TRENDY: Gene Regulatory Network Inference Enhanced by Transformer

Xueying Tian[1], Yash Patel[2], and Yue Wang[3,*]

[1]School of Information, University of California, Berkeley, CA 94720, USA
[2]Department of Mathematics, University of Miami, Coral Gables, FL 33146, USA
[3]Irving Institute for Cancer Dynamics and Department of Statistics, Columbia University, New York, NY 10027, USA
[*]Corresponding author: yw4241@columbia.edu

Abstract

Gene regulatory networks (GRNs) play a crucial role in the control of cellular functions. Numerous methods have been developed to infer GRNs from gene expression data, including mechanism-based approaches, information-based approaches, and more recent deep learning techniques, the last of which often overlooks the underlying gene expression mechanisms. In this work, we introduce TRENDY, a novel GRN inference method that integrates transformer models to enhance the mechanism-based WENDY approach. Through testing on both simulated and experimental datasets, TRENDY demonstrates superior performance compared to existing methods. Furthermore, we apply this transformer-based approach to three additional inference methods, showcasing its broad potential to enhance GRN inference.

1 Introduction

The expression of genes can be regulated by other genes. We can construct a directed graph, where vertices are genes, and edges are regulatory relationships. This graph is called a gene regulatory network (GRN). GRNs are essential to understanding the mechanisms governing cellular processes, including how cells respond to various stimuli, differentiate, and maintain homeostasis [1, 2]. Understanding the GRN structure is important for developmental biology [3, 4, 5, 6], disease research [7, 8, 9], and even studying macroscopic behavior [10, 11, 12].

It is difficult to directly determine the GRN structure with experiments. Instead, researchers develop methods that infer the GRN structure through gene

expression data. Some methods, such as WENDY [13] and NonlinearODEs [14], are mechanism-based: they construct dynamical models for gene expression, and fit with expression data to determine the GRN [15, 16]. Some methods, like GENIE3 [17] and SINCERITIES [18], are information-based: they directly find genes that contain information (predictability) for the target gene, and assume that such information implies regulation [19, 20].

Recently, there are some deep learning-based methods that infer GRN with different neural network frameworks, such as convolutional neural networks [21], recurrent neural networks [22], variational autoencoders [23], graph neural networks [24, 25], etc. Deep learning frameworks can perform well on many tasks. However, because of the large number of tunable parameters in a neural network, it is difficult to explain why it works. These deep learning-baed GRN inference methods have similar problems: they generally apply popular neural networks as black boxes without integration with biological mechanisms, and thus lacking interpretability.

The transformer model is a deep learning architecture designed to handle sequential data [26]. It can capture long-range dependencies and relationships through the self-attention mechanism. The encoder layer of the transformer model can process the input through a neural network, so that the output is close to the given target.

Some researchers have developed GRN inference methods based on the transformer model [27] or its variants [28]. The STGRNS method [27] predicts the existence of regulation between two genes solely based on the expression levels of these two genes. Since other genes are not considered, this approach makes it difficult to distinguish between direct regulation ($G_i \to G_j$) and indirect regulation through a third gene ($G_i \to G_k \to G_j$). The GRN-transformer method [28] uses multiple inputs, including the GRN inferred by an information-based method, PIDC [29], but it also does not consider the biological dynamics of gene regulation.

To combine the powerful deep learning techniques and biological understanding of gene regulation, we propose a novel GRN inference method that applies transformer models to enhance a mechanism-based GRN inference method, WENDY. This new method, TRansformer-Enhanced weNDY, is abbreviated as TRENDY. WENDY method is based on a dynamical model for the covariance matrix of genes. In TRENDY, we first use a transformer model to construct a pseudo-covariance matrix that performs better in WENDY. Then we apply another transformer model that directly enhances the inferred GRN.

The idea for the second half of TRENDY can be used to enhance the inferred results by other GRN inference methods. We apply transformer models to three GRN inference methods, GENIE3, SINCERITIES, and NonlinearODEs, to obtain their enhanced versions, GENIE3-rev, SINCERITIES-rev, and NonlinearODEs-rev.

We test four transformer-enhanced methods against their non-transformer counterparts on two simulated data sets and two experimental data sets. All four transformer-enhanced methods outperform existing methods, and TRENDY ranks the first among all methods.

In order to train a deep learning model, such as the transformer model used in this paper, we need a sufficient amount of training data. Specifically, since we want the GRN inference method to work well in different situations, the training data should contain many different data sets that correspond to different GRNs. In reality, there are not many available experimental gene expression data sets with known GRNs. Therefore, we consider generating synthetic data sets.

To simulate new gene expression data, the most common approach is to assume that gene expression follows a certain mechanism, such as an ordinary differential equation (ODE) system [30, 31] or a stochastic differential equation (SDE) system [32, 33]. After determining the parameters of this system, one can simulate a discretized version of this system and obtain new data. Readers may refer to a review [34] for this mechanism approach. Another approach is to apply deep learning to learn from experimental gene expression data and generate new data that mimic the existing data. Such generative neural networks can be variational autoencoders [23] or generative adversarial networks [35].

The advantage of the differential equation approach is that one can change the parameters to generate new data that correspond to another GRN, while the deep learning approach can only reproduce existing data. The disadvantage of the differential equation approach is that the modeled mechanism might not be a perfect fit to reality. Therefore, the generated data might be questionable, especially those generated by an oversimplified model, such as a linear ODE system. Instead, the deep learning approach can generate new data that are indistinguishable from the experimental data.

In this paper, we need to feed the transformer models with data from many different GRNs. Therefore, we adopt the mechanism approach with a frequently used nonlinear and stochastic system [36, 18, 13]:

$$\mathrm{d}X_j(t) = V\left\{\beta\prod_{i=1}^{n}\left[1+(A_{\text{true}})_{i,j}\frac{X_i(t)}{X_i(t)+1}\right]-\theta X_j(t)\right\}\mathrm{d}t+\sigma X_j(t)\mathrm{d}W_j(t). \tag{1}$$

The matrix A_{true} is a randomly generated ground truth GRN, where $(A_{\text{true}})_{i,j} > 0/ = 0/ < 0$ means that gene i has positive/no/negative regulatory effects on gene j. $X_i(t)$ is the expression level of gene i at time t, and $W_j(t)$ is a standard Brownian motion, independent with other $W_i(\tau)$. The parameter values are $V = 30$, $\beta = 1$, $\theta = 0.2$, and $\sigma = 0.1$. Our goal is to infer A_{true} from $X_j(t)$.

