forward process, which keeps adding small amounts of noise to the data for a number of steps following a pre-defined schedule, and a parameterized reverse process, which reconstructs a less-noisy version of the data. The reverse process is trained on predicting the added noise so that the de-noised data can be recovered by subtracting the predicted noise from the noisy data. Recent studies have suggested similarity between diffusion models and a generalized form of VAE with many latent spaces [19]. While a classic VAE enforces normality in the center latent space with KL divergence, diffusion models enforce a trajectory to normality by the diffusion process. In addition, some studies further connect diffusion models with score matching models via annealed Langevin dynamics sampling [37]. This finding puts the diffusion process in the framework of stochastic differential equations (SDE) and enables continuous diffusion steps. To date there are a few very recent pre-prints using diffusion models for single-cell data [40,32], but none of these considered modeling the regulatory relationships among genes.

Here, we introduce RegDiffusion, a novel diffusion probabilistic model focusing on the interactions among variables. To our knowledge, this is the first time a diffusion model has been applied for this purpose. We demonstrate that GRNs can be effectively learned using the objective of predicting added noise in a diffusion process. The network architecture of our model is surprisingly simple, yet it outperforms several benchmark GRN inference methods. Overall, RegDiffusion has the following advantages:

- RegDiffusion combines high-performance GRN inference with high stability. Compared to VAE-based deep learning models, RegDiffusion enforces a trajectory to normality by its diffusion process, which helps stabilize the learning process.
- Compared to VAE-based models, RegDiffusion only models the reverse (de-noising) process. Therefore, it avoids the costly adjacency matrix inversion step used by DAZZLE and DeepSEM. Removing this step makes it easier to learn the adjacency matrix and results in a significant speedup.
- Although presented as a deep learning method, RegDiffusion can be viewed as a form of Bayesian Networks and is based on widely-used assumptions in gene regulatory research [33].
- The success of RegDiffusion also illustrates the potential of applying diffusion models to single-cell and other kinds of noisy tabular data.

2 Methods

2.1 Problem Statement

Given a single-cell expression count table $\mathbf{X} \in \mathbb{R}^{n \times m}$, where n is the number of cells and m is the number of genes, our objective is to infer the weighted adjacency matrix of the underlying gene regulatory network $\mathbf{A} \in \mathbb{R}^{m \times m}$. It is reasonable to assume that the count table $\mathbf{X}$ reflects the typical zero-inflation and background noise affecting most single-cell expression technologies. We use $\mathbf{x} \in \mathbb{R}^m$ to denote the expression counts measured in one cell.

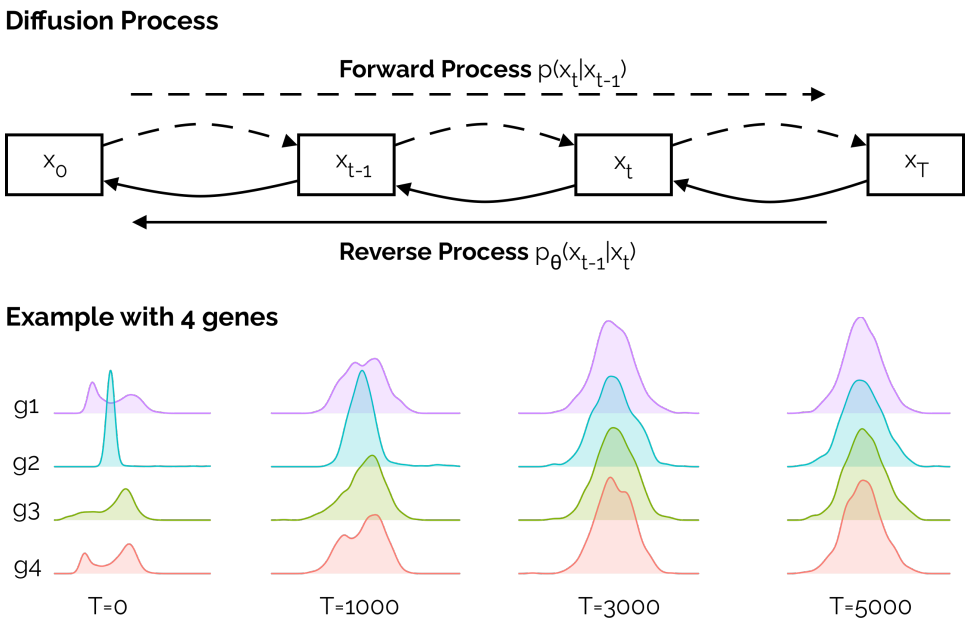

Fig. 1: The Diffusion Process is an iterative process that consists of both the forward diffusion pass and the reverse pass. As shown in this 4-gene example, each gene starts with its unique distribution over all cells. The forward process transforms all data into Gaussian noise in T steps, while the reverse process aims to recover the original distribution from the near-normal distribution.