Most deep learning-based GRN inference methods are trained on experimental data sets, whose amount is very limited. Therefore, such methods need to carefully choose the neural network structure and the training procedure under this limitation. In this paper, we consider a scenario in which the training data set is sufficiently large, and study how deep learning can achieve the best performance without data limitation.

2 Results

2.1 TRENDY method

WENDY method [13] uses single-cell gene expression data measured at two time points, each of size $m \times n$ (mRNA counts of n genes for m cells). Here we do not know how cells at different time points correspond to each other, since the measurement of single-cell level gene expression needs to kill the cells [37]. For data at two time points, the cell numbers m can be different.

For gene expression data at time points 0 and t, WENDY applies graphical lasso [38] to calculate the covariance matrices for different genes, K_0 and K_t, each of size $n \times n$. We can construct a general SDE system like Eq. 1 to model the dynamics of gene regulation. After linearization of this system, one can obtain an approximated relation for K_0, K_t, and the ground truth GRN A_{true}, also of size $n \times n$:

$$K_t = (I + tA_{\text{true}}^{\text{T}})K_0(I + tA_{\text{true}}) + D + E. \tag{2}$$

Here I is an $n \times n$ identity matrix, D is an unknown $n \times n$ diagonal matrix, and E is the unknown error introduced by linearization. Then WENDY solves the GRN matrix A as a non-convex optimization problem:

$$\arg\min_A f(A) := \frac{1}{2}\sum_{i \neq j}\{[K_t - (I + tA^{\text{T}})K_0(I + tA)]_{i,j}\}^2. \tag{3}$$

Due to the unknown diagonal matrix D, this summation does not count elements with $i = j$. The result of this optimization problem is denoted as

$$A_0 = \text{WENDY}(K_0, K_t).$$

If WENDY works perfectly, then

$$K_t = (I + tA_0^{\text{T}})K_0(I + tA_0) + D,$$

meaning that $(I+tA_{\text{true}}^{\text{T}})K_0(I+tA_{\text{true}})$ and $(I+tA_0^{\text{T}})K_0(I+tA_0)$ differ by E, the error introduced by the linearization, and A_0 may not match A_{true} accurately. If we want to derive a more accurate and complicated equation of K_0, K_t, and A_{true} than Eq. 2, this equation might be numerically unstable to solve. Therefore, we need to find another way to improve the WENDY method.

Define

$$K_t^* = (I + tA_{\text{true}}^{\text{T}})K_0(I + tA_{\text{true}}). \tag{4}$$

If we replace K_t by K_t^* in Eq. 3 and solve A, then this inferred A should be very close to A_{true}. However, in practice, we only have K_0 and K_t, but not K_t^*, since A_{true} is unknown.

The core idea is to train a transformer model TE($k = 1$) with input K_t, so that the output K_t' is close to K_t^*. Then we can calculate

$$A_1 = \text{WENDY}(K_0, K_t'),$$

which is closer to A_{true} than A_0. We can further improve A_1 using another transformer model TE($k = 3$), along with the information in K_0, K_t, so that the output A_2 is even closer to A_{true}. These two transformer models will be discussed in the next subsection.

Fig. 1 describes the training and testing workflow for TRENDY. To train TRENDY, one needs to
(1) generate many random A_{true} and their corresponding gene expression data;
(2) calculate the covariance matrices K_0 and K_t, and then calculate K_t^* from $K_0, K_t, A_{\text{true}}$;
(3) use K_t as input and K_t^* as target to train the TE($k = 1$) model, and then use the trained TE($k = 1$) model to calculate K_t' from K_t;
(4) use WENDY to calculate A_1 from K_0, K_t';
(5) use A_1, K_0, K_t as input and A_{true} as target to train the TE($k = 3$) model.

For applying TRENDY to infer GRN from existing data, one needs to omit step (1) and
(2) calculate the covariance matrices K_0 and K_t;
(3) use the trained TE($k = 1$) model to calculate K_t' from K_t;
(4) use WENDY to calculate A_1 from K_0, K_t';
(5) feed A_1, K_0, K_t to the trained TE($k = 3$) model to calculate the final inferred GRN A_2, the final output of TRENDY.

See https://github.com/YueWangMathbio/TRENDY for all code files and data used in this paper.

2.2 Transformer model TE(k)

We build a deep learning framework TE(k) that is based on transformer encoder layers, where k is the major hyperparameter that describes the number of input matrices. Besides, there are three hyperparameters that can be tuned: the model dimension d; the number of encoder layers l; the number of attention heads h of the encoder layer. We use this TE(k) model with different hyperparameters in TRENDY and other transformer-enhanced methods. See Table 1 for the general structure of TE(k), along with the shape of data after each layer. Notice that the same transformer encoder layer can handle inputs of different lengths. This means that the gene number n is not a predetermined hyperparameter, and we do not need to train a different model for each n. We will train the TE(k) model with $n = 10$ genes and test on data sets with $n = 10/18/20$ genes.

For the first half of the TRENDY method, the TE($k = 1$) model has $k = 1$ input matrix K_t, $d = 64$ model dimension, $l = 7$ encoder layers, and $h = 4$ attention heads. For the second half of the TRENDY method, the TE($k = 3$) model has $k = 3$ input matrices of the same size, A_1, K_0, K_t, $d = 64$ model dimension, $l = 7$ encoder layers, and $h = 8$ attention heads.