2.2 Denoising Diffusion Probabilistic Models

Forward process As outlined in the Introduction, a DDPM consists of a forward process and a reverse process. The forward process, denoted by $q(\mathbf{x}_T|\mathbf{x}_0)$, is a non-parameterized Markov Chain that transforms an original unperturbed gene expression vector $\mathbf{x}_0$ to Gaussian noise $\mathbf{x}_T$ in T steps. This process generates a series of noisy samples $\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_T$ and as it proceeds, $\mathbf{x}$ gradually loses its distinct features as illustrated in Figure 1.

$$q(\mathbf{x}_T|\mathbf{x}_0) = \prod_{t=0}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})$$

Each step of the forward process, denoted by $q(\mathbf{x}_t|\mathbf{x}_{t-1})$, follows a noise schedule $\beta_t \in (0, 1)$. The value of β_t is usually very small. Intuitively, β_t represents the amount of information loss at each step. In the original DDPM paper, the authors proposed using the transformation function shown below [10]. This transformation uses β_t to control both the mean and variance of the sampling process. The fact that $\sqrt{1-\beta_t}$ is always less than 1 ensures that after a number of steps, the mean of the final distribution will approach 0:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I}).$$

In practice, however, it is not feasible to do iterative sampling during training. To solve this problem, the authors of DDPM proposed a reparameterization trick that enables the generation of $\mathbf{x}_t$ from $\mathbf{x}_0$, or $q(\mathbf{x}_t|\mathbf{x}_0)$, in one single step. We briefly explain this technique here, as it is crucial to our method.

Let $\alpha_t = 1 - \beta_t$ and $\overline{\alpha}_t = \prod_{i=0}^{t} \alpha_i$. The reparameterization trick generates standard Gaussian noise $\mathbf{z} = \mathcal{N}(0, 1)$. We can rewrite the transformation with the following equations:

$$
\begin{aligned}
\mathbf{x}_t &= \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\mathbf{z}_{t-1} \\
&= \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\mathbf{z}_{t-1} \\
&= \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{(1-\alpha_t)+\alpha_t(1-\alpha_{t-1})}\mathbf{z}_{t-2} \\
&= \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\mathbf{z}_{t-2} \\
&= ... \\
&= \sqrt{\overline{\alpha_t}}\mathbf{x}_0 + \sqrt{1-\overline{\alpha_t}}\mathbf{z}
\end{aligned} \tag{1}
$$

Note that here, the diffusion schedules β_t, α_t, and $\overline{\alpha_t}$ can all be pre-generated. Therefore, during the training process, we only need the unperturbed data $\mathbf{x}_0$ and a randomly sampled diffusion time step t to generate the perturbed data $\mathbf{x}_t$ at time step t.

Reverse Process Recall that the reverse process is a Markov Chain aiming to recover the original input $\mathbf{x}_0$ from $\mathbf{x}_T$. Since computing the exact reverse step $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ is intractable, we use a parameterized model $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ to approximate this process.

$$p_\theta(\mathbf{x}_0) = p(\mathbf{x}_T)\prod_{t=T}^{0} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$$

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(x_{t-1}; \mu_\theta(\mathbf{x}_t, t), \sigma_\theta(\mathbf{x}_t, t))$$

For simplicity, we chose to use the simplified loss function proposed in the DDPM paper:

$$L_{simple}(\theta) = \mathbb{E}_{\mathbf{x}_0,t,\mathbf{z}}\left[||\mathbf{z} - \mathbf{z}_\theta(\sqrt{\overline{\alpha_t}}\mathbf{x}_0 + \sqrt{1-\overline{\alpha_t}}\mathbf{z}, t)||^2\right] \tag{2}$$

In other words, we need to create a model to estimate the added noise $\mathbf{z}$ used in the forward pass (Eq. 1). The input of the model should be the perturbed data $\mathbf{x}_t$ and the time step t. But how can we predict the error added to a single-cell expression count table while inferring the adjacency matrix? In computer vision, a common practice is to use image segmentation models, such as U-Net [31], to 'segment' the noise. Here, we introduce an analogous approach for single cell data.

2.3 RegDiffusion

Linear Additive Assumption Our proposed RegDiffusion method relies on the linear additive assumption, which is commonly used in many GRN inference methods, especially in those based on Bayesian Networks [5,33]. This assumption assumes the log probability of a particular gene is the linear combination of all of the regulator genes.

$$\mathbf{X} = \mathbf{X}\mathbf{A}$$

We further assume that the Gaussian perturbed expression can be expressed in a similar form:

$$\mathbf{X}_t = \mathbf{X}_t\mathbf{A} + \lambda_t\mathbf{Z},$$

where $\mathbf{Z}$ is the Gaussian noise and λ_t is some constant for time step t. This equation can be easily transformed into a conceptual equation to estimate $\mathbf{Z}$:

$$
\begin{aligned}
\mathbf{Z} &\propto \mathbf{X}_t - \mathbf{X}_t\mathbf{A} \\
&\propto \mathbf{X}_t(\mathbf{I} - \mathbf{A})
\end{aligned} \tag{3}
$$

Combining the concept in Equation 3 and the loss function from Equation 2 leads to a very straightforward neural network design, as shown in Figure 2. We will explain each section in the following paragraphs.