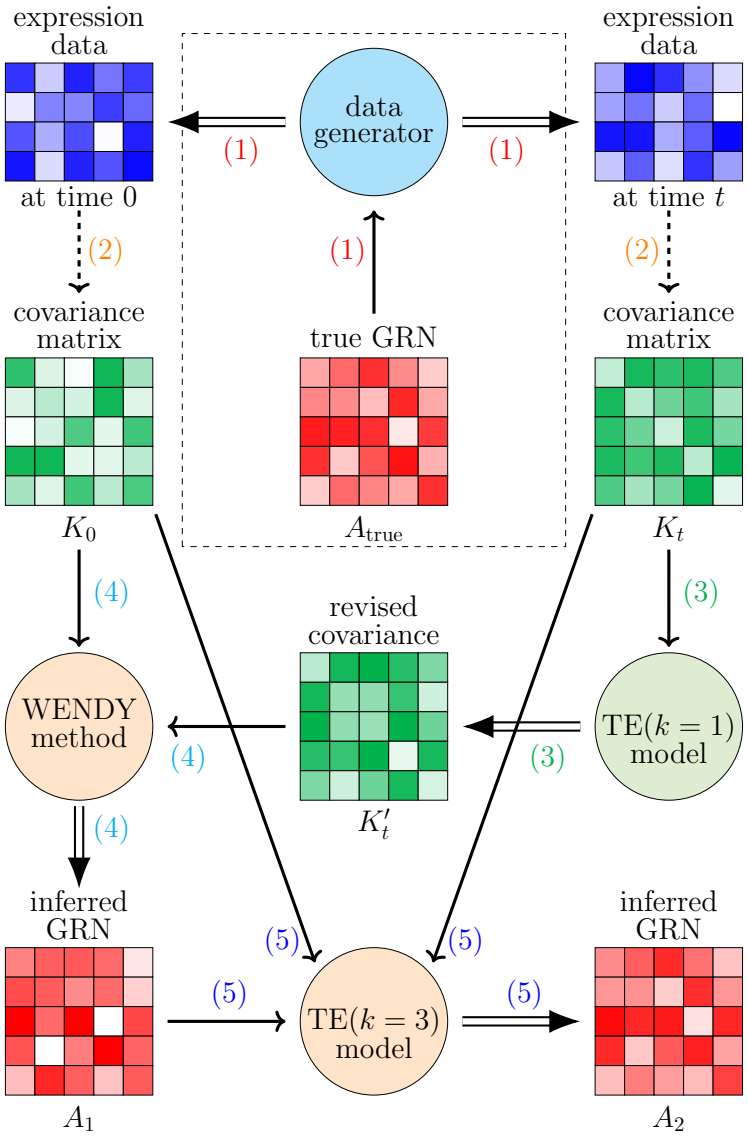

Figure 1: Workflow of training and testing the TRENDY method. Rectangles with grids are numerical matrices, where the color scale represents the numerical value. Circles are mechanisms that take in matrices and produce matrices. Single and double arrows represent inputs and outputs of each mechanism, and dashed arrows represent direct calculations. To apply TRENDY after training, the step in the dashed rectangle is omitted.

input (k matrices)	k groups of $n \times n$
linear embedding layer (1 to d)	k groups of $n \times n \times d$
segment embedding layer (omitted when $k = 1$)	k groups of $n \times n \times d$
2-D positional encoding layer	k groups of $n \times n \times d$
flattening and concatenation	$(n^2) \times (dk)$
l layers of transformer encoder with head number h	$(n^2) \times (dk)$
linear embedding layer ((dk) to 1)	$n^2 \times 1$
reshaping and output	$n \times n$
training target	$n \times n$

Table 1: Structure of the TE(k) model. The left column lists all layers. The right column illustrates the shape of data after each layer, where n is the number of genes, d is the model dimension, and k is the number of input matrices. Notice that k, d, l, h are tunable hyperparameters, but n is not a predetermined hyperparameter.

2.3 Other transformer-enhanced methods

GENIE3 [17] works on single-cell expression data at one time point. It uses random forest to select genes that have predictability for the target gene, thus being information-based.

SINCERITIES [18] works on single-cell expression data at multiple time points. It runs regression on the difference between distributions of expression levels, also being information-based.

NonlinearODEs [14] works on bulk expression data at multiple time points. This method fits the data with a nonlinear differential equation system, which is thus mechanism-based.

For each of these three methods, we train a TE(k) model to enhance the inferred results, similar to the TE($k = 3$) model in the second half of TRENDY. Here each TE(k) model has $d = 64, l = 7, h = 8$, and the training target is A_{true}.

For inferred GRN A_{G} by GENIE3, we train a TE($k = 2$) model with A_{G} and the covariance matrix K as input. Then we can use this trained model to calculate a more accurate A'_{G} from A_{G} and K. This method is named GENIE3-rev.

For inferred GRN A_{S} by SINCERITIES, we train a TE($k = 1$) model with A_{S} as input, and name it SINCERITIES-rev.

For inferred GRN A_{N} by NonlinearODEs, we train a TE($k = 1$) model with A_{N} as input, and name it NonlinearODEs-rev.

2.4 Performance of different methods

The same as the WENDY paper [13], we test the performance of different GRN inference methods on two simulated data sets and two experimental data sets. To compare the inferred GRN A_{pred} and the ground truth GRN A_{true}, we adopt

the common practice that calculates two area under curve (AUC) scores: AUROC and AUPRC [13]. Both AUC scores are between 0 and 1, where 1 means perfect matching, and 0 means perfect mismatching.

2.4.1 SINC data sets

We use the same method for generating training data to generate 1000 testing samples, and name this data set as SINC [18, 13]. Each sample with a random A_{true} has 100 cells measured at time points $0.0, 0.1, \ldots, 1.0$.

For WENDY and TRENDY, we set the first time point to 0.0 and vary the second time point from 0.1 to 1.0. For GENIE3 and GENIE3-rev, we set the time point to any of $0.1, \ldots, 1.0$. For SINCERITIES, SINCERITIES-rev, NonlinearODEs, and NonlinearODEs-rev, we use all time points. See Fig. 2 and Table 2 for the results.

2.4.2 DREAM4 data set

DREAM4 data set is also generated by simulating an SDE system, and consists of multiple data sets under different settings [39]. DREAM4 is commonly used as a benchmark for GRN inference [13, 18]. Here we use the 10-gene time series data set. There are five A_{true}, each corresponding to the levels of 10 genes for five cells at 21 time points.

For WENDY and TRENDY, we consider any pair of time points and take average. For GENIE3 and GENIE3-rev, we consider any one time point and take average. For SINCERITIES, SINCERITIES-rev, NonlinearODEs, and NonlinearODEs-rev, we use any consecutive 11 time points and take average. See Fig. 3 and Table 2 for the results.

2.4.3 THP-1 data set

THP-1 data set measures monocytic THP-1 human myeloid leukemia cells at eight time points [40]. Each time point has 120 cells measured. The same as previous papers [18, 13], we consider the expression levels of 20 genes.

For WENDY and TRENDY, we consider any pair of time points and take average. For GENIE3 and GENIE3-rev, we consider any one time point and take average. For SINCERITIES, SINCERITIES-rev, NonlinearODEs, and NonlinearODEs-rev, we use all eight time points. See Fig. 4 and Table 2 for the results.

2.4.4 hESC data set

hESC data set measures human embryonic stem cell-derived progenitor cells at six time points [41]. Each time point has 66–172 cells measured. The same as in a previous paper [13], we consider the expression levels of 18 genes.

For WENDY and TRENDY, we consider any pair of time points and take average. For GENIE3 and GENIE3-rev, we consider any one time point and take average. For SINCERITIES, SINCERITIES-rev, NonlinearODEs, and

NonlinearODEs-rev, we use all six time points. See Fig. 5 and Table 2 for the results. Notice that the ground truth GRN A_{true} has very few nonzero values. This is why the AUPRC value is much lower than the AUROC value.

2.4.5 Overall evaluation

We summarize the performance of all eight methods on four data sets. See Table 2 for details. The four transformer-enhanced methods (TRENDY, GENIE3-rev, SINCERITIES-rev, NonlinearODEs-rev) are all better than the four known methods. This means that our idea of enhancing GRN inference methods with TE(k) models works well universally. Although the training data only has $n = 10$ genes, and the THP-1 and hESC data sets have $n = 20$ and $n = 18$ genes, the overall performance of transformer-enhanced methods is better than their non-transformer counterparts.

The core method of this paper, TRENDY, ranks the first among all eight methods. Besides, the TRENDY method is better than the WENDY method on three data set, and has almost the same performance as WENDY on one data set (THP-1). The other three transformer-enhanced methods are worse than their non-transformer counterparts on at least one data set. This means that TRENDY's performance is robust on different data sets. In sum, TRENDY has satisfactory performance among all methods tested.

3 Discussion

In this paper, we present the TRENDY method, which uses the transformer model TE(k) to enhance the mechanism-based GRN inference method WENDY. TRENDY is tested against three other transformer-enhanced methods and their original forms, and ranks the first. This work explores the potential of deep learning methods in GRN inference when there are sufficient training data. Besides, TRENDY is developed on the biological dynamics of gene regulation, and it is more interpretable than other deep learning-based methods that treat GRN inference as a pure prediction task.

The essential difficulty of GRN inference is the lack of data with experimentally verified GRNs. Many deep learning models need significantly more data with known GRNs, and we have to simulate new data from a generator. Since we do not fully understand the relation between the dynamics of gene expression and the ground truth GRN, it is difficult to evaluate the quality of generated data along with a random GRN. We also cannot guarantee that GRN inference methods that perform well on known experimental data sets can still be advantageous if there are many more experimental data with different GRNs.

We train the models with data of $n = 10$ genes, and they work well on data with $n = 10/18/20$ genes. If we want the models to work on data with many more genes, we need to train the models with corresponding data. However, when the gene number n is large, the number of different GRNs increases exponentially, and the number of training samples should also increase violently. The

	WEN	TRE	GEN	GENr	SIN	SINr	NON	NONr
S-RO	0.6654	0.8547	0.4062	0.8703	0.6783	0.7964	0.5076	0.5976
S-PR	0.6109	0.7519	0.5061	0.7672	0.5829	0.6637	0.5313	0.5658
D-RO	0.4899	0.5341	0.5636	0.4589	0.4908	0.4995	0.4806	0.5712
D-PR	0.2080	0.2177	0.2286	0.1799	0.1919	0.2034	0.1705	0.2452
T-RO	0.5261	0.5557	0.4484	0.5506	0.6261	0.5251	0.5338	0.4808
T-PR	0.3972	0.3669	0.3546	0.3781	0.3852	0.3412	0.3486	0.3302
h-RO	0.4997	0.5311	0.5913	0.6008	0.4198	0.4871	0.5971	0.6233
h-PR	0.0392	0.0376	0.0468	0.0435	0.0261	0.0294	0.0534	0.0641
Total	3.4364	3.8497	3.1456	3.8493	3.4011	3.5458	3.2229	3.4782
Rank	5th	1st	8th	2nd	6th	3rd	7th	4th

Table 2: Performance summary of eight GRN inference methods on four data sets. For row names, S, D, T, h stand for SINC, DREAM4, THP-1, hESC data sets; -RO and -PR represent AUROC and AUPRC values. For column names, WEN, TRE, GEN, GENr, SIN, SINr, NON, NONr represent WENDY, TRENDY, GENIE3, GENIE3-rev, SINCERITIES, SINCERITIES-rev, NonlinearODEs, NonlinearODEs-rev. The "Total" row is the sum of all eight AUC values, and the last row is the rank of each method by the total score. Red numbers indicate that they are smaller than their non-transformer counterparts (left neighbors) by at least 0.01, and blue color means larger by at least 0.01. For instance, TRENDY has S-RO= 0.8547, larger than WENDY's S-RO= 0.6654 by at least 0.01, so that 0.8547 is in blue.

time cost for generating such data can be quite long. Even with the same amount of training samples, the training speed of the TE(k) model is proportional to n^4, since the input length of the transformer encoder layer is proportional to n^2, and it needs to calculate the attention scores between all pairs of input tokens. Therefore, scaling TRENDY for a much larger n is extremely time-consuming.

In this paper, the training data are generated by Eq. 1, which only describes the dynamics of mRNA count as a continuous-state process. This might be an oversimplification of the gene regulation in reality, where the gene state and the protein count also should be considered [42]. Besides, the experimental gene expression data often suffer from incomplete measurement, where many mRNAs are not recorded, and there are many missing values. Therefore, when we train on perfectly measured data and test on data with missing values, the performance is not guaranteed. We can manually add measurement errors and missing values to the training data to solve this problem.

In our testing, we find that using TE($k = 1$) model to directly enhance an inferred GRN, like SINCERITIES-rev or NonlinearODEs-rev, may not be sufficiently stable. Instead, using TE($k > 1$) model with both inferred GRN and corresponding covariance matrices, like TRENDY or GENIE3-rev, can achieve better results, since the covariance matrices may contain extra information besides that contained in the inferred GRN.

Some deep learning-based GRN inference methods directly use the raw gene expression data as the model input. We think that the covariance matrix contains most of the information regarding the relation between any two genes, and the covariance matrix is calculated from graphical lasso, thus being more regular than the original expression data. Therefore, using covariance matrices in deep learning GRN inference methods should be a promising approach.