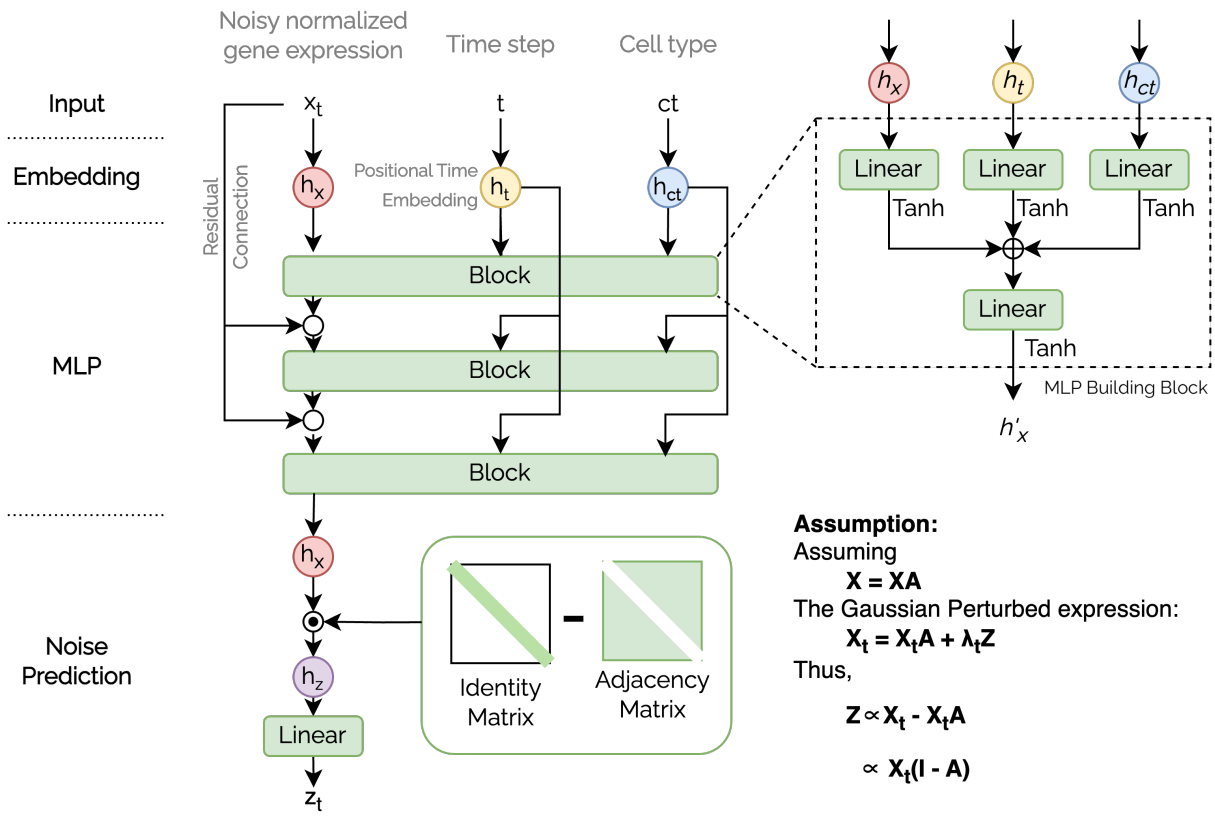

Fig. 2: Graphic illustration of the RegDiffusion process proposed in this section.

Data Preprocessing and Normalization RegDiffusion requires a single cell count table as input. We suggest users first follow common data preprocessing steps to address their specific needs. In experiments described here, we simply transform the data using the log2(count-plus-one) transformation and apply a standard quality assurance workflow (removing cells with high mitochondrial content or low UMI or gene counts). There is no need to restrict consideration to only the most variable genes, because RegDiffusion with all detected genes runs fairly quickly on modern GPUs.

We first normalize the expression table by cells using min-max normalization to balance inter-cell differences. Then, we z-score normalize the genes.

Embedding As shown in Figure 2, our model starts with an embedding process that transform the inputting gene expression vector x_t, diffusion time step t, and (optionally designated) cell type ct (optional) to corresponding embeddings h_x, h_t, and h_{ct}. For each cell, the gene embedding h_x is a 2D array in the shape of $\mathbb{R}^{m \times d_{gene}}$, where m is the number of genes and d_{gene} is the size of gene embedding. The first column is always the normalized expression values. The other columns are trainable gene features that are shared across cells. Next, following the standard practice for Diffusion Probabilistic Models, we use a Sinusoidal Positional Embedding to embed the time steps t. This process transforms an ordinal time step to a meaningful vector $h_t \in \mathbb{R}^{d_{time}}$ using sine and cosine transformation. Finally, the cell type embedding h_{ct} is anther 2D array in the shape of $\mathbb{R}^{c \times d_{celltype}}$, where c is the number of cell types and $d_{celltype}$ is the dimension of the cell type embedding. The provided cell type index is used to pull up the corresponding cell type feature during training.