4 Methods

4.1 Training data

To generate training data for our TRENDY method, we use Eq. 1 to numerically simulate gene expression data with the Euler-Maruyama method [43] for $n = 10$ genes. Every time we generate a random ground truth GRN A_{true}, where each element has probability 0.1/0.8/0.1 to be $-1/0/1$. Then we use Eq. 1 with this random A_{true} to run the simulation from time 0.0 to time 1.0 with time step 0.01 to obtain one trajectory, where the initial state at time 0 is random. This is repeated 100 times to obtain 100 trajectories, similar to the situation where we measure the single-cell expression levels of n genes for 100 cells [44]. We only record expression levels at time points $0.0, 0.1, 0.2, \ldots, 1.0$, and then calculate the covariance matrix for $n = 10$ genes at each time point with graphical lasso.

We repeat the simulation to obtain 1.01×10^5 samples, where 10^5 samples are for training, and 10^3 samples are for validation (used in hyperparameter tuning and early stopping). Since the TRENDY method only needs two time points, we fix the first time point to be 0.0, and the second time point can be

$0.1, 0.2, \ldots, 1.0$. Therefore, each simulated sample corresponds to 10 pairs of K_0 and K_t. This means that the training set has 10^6 pairs of K_0 and K_t, where each t in $0.1, 0.2, \ldots, 1.0$ corresponds to 10^5 pairs.

For GENIE3-rev, we consider each time point t in $0.1, 0.2, \ldots, 1.0$ and calculate the inferred GRN A_{G}. Then there are 10^6 pairs of A_{G} and K_t for training the TE($k = 2$) model. For SINCERITIES-rev and NonlinearODEs-rev, we use the data at all time points $0.0, 0.1, 0.2, \ldots, 1.0$ to infer the GRN A_{S} and A_{N}. Therefore, each method has 10^5 samples for training the TE($k = 1$) model.

4.2 Loss function

We use AUROC and AUPRC to measure the difference between A_{true} and the inferred A_{pred}. Since A_{true} can take three values $-1/0/1$, the calculation of AUROC and AUPRC need some revisions [13]. Unfortunately, for fixed A_{true} and arbitrary A_{pred}, AUROC and AUPRC can only take finitely many values, and are not continuous functions of A_{pred}. We cannot train a neural network with a non-continuous loss function, since it is not differentiable, and the back propagation cannot work. Therefore, we cannot directly use AUROC or AUPRC as the loss function when training TE(k) models. There have been some surrogate loss functions for AUROC or AUPRC [45, 46], but we find that with enough training data, the mean square error (MSE) function can lead to satisfactory training results. We use MSE as the loss function for the TE($k = 1$) model in TRENDY. For all other TE(k) models used, the loss function is also MSE, but the diagonal elements are ignored, since they represent the autoregulation of one gene i to itself, which cannot be inferred by the methods discussed in this paper [47].

4.3 Segment embedding

For TE(k) model with $k > 1$ input matrices, after processing them with the linear embedding layer, we need to add segment embeddings to them. We give the segment ID $0/1/2/\ldots$ to the first/second/third/... group of processed inputs, and use an embedding layer to map each segment ID to a d-dimensional embedding vector. Repeat this embedding vector $n \times n$ times to obtain an $(n \times n \times d)$ array, and add it to the $(n \times n \times d)$ arrays after the linear embedding layer.

4.4 Positional encoding

For the TE(k) model, the 2-dimensional positional encoding layer [27] adds an $(n \times n \times d)$ array PE to each of the embedded input. For x and y in $1, 2, \ldots, n$

and j in $1, 2, \ldots, d/4$, PE is defined as

$$\begin{aligned}
&\mathrm{PE}[x, y, 2j-1] = \mathrm{PE}[x, y, 2j-1+d/2] = \sin[(y-1) \times 10^{-32(j-1)/d}],\\
&\mathrm{PE}[x, y, 2j] = \mathrm{PE}[x, y, 2j+d/2] = \cos[(y-1) \times 10^{-32(j-1)/d}],\\
&\mathrm{PE}[x, y, 2j-1+d/4] = \mathrm{PE}[x, y, 2j-1+3d/4] = \sin[(x-1) \times 10^{-32(j-1)/d}],\\
&\mathrm{PE}[x, y, 2j+d/4] = \mathrm{PE}[x, y, 2j+3d/4] = \cos[(x-1) \times 10^{-32(j-1)/d}].
\end{aligned} \tag{5}$$

In our numerical simulations, we find that this 2-dimensional positional encoding is crucial for the TE(k) model to produce better results.

4.5 Training setting and cost

All TE(k) models used in this paper are trained for 100 epochs. For each epoch, we evaluate the model performance on the validation data set. If the performance does not improve for 10 consecutive epochs, we stop training early. For all transformer encoder layers, the dropout rate is 0.1. The optimizer is Adam with learning rate 0.001.

Data generation and model training are conducted on a desktop computer with Intel i7-13700 CPU and NVIDIA RTX 4080 GPU. Data generation takes about one week with CPU parallelization. Training for each TE(k) model takes about two hours with GPU acceleration. After training, the time cost of applying TRENDY on a data set with ~ 10 genes and ~ 100 cells is under one second.

Acknowledgments

The authors would like to thank Dr. Zikun Wang for helpful comments. YW would like to thank Dr. Mingda Zhang, Dr. Lingxue Zhu, and Mr. Vincent Zhang for an intriguing discussion. YP would like to thank Irving Institute for Cancer Dynamics of Columbia University for hosting the Summer Research Program.

Data and code availability

Main function of TRENDY method, including a tutorial and the training files, along with other data and code files used in this paper, can be found in

`https://github.com/YueWangMathbio/TRENDY`

References

[1] Zikun Wang, Samantha Lincoln, Andrew D Nguyen, Wanhe Li, and Michael W Young. Chronic sleep loss disrupts rhythmic gene expression in drosophila. *Frontiers in Physiology*, 13:1048751, 2022.

[2] Zikun Wang. *Identification of Gene Expression Changes in Sleep Mutants Associated With Reduced Longevity in Drosophila*. PhD thesis, The Rockefeller University, 2020.

[3] Yue Wang, Jérémie Kropp, and Nadya Morozova. Biological notion of positional information/value in morphogenesis theory. *International Journal of Developmental Biology*, 64(10-11-12):453–463, 2020.

[4] Yu-Chen Cheng, Yun Zhang, Shubham Tripathi, BV Harshavardhan, Mohit Kumar Jolly, Geoffrey Schiebinger, Herbert Levine, Thomas O McDonald, and Franziska Michor. Reconstruction of single-cell lineage trajectories and identification of diversity in fates during the epithelial-to-mesenchymal transition. *Proceedings of the National Academy of Sciences*, 121(32):e2406842121, 2024.

[5] Rene Yu-Hong Cheng, King L Hung, Tingting Zhang, Claire M Stoffers, Andee R Ott, Emmaline R Suchland, Nathan D Camp, Iram F Khan, Swati Singh, Ying-Jen Yang, et al. Ex vivo engineered human plasma cells exhibit robust protein secretion and long-term engraftment in vivo. *Nature Communications*, 13(1):6110, 2022.

[6] Yutong Sha, Yuchi Qiu, Peijie Zhou, and Qing Nie. Reconstructing growth and dynamic trajectories from single-cell transcriptomics data. *Nature Machine Intelligence*, 6(1):25–39, 2024.

[7] Thomas O McDonald, Yu-Chen Cheng, Christopher Graser, Phillip B Nicol, Daniel Temko, and Franziska Michor. Computational approaches to modelling and optimizing cancer treatment. *Nature Reviews Bioengineering*, 1(10):695–711, 2023.

[8] Yu-Chen Cheng, Shayna Stein, Agostina Nardone, Weihan Liu, Wen Ma, Gabriella Cohen, Cristina Guarducci, Thomas O McDonald, Rinath Jeselsohn, and Franziska Michor. Mathematical modeling identifies optimum palbociclib-fulvestrant dose administration schedules for the treatment of patients with estrogen receptor–positive breast cancer. *Cancer Research Communications*, 3(11):2331–2344, 2023.

[9] Erin Angelini, Yue Wang, Joseph Xu Zhou, Hong Qian, and Sui Huang. A model for the intrinsic limit of cancer therapy: Duality of treatment-induced cell death and treatment-induced stemness. *PLOS Computational Biology*, 18(7):e1010319, 2022.

[10] Wanhe Li, Zikun Wang, Sheyum Syed, Cheng Lyu, Samantha Lincoln, Jenna O'Neil, Andrew D Nguyen, Irena Feng, and Michael W Young. Chronic social isolation signals starvation and reduces sleep in drosophila. *Nature*, 597(7875):239–244, 2021.

[11] Vikram Vijayan, Zikun Wang, Vikram Chandra, Arun Chakravorty, Rufei Li, Stephanie L Sarbanes, Hessameddin Akhlaghpour, and Gaby Maimon.

An internal expectation guides drosophila egg-laying decisions. *Science Advances*, 8(43):eabn3852, 2022.

[12] Sofia Axelrod, Xiaoling Li, Yingwo Sun, Samantha Lincoln, Andrea Terceros, Jenna O'Neil, Zikun Wang, Andrew Nguyen, Aabha Vora, Carmen Spicer, et al. The drosophila blood–brain barrier regulates sleep via moody g protein-coupled receptor signaling. *Proceedings of the National Academy of Sciences*, 120(42):e2309331120, 2023.

[13] Yue Wang, Peng Zheng, Yu-Chen Cheng, Zikun Wang, and Aleksandr Aravkin. Gene regulatory network inference with covariance dynamics. *Mathematical Biosciences*, page 109284, 2024.

[14] Baoshan Ma, Mingkun Fang, and Xiangtian Jiao. Inference of gene regulatory networks based on nonlinear ordinary differential equations. *Bioinformatics*, 36(19):4885–4893, 2020.

[15] Cassandra Burdziak, Chujun Julia Zhao, Doron Haviv, Direna Alonso-Curbelo, Scott W Lowe, and Dana Pe'er. sckinetics: inference of regulatory velocity with single-cell transcriptomics data. *Bioinformatics*, 39(Supplement_1):i394–i403, 2023.

[16] Lingfei Wang, Nikolaos Trasanidis, Ting Wu, Guanlan Dong, Michael Hu, Daniel E Bauer, and Luca Pinello. Dictys: dynamic gene regulatory network dissects developmental continuum with single-cell multiomics. *Nature Methods*, 20(9):1368–1378, 2023.

[17] Vân Anh Huynh-Thu, Alexandre Irrthum, Louis Wehenkel, and Pierre Geurts. Inferring regulatory networks from expression data using tree-based methods. *PloS one*, 5(9):e12776, 2010.

[18] Nan Papili Gao, SM Minhaz Ud-Dean, Olivier Gandrillon, and Rudiyanto Gunawan. SINCERITIES: inferring gene regulatory networks from time-stamped single cell transcriptional expression profiles. *Bioinformatics*, 34(2):258–266, 2018.

[19] Ruiqing Zheng, Min Li, Xiang Chen, Fang-Xiang Wu, Yi Pan, and Jianxin Wang. Bixgboost: a scalable, flexible boosting-based method for reconstructing gene regulatory networks. *Bioinformatics*, 35(11):1893–1900, 2019.

[20] Vân Anh Huynh-Thu and Pierre Geurts. dyngenie3: dynamical genie3 for the inference of gene networks from time series expression data. *Scientific reports*, 8(1):3384, 2018.

[21] Meike Nauta, Doina Bucur, and Christin Seifert. Causal discovery with attention-based convolutional neural networks. *Machine Learning and Knowledge Extraction*, 1(1):19, 2019.

[22] Kyriakos Kentzoglanakis and Matthew Poole. A swarm intelligence framework for reconstructing gene networks: searching for biologically plausible architectures. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 9(2):358–371, 2011.

[23] Hantao Shu, Jingtian Zhou, Qiuyu Lian, Han Li, Dan Zhao, Jianyang Zeng, and Jianzhu Ma. Modeling gene regulatory networks using neural network architectures. *Nature Computational Science*, 1(7):491–501, 2021.

[24] Ke Feng, Hongyang Jiang, Chaoyi Yin, and Huiyan Sun. Gene regulatory network inference based on causal discovery integrating with graph neural network. *Quantitative Biology*, 11(4):434–450, 2023.

[25] Guo Mao, Zhengbin Pang, Ke Zuo, Qinglin Wang, Xiangdong Pei, Xinhai Chen, and Jie Liu. Predicting gene regulatory links from single-cell rna-seq data using graph neural networks. *Briefings in Bioinformatics*, 24(6):bbad414, 2023.

[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[27] Jing Xu, Aidi Zhang, Fang Liu, and Xiujun Zhang. Stgrns: an interpretable transformer-based method for inferring gene regulatory networks from single-cell transcriptomic data. *Bioinformatics*, 39(4):btad165, 2023.