Feature Learning MLPs In RegDiffusion, we use a set of multilayer perceptrons (MLPs) to combine the features of gene, time step, and cell types. Ultimately, the goal is to create a feature matrix h_x ready to be

used for noise prediction, as in Equation 3. To achieve this goal, we design an MLP building block as shown in Figure 2. This building block consumes the previous gene feature h_x^{l-1}, the time step embedding h_t^0, and the cell type embedding h_{ct}^0. These features are converted into the same dimension using three separated linear models and the activated values are summed together followed by another linear layer to build the new gene feature h_x^l.

Adjacency Matrix and Noise Prediction The adjacency matrix in RegDiffusion is a large set of trainable model parameters (m^2, where m is the number of genes). During the training process, the diagonal entries of the adjacency matrix are set to zeros to discourage self-regulation. To increase the sparsity of the adjacency matrix, we applied L1 loss on the matrix. To prevent overfitting and increase model robustness, we also apply dropout to the adjacency matrix; specifically, a proportion of edge weights are set to 0 during training.

Following equation 3, with the learned feature matrix h_x coming out of the stacked blocks and the adjacency matrix, it is easy to calculate the feature of the added noise h_z using matrix multiplication. This feature is then converted into the prediction of 'z' using a linear layer.

Top Edge Extraction GRN inference is complete once the model converges, since we can simply extract the edge weights from the adjacency matrix. The top predicted edges are those with the largest absolute predicted edge weights. For very large adjacency matrices, it is recommended to setup a cutoff threshold to speed up the sorting process.

3 Experiments and Results

3.1 Experiment Setup

The BEELINE single-cell benchmark [28] comprises 7 distinct preprocessed single-cell RNAseq datasets from both human (hESC, hHep) and mouse (mDC, mESC, mHSC-E, mHSC-GM, mHSC-L), offering a broad scope for evaluation. With the provided benchmark-generation scripts in BEELINE, users can specify the number of target genes and BEELINE selects the most varied genes based on the variance and p values returned by a fitted generalized additive model (GAM). BEELINE also comes with three sets of ground truth networks, namely the STRING v11 database [39], non-cell-type specific transcriptional networks, and cell-type specific networks. The non-cell-type specific network combines ChipSeq-based data collected from DoRothEA[6], RegNetwork[18], and TRRUST v2[8], while the cell-type specific networks were experimentally generated by the BEELINE authors. Here, we use the STRING network as the ground truth and set the number of target genes to be 1,000. The exact numbers of genes/cells/true edges for each dataset are reported in Table 1. Our results on non-cell-type specific networks, yielding similar findings, are included in the Supplement.

We compared the performance of RegDiffusion with 4 baseline methods, including GENIE3, GRNBoost2, DeepSEM, and our previous proposed work DAZZLE. GENIE3 and GRNBoost2 are two tree-based machine learning methods that have been found to perform well on single cell data according to recent benchmarks [14,28]. Particularly, as previously reported, with the same BEELINE benchmark we use, they are among the best three methods (together with PIDC [1]) and have been found to outperform SCODE [21], SINCERITIES [26] and PPCOR [16]. DeepSEM and DAZZLE are both variational autoencoder based neural networks. For GENIE3 and GRNBoost2, we used the implementation from the Python package arboreto v0.1.6 [24]. For DeepSEM and DAZZLE, we used the implementation from the Python package GRN-dazzle v0.0.2 [42]. Note that as described by the authors, DeepSEM is an ensemble algorithm which combines results from 10 runs. Because of its speed, we include the single-run version, denoted by DeepSEM*, in our comparison as well.

Following the BEELINE paper, we use Area Under the Precision Recall Curve (AUPRC) *Ratio* as the main evaluation metric. AUPRC Ratio is simply the AUPRC score divided by that of a random predictor. Since the number of all possible edges and the number of true edges are usually very large in actual GRNs, the values of the AUPRC itself tend to be small. Converting them to ratios make it easier to understand the performance across benchmarks. We also report the ratio value of Early Precision (EP), which is in fact top-k Precision where k is the number of true edges. Our results for the EP Ratio are included in the Supplement.

	hESC	hHep	mDC	mESC	mHSC-E	mHSC-GM	mHSC-L
# of Genes	1410	1448	1321	1620	1204	1132	692
# of Cells	758	425	383	421	1071	889	847
# of True Edges	5,149	9,000	5,898	8,479	1,826	1,311	154
GENIE3	1.98 (0.01)	1.86 (0.01)	1.72 (0.01)	2.05 (0.01)	4.19 (0.05)	6.27 (0.04)	7.02 (0.08)
GRNBoost2	1.67 (0.01)	1.50 (0.01)	1.43 (0.01)	1.88 (0.02)	3.65 (0.05)	5.16 (0.11)	7.07 (0.08)
DeepSEM* (single-run)	1.95 (0.08)	1.66 (0.10)	1.57 (0.05)	2.20 (0.03)	5.00 (0.49)	5.83 (0.81)	6.69 (1.81)
DeepSEM (ensemble)	2.11 (0.02)	1.85 (0.02)	1.67 (0.02)	2.31 (0.03)	5.67 (0.13)	6.76 (0.17)	7.50 (0.12)
DAZZLE	2.50 (0.05)	1.87 (0.02)	1.60 (0.02)	2.26 (0.06)	6.02 (0.15)	6.39 (0.16)	7.65 (0.12)
RegDiffusion	2.40 (0.02)	2.09 (0.01)	1.63 (0.01)	2.47 (0.02)	6.19 (0.06)	7.21 (0.06)	7.21 (0.08)

Table 1: RegDiffusion outperforms most GRN inference methods on BEELINE benchmark datasets (Metric: AUPRC Ratio; Ground truth: STRING; Number of target genes: 1000). Numbers reported are mean and std. of AUPRC Ratio compared with random guess over 10 runs. Higher ratios indicate better performance. Here, Red and orange text indicate the best and 2nd best algorithms. In this experiment, we executed each algorithm on all available data with the default settings.