[28] Hantao Shu, Fan Ding, Jingtian Zhou, Yexiang Xue, Dan Zhao, Jianyang Zeng, and Jianzhu Ma. Boosting single-cell gene regulatory network reconstruction via bulk-cell transcriptomic data. *Briefings in Bioinformatics*, 23(5):bbac389, 2022.

[29] Thalia E Chan, Michael PH Stumpf, and Ann C Babtie. Gene regulatory network inference from single-cell data using multivariate information measures. *Cell systems*, 5(3):251–267, 2017.

[30] Hendrik Hache, Christoph Wierling, Hans Lehrach, and Ralf Herwig. Genge: systematic generation of gene regulatory networks. *Bioinformatics*, 25(9):1205–1207, 2009.

[31] Kenji Kamimoto, Blerta Stringa, Christy M Hoffmann, Kunal Jindal, Lilianna Solnica-Krezel, and Samantha A Morris. Dissecting cell identity via network inference and in silico gene perturbation. *Nature*, 614(7949):742–751, 2023.

[32] Payam Dibaeinia and Saurabh Sinha. SERGIO: a single-cell expression simulator guided by gene regulatory networks. *Cell systems*, 11(3):252–271, 2020.

[33] Thomas Schaffter, Daniel Marbach, and Dario Floreano. GeneNetWeaver: in silico benchmark generation and performance profiling of network inference methods. *Bioinformatics*, 27(16):2263–2270, 2011.

[34] Shailesh Tripathi, Jason Lloyd-Price, Andre Ribeiro, Olli Yli-Harja, Matthias Dehmer, and Frank Emmert-Streib. sgnesr: An r package for simulating gene expression data from an underlying real gene network structure considering delay parameters. *BMC bioinformatics*, 18:1–12, 2017.

[35] Mohamed Marouf, Pierre Machart, Vikas Bansal, Christoph Kilian, Daniel S Magruder, Christian F Krebs, and Stefan Bonn. Realistic in silico generation and augmentation of single-cell rna-seq data using generative adversarial networks. *Nature communications*, 11(1):166, 2020.

[36] Andrea Pinna, Nicola Soranzo, and Alberto De La Fuente. From knockouts to networks: establishing direct cause-effect relationships through graph analysis. *PloS one*, 5(10):e12912, 2010.

[37] Yue Wang and Zikun Wang. Inference on the structure of gene regulatory networks. *Journal of Theoretical Biology*, 539:111055, 2022.

[38] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

[39] Daniel Marbach, James C Costello, Robert Küffner, Nicole M Vega, Robert J Prill, Diogo M Camacho, Kyle R Allison, Manolis Kellis, James J Collins, et al. Wisdom of crowds for robust gene network inference. *Nature methods*, 9(8):796–804, 2012.

[40] Tsukasa Kouno, Michiel de Hoon, Jessica C Mar, Yasuhiro Tomaru, Mitsuoki Kawano, Piero Carninci, Harukazu Suzuki, Yoshihide Hayashizaki, and Jay W Shin. Temporal dynamics and transcriptional control using single-cell gene expression analysis. *Genome biology*, 14:1–12, 2013.

[41] Li-Fang Chu, Ning Leng, Jue Zhang, Zhonggang Hou, Daniel Mamott, David T Vereide, Jeea Choi, Christina Kendziorski, Ron Stewart, and James A Thomson. Single-cell rna-seq reveals novel regulators of human embryonic stem cell differentiation to definitive endoderm. *Genome biology*, 17:1–20, 2016.

[42] James Holehouse, Zhixing Cao, and Ramon Grima. Stochastic modeling of autoregulatory genetic feedback loops: A review and comparative study. *Biophysical Journal*, 118(7):1517–1525, 2020.

[43] Peter E Kloeden, Eckhard Platen, Peter E Kloeden, and Eckhard Platen. *Stochastic differential equations*. Springer, 1992.

[44] Hong Qian and Yu-Chen Cheng. Counting single cells and computing their heterogeneity: from phenotypic frequencies to mean value of a quantitative biomarker. *Quantitative Biology*, 8(2):172–176, 2020.

[45] Qi Qi, Youzhi Luo, Zhao Xu, Shuiwang Ji, and Tianbao Yang. Stochastic optimization of areas under precision-recall curves with provable convergence. *Advances in neural information processing systems*, 34:1752–1765, 2021.

[46] Zhuoning Yuan, Dixian Zhu, Zi-Hao Qiu, Gang Li, Xuanhui Wang, and Tianbao Yang. Libauc: A deep learning library for x-risk optimization. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5487–5499, 2023.

[47] Yue Wang and Siqi He. Inference on autoregulation in gene expression with variance-to-mean ratio. *Journal of Mathematical Biology*, 86(5):87, 2023.