3.2 RegDiffusion outperforms other GRN methods on BEELINE benchmarks

Table 1 presents a performance comparison of AUPRC ratios for several GRN inference methods on the BEELINE benchmark data sets. The seven data sets differ in terms of the number of cells and the number of edges in the ground truth networks. They tend to have similar numbers of genes since we selected the most varying 1,000 target genes.

The proposed RegDiffusion method generally outperforms most other GRN inference techniques across the different datasets, as indicated by higher AUPRC Ratios. It achieves top results for the hHep, mESC, mHSC-E, and mHSC-GM data sets, and it is either the second or third best for the others. DAZZLE, DeepSEM (ensemble), and GENIE3 come after. Furthermore, the performance of RegDiffusion is very stable across runs, as indicated by the small standard deviations of AUPRC Ratios in 10 repeated runs. It has even better stability than the ensemble version of DeepSEM and is almost as stable as GENIE3 and GRNBoost2.

3.3 Runtime Analysis

	hESC	hHep	mDC	mESC	mHSC-E	mHSC-GM	mHSC-L
# of Genes	1410	1448	1321	1620	1204	1132	692
# of Cells	758	425	383	421	1071	889	847
GENIE3	6,462	2,931	2,709	4,173	7,723	5,674	2,493
GRNBoost2	2,055	992	962	2,171	2,442	1,382	379
DeepSEM* (single run)	29	15	12	17	30	23	15
DeepSEM (ensemble)	251	147	116	168	298	229	139
DAZZLE	31	16	13	18	34	27	19
RegDiffusion	10	5	3	5	13	10	9

Table 2: Average algorithm execution time in seconds over 10 runs. All code was executed on HPC computing nodes with 4 cores and 12G memory. DeepSEM, DeepSEM*, DAZZLE, and RegDiffusion are executed with an additional NVIDIA A100 card.

Table 2 summarizes the average execution time in seconds needed to run the benchmark experiments above. All computations were completed with the same number of CPU cores and the same memory, but the deep learning algorithms are executed with one additional NVIDIA A100 card. With accelerated computing,