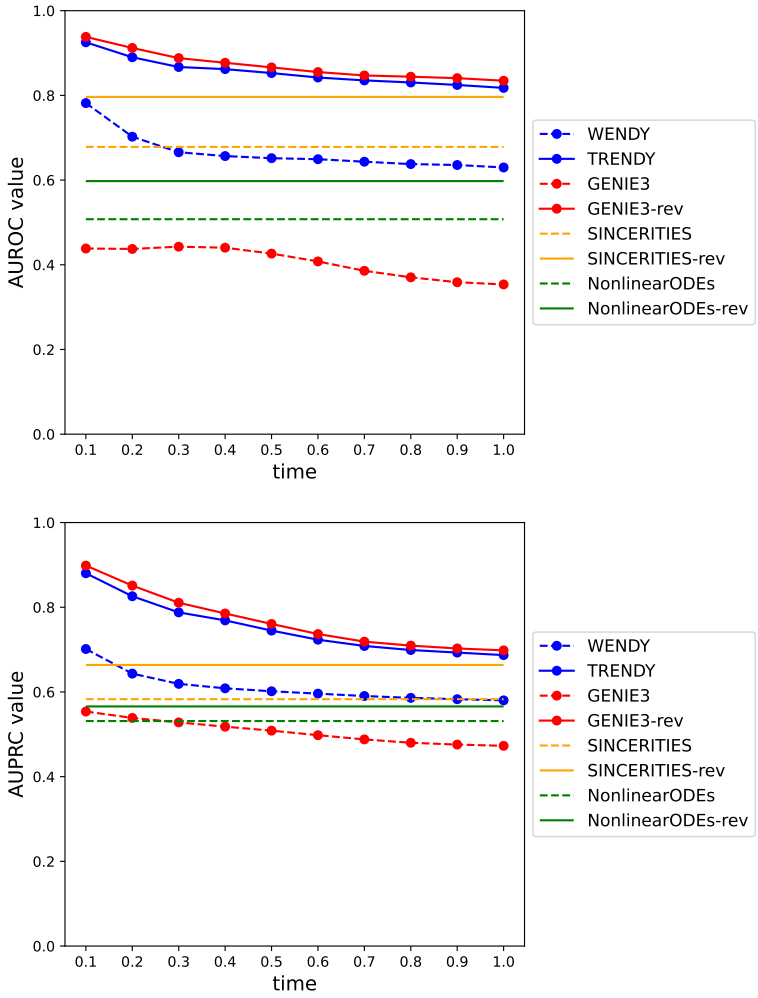

Figure 2: Average AUROC and AUPRC values of eight GRN inference methods on the SINC data set. Notice that SINCERITIES, NonlinearODEs and their revised versions need data at all time points.

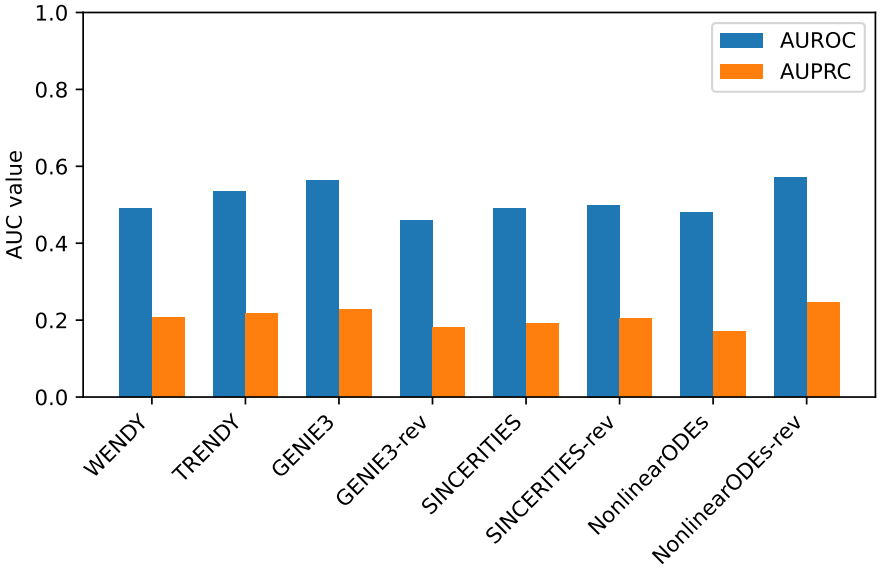

Figure 3: Average AUROC and AUPRC values of eight GRN inference methods on the DREAM4 data set.

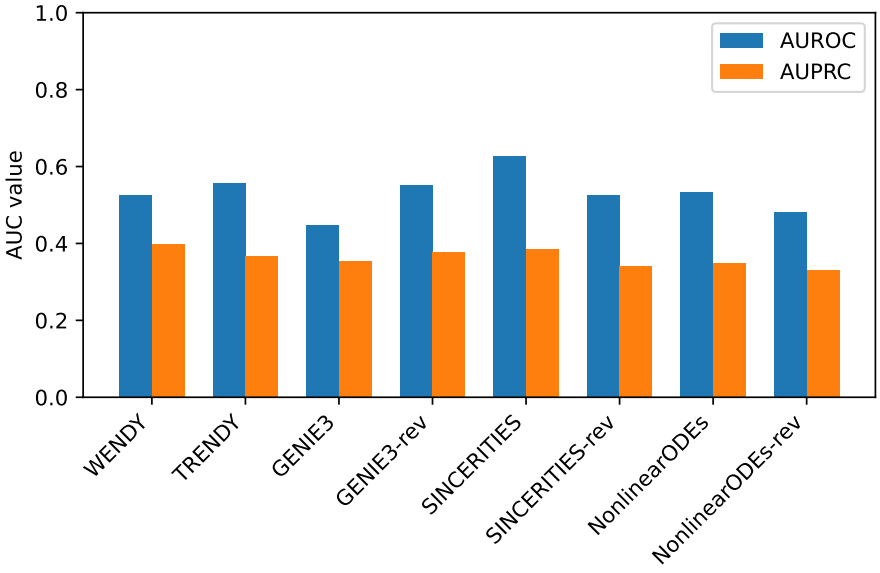

Figure 4: AUROC and AUPRC values of eight GRN inference methods on the THP-1 data set.

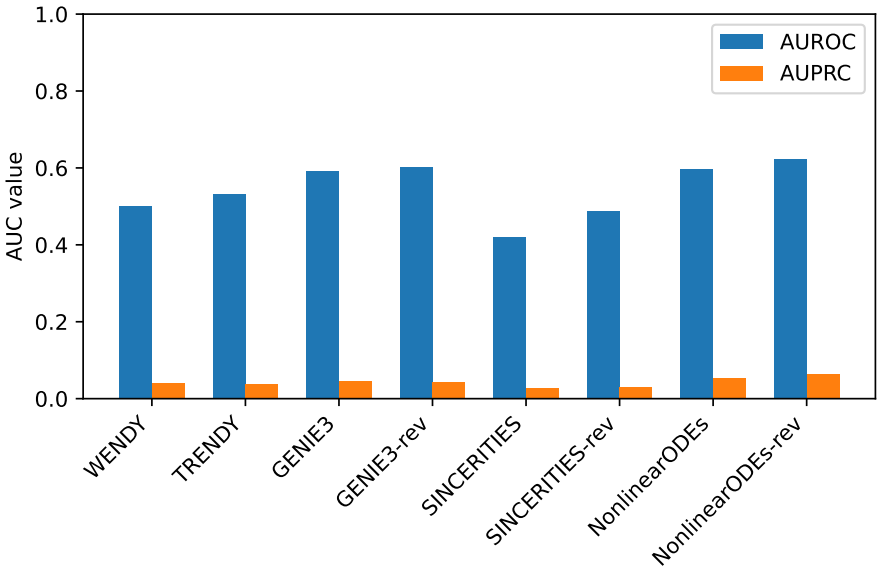

Figure 5: AUROC and AUPRC values of eight GRN inference methods on the hESC data set.