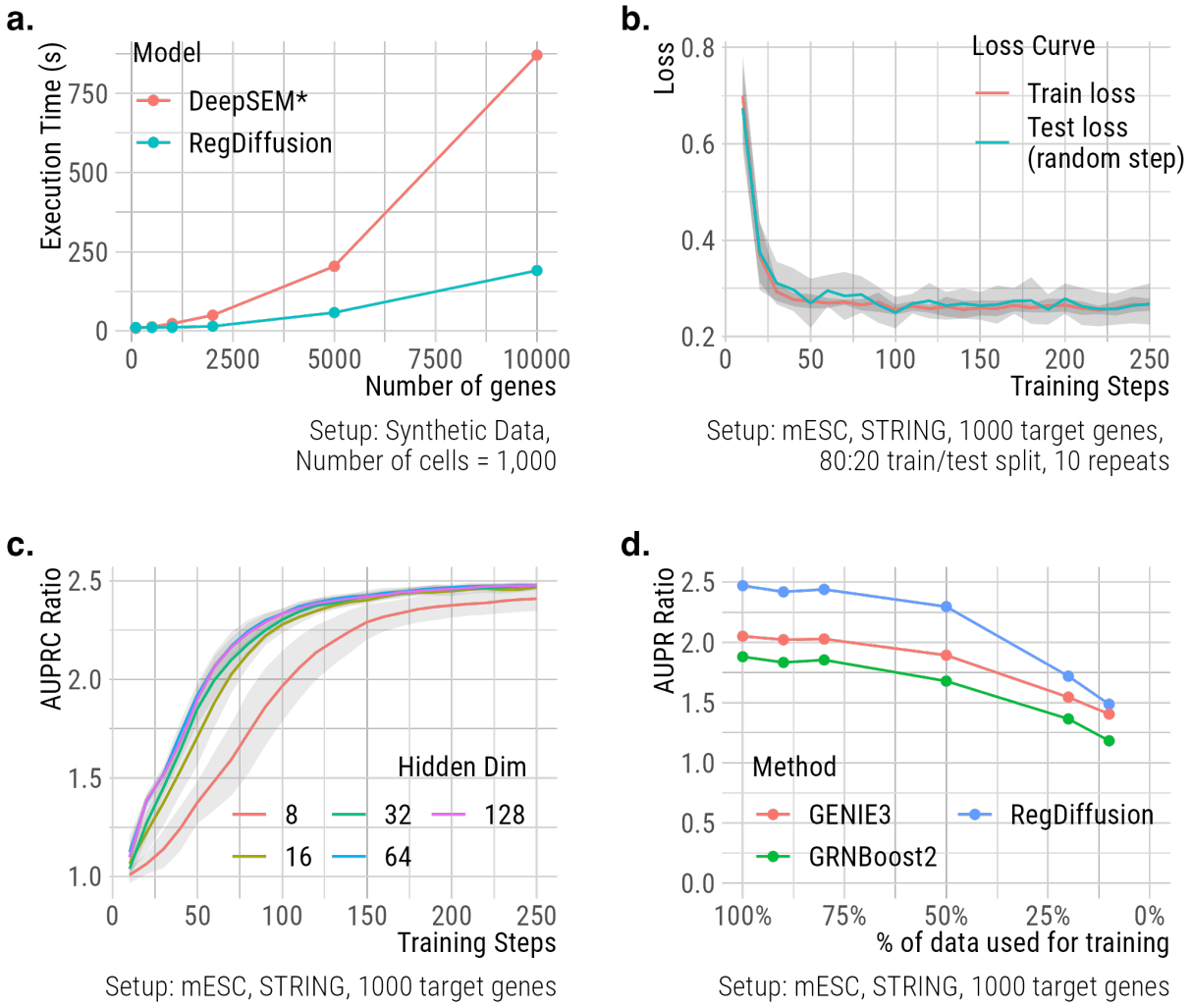

Fig. 3: Properties of RegDiffusion. **a.** Execution time of RegDiffusion scales much more gradually with the number of genes than DeepSEM* on a synthetic dataset with 1,000 cells; **b.** Loss curves on train/test splits (4:1) show that the model was not overfitted (grey areas represent standard deviations over 10 runs); **c.** Model performance is rather stable regardless of hidden dimensions in the neural networks; **d.** Downsampling shows that RegDiffusion performs better than bulk RNAseq methods even when sample size is limited.

RegDiffusion completes GRN inference in a fraction of the time required by all the comparator methods. For example, for the mDC data, while GENIE3 takes 45 minutes, RegDiffusion runs in 3.2 seconds.

Since all of these algorithms are estimating an $m \times m$ adjacency matrix at some point, where m is the number of genes, they should all run in $O(m^2)$ time complexity with respect to the number of genes. However, as shown in Table 2, when the number of genes is below 1,500, the impact of the number of genes is almost unnoticeable for RegDiffusion. For example, while the number of genes in hESC is more than two times of that in mHSC-L, RegDiffusion finishes inferences in similar time on these two datasets. We further analyze this behavior by comparing RegDiffusion with the second fastest algorithm DeepSEM* using a fully synthetic data set with 1,000 cells and increasing numbers of genes. As shown in Figure 3a, although both RegDiffusion and DeepSEM* require super-linear time, the scaling constant for RegDiffusion is much lower. The reason why we can get such a speed gain is that both DeepSEM* and DAZZLE rely on a VAE model that does an expensive matrix inversion. In RegDiffusion, the elimination of the matrix inversion step significantly brings down the calculation time, especially when the number of genes is large.

3.4 Model Tuning

Since RegDiffusion is trained only on the objective of predicting added noise, the ground truth GRN network never leaks into the training process. Still, since RegDiffusion is a deep neural network with a lot of parameters, we would like to ensure the model is not overfitted. We therefore apply a train/test split on cells, allowing adjustment of model hyperparameters if necessary. Figure 3b illustrates the training curves on the typical mESC data set when we applied a 4:1 train/test split. As shown in the figure, the loss curves for the train and test splits are never separated too much, showing that the model was not overfitted.

At the same time, we would like to reduce the number of parameters in the model as much as possible. Overall, we found that with reasonable number of model parameters, the performance of RegDiffusion is rather stable across different model sizes. As shown in Figure 3c, when we adjust the dimension of the hidden layer in the MLP building block, the performance of the model doesn't change too much except when the dimension is 8, which we think is too small. We thus set the default size of the hidden layer to 16 for all data sets.

3.5 Impact of Training Size

One question coming out of the train/test split experiment above is how the models would respond to different training sizes. In other words, do methods that were originally proposed for bulk RNAseq data, such as GENIE3 and GRNboost2, perform better when the number of cells is limited? To answer this question, we down-sampled the training data. When we generated the partition, we used the same randomization seed to ensure all models were seeing the same data. As shown in Figure 3d, we found that the performance at convergence of all three models (RegDiffusion, GENIE3, and GRNBoost2) dropped as training size decreased and RegDiffusion consistently performed better than the others for all partitions. Also, we noticed that AUPR Ratios in all cases drops back towards 1.0 (no better than random guessing) following a very similar pattern. For example, we can see a dip in all curves when we use 90% of the data for training, suggesting an artifact perhaps due to a bad data partition that is impacting all models equally.

4 Conclusion

We have presented RegDiffusion, a novel diffusion probabilistic model designed to infer GRNs from single-cell RNAseq data. RegDiffusion simulates the information diffusion process by iteratively adding Gaussian noise following a fixed schedule. Based on the benchmarking results, compared to other top methods, RegDiffusion demonstrates consistent superiority in GRN inference capacity while remain stable across runs. It also achieves a significant runtime reduction over comparable methods. Even with limited data, RegDiffusion still outperforms competitors.

The success of RegDiffusion in GRN inference also suggests several potential future research paths. First, it would be very interesting to explore the effects of RegDiffusion and its learned embeddings on other single-cell analysis tasks. Also, to our knowledge, RegDiffusion is the first diffusion model working on tabular data for causal inference. It would be interesting to see if it can be extended as a general causal inference method. Another important future research direction is understanding the inferred networks. It has become much easier and faster today to infer large gene regulatory networks, but the difficulties of understanding 100 links and 100 million links and not at all comparable. It is crucial to discover more sophisticated ways to analyze these large networks under various experimental conditions. Finally, it would be helpful to find a solution to learn detailed features in single-cell data given the existence of background noise. In our experiments, we observed that the models are good at learning crude features (at high diffusion steps) while having some difficulties at learning fine features at low diffusion steps, a phenomenon that has been observed many times in other diffustion models.

Diffusion models in general have sparked a lot of research interest in recent years thanks to their huge success in computer vision. Together with a few recent pre-prints in 2023 [40,32], RegDiffusion presents as one of the initial attempts to solve the problems in single-cell data with diffusion models and is the first one that considers gene regulatory relationships. We anticipate that RegDiffusion many not only offer a reliable and convenient tool for GRN inference, but may also advance our understanding of regulatory genomics and single-cell analysis.

References

1. Chan, T.E., Stumpf, M.P., Babtie, A.C.: Gene regulatory network inference from single-cell data using multivariate information measures. Cell systems **5**(3), 251–267 (2017)
2. Davidson, E., Levin, M.: Gene regulatory networks. Proceedings of the National Academy of Sciences **102**(14), 4935–4935 (2005)
3. Emmert-Streib, F., Dehmer, M., Haibe-Kains, B.: Gene regulatory networks and their applications: understanding biological and medical problems in terms of networks. Frontiers in cell and developmental biology **2**, 38 (2014)
4. Faith, J.J., Hayete, B., Thaden, J.T., Mogno, I., Wierzbowski, J., Cottarel, G., Kasif, S., Collins, J.J., Gardner, T.S.: Large-scale mapping and validation of escherichia coli transcriptional regulation from a compendium of expression profiles. PLoS biology **5**(1), e8 (2007)
5. Friedman, N., Linial, M., Nachman, I., Pe'er, D.: Using bayesian networks to analyze expression data. In: Proceedings of the fourth annual international conference on Computational molecular biology. pp. 127–135 (2000)
6. Garcia-Alonso, L., Holland, C.H., Ibrahim, M.M., Turei, D., Saez-Rodriguez, J.: Benchmark and integration of resources for the estimation of human transcription factor activities. Genome research **29**(8), 1363–1375 (2019)
7. Ghazanfar, S., Bisogni, A.J., Ormerod, J.T., Lin, D.M., Yang, J.Y.: Integrated single cell data analysis reveals cell specific networks and novel coactivation markers. BMC systems biology **10**(5), 11–24 (2016)
8. Han, H., Cho, J.W., Lee, S., Yun, A., Kim, H., Bae, D., Yang, S., Kim, C.Y., Lee, M., Kim, E., et al.: Trrust v2: an expanded reference database of human and mouse transcriptional regulatory interactions. Nucleic acids research **46**(D1), D380–D386 (2018)
9. Hecker, M., Lambeck, S., Toepfer, S., Van Someren, E., Guthke, R.: Gene regulatory network inference: data integration in dynamic models—a review. Biosystems **96**(1), 86–103 (2009)
10. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. Advances in neural information processing systems **33**, 6840–6851 (2020)
11. Huynh-Thu, V.A., Geurts, P.: dyngenie3: dynamical genie3 for the inference of gene networks from time series expression data. Scientific reports **8**(1), 3384 (2018)
12. Huynh-Thu, V.A., Irrthum, A., Wehenkel, L., Geurts, P.: Inferring regulatory networks from expression data using tree-based methods. PloS one **5**(9), e12776 (2010)
13. Janssen, P., Kliesmete, Z., Vieth, B., Adiconis, X., Simmons, S., Marshall, J., McCabe, C., Heyn, H., Levin, J.Z., Enard, W., et al.: The effect of background noise and its removal on the analysis of single-cell expression data. Genome Biology **24**(1), 140 (2023)
14. Kang, Y., Thieffry, D., Cantini, L.: Evaluating the reproducibility of single-cell gene regulatory network inference algorithms. Frontiers in genetics **12**, 362 (2021)
15. Karlebach, G., Shamir, R.: Modelling and analysis of gene regulatory networks. Nature reviews Molecular cell biology **9**(10), 770–780 (2008)
16. Kim, S.: ppcor: an r package for a fast calculation to semi-partial correlation coefficients. Communications for statistical applications and methods **22**(6), 665 (2015)
17. Lèbre, S.: Inferring dynamic genetic networks with low order independencies. Statistical applications in genetics and molecular biology **8**(1) (2009)
18. Liu, Z.P., Wu, C., Miao, H., Wu, H.: Regnetwork: an integrated database of transcriptional and post-transcriptional regulatory networks in human and mouse. Database **2015** (2015)
19. Luo, C.: Understanding diffusion models: A unified perspective. arXiv preprint arXiv:2208.11970 (2022)
20. Margolin, A.A., Nemenman, I., Basso, K., Wiggins, C., Stolovitzky, G., Favera, R.D., Califano, A.: Aracne: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. In: BMC bioinformatics. vol. 7, pp. 1–15. BioMed Central (2006)
21. Matsumoto, H., Kiryu, H., Furusawa, C., Ko, M.S., Ko, S.B., Gouda, N., Hayashi, T., Nikaido, I.: Scode: an efficient regulatory network inference algorithm from single-cell rna-seq during differentiation. Bioinformatics **33**(15), 2314–2321 (2017)
22. Mercatelli, D., Scalambra, L., Triboli, L., Ray, F., Giorgi, F.M.: Gene regulatory network inference resources: A practical overview. Biochimica et Biophysica Acta (BBA)-Gene Regulatory Mechanisms **1863**(6), 194430 (2020)
23. Meyer, P.E., Kontos, K., Lafitte, F., Bontempi, G.: Information-theoretic inference of large transcriptional regulatory networks. EURASIP journal on bioinformatics and systems biology **2007**, 1–9 (2007)
24. Moerman, T., Aibar Santos, S., Bravo González-Blas, C., Simm, J., Moreau, Y., Aerts, J., Aerts, S.: Grnboost2 and arboreto: efficient and scalable inference of gene regulatory networks. Bioinformatics **35**(12), 2159–2161 (2019)
25. Nguyen, H., Tran, D., Tran, B., Pehlivan, B., Nguyen, T.: A comprehensive survey of regulatory network inference methods using single cell rna sequencing data. Briefings in bioinformatics **22**(3), bbaa190 (2021)
26. Papili Gao, N., Ud-Dean, S.M., Gandrillon, O., Gunawan, R.: Sincerities: inferring gene regulatory networks from time-stamped single cell transcriptional expression profiles. Bioinformatics **34**(2), 258–266 (2018)

27. Penfold, C.A., Wild, D.L.: How to infer gene networks from expression profiles, revisited. Interface focus **1**(6), 857–870 (2011)
28. 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**(2), 147–154 (2020)
29. Rau, A., Jaffrézic, F., Foulley, J.L., Doerge, R.W.: An empirical bayesian method for estimating biological networks from temporal microarray data. Statistical Applications in Genetics and Molecular Biology **9**(1) (2010)
30. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10684–10695 (2022)
31. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18. pp. 234–241. Springer (2015)
32. Sadria, M., Layton, A.: The power of two: integrating deep diffusion models and variational autoencoders for single-cell transcriptomics analysis. bioRxiv pp. 2023–04 (2023)
33. Sanchez-Castillo, M., Blanco, D., Tienda-Luna, I.M., Carrion, M., Huang, Y.: A bayesian framework for the inference of gene regulatory networks from time and pseudo-time series data. Bioinformatics **34**(6), 964–970 (2018)
34. Schaffter, T., Marbach, D., Floreano, D.: Genenetweaver: in silico benchmark generation and performance profiling of network inference methods. Bioinformatics **27**(16), 2263–2270 (2011)
35. Shu, H., Zhou, J., Lian, Q., Li, H., Zhao, D., Zeng, J., Ma, J.: Modeling gene regulatory networks using neural network architectures. Nature Computational Science **1**(7), 491–501 (2021)
36. Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., Ganguli, S.: Deep unsupervised learning using nonequilibrium thermodynamics. In: International conference on machine learning. pp. 2256–2265. PMLR (2015)
37. Song, Y., Durkan, C., Murray, I., Ermon, S.: Maximum likelihood training of score-based diffusion models. Advances in Neural Information Processing Systems **34**, 1415–1428 (2021)
38. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research **15**(1), 1929–1958 (2014)
39. Szklarczyk, D., Gable, A.L., Lyon, D., Junge, A., Wyder, S., Huerta-Cepas, J., Simonovic, M., Doncheva, N.T., Morris, J.H., Bork, P., et al.: String v11: protein–protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets. Nucleic acids research **47**(D1), D607–D613 (2019)
40. Tang, W., Liu, R., Wen, H., Dai, X., Ding, J., Li, H., Fan, W., Xie, Y., Tang, J.: A general single-cell analysis framework via conditional diffusion generative models. bioRxiv pp. 2023–10 (2023)
41. Van Dam, S., Vosa, U., van der Graaf, A., Franke, L., de Magalhaes, J.P.: Gene co-expression analysis for functional classification and gene–disease predictions. Briefings in bioinformatics **19**(4), 575–592 (2018)
42. Zhu, H., Slonim, D.: Improving gene regulatory network inference using dropout augmentation. bioRxiv pp. 2023–01 (2023)