QWENDY: Gene Regulatory Network Inference Enhanced by Large Language Model and Transformer

Yue Wang[1,*] and Xueying Tian[2]

[1]Irving Institute for Cancer Dynamics and Department of Statistics, Columbia University, New York
[2]School of Information, University of California, Berkeley
*Corresponding author, yw4241@columbia.edu, ORCID: 0000-0001-5918-7525

Abstract

Knowing gene regulatory networks (GRNs) is important for understanding various biological mechanisms. In this paper, we present a method, QWENDY, that uses single-cell gene expression data measured at four time points to infer GRNs. Based on a linear gene expression model, it solves the transformation for the covariance matrices. Unlike its predecessor WENDY, QWENDY avoids solving a non-convex optimization problem and produces a unique solution. Then we enhance QWENDY by the transformer neural networks and large language models to obtain two variants: TEQWENDY and LEQWENDY. We test the performance of these methods on two experimental data sets and two synthetic data sets. TEQWENDY has the best overall performance, while QWENDY ranks the first on experimental data sets.

1 Introduction

One gene can activate or inhibit the expression of another gene. Genes and such regulation relations form a gene regulatory network (GRN). For n genes, the corresponding GRN is commonly expressed as an $n \times n$ matrix A, where $A_{ij} > 0/ = 0/ < 0$ means gene i has positive/no/negative regulation effect on gene j, and its absolute value represents the regulation strength. If the regulation relations of concerned genes are known, one can understand the corresponding biological process or even control it with gene perturbation. Therefore, knowledge of GRNs can be useful in developmental biology [1, 2, 3], and even in the study of macroscopic behavior [4, 5, 6]. Since it is difficult to determine the

GRN directly, the common practice is to infer the GRN from gene expression data.

With new measurement techniques, such as scRNA-seq, one can measure the expression levels (mRNA counts) of different genes for a single cell. Since gene expression at the single cell level is random, this measurement can be repeated for different cells, and we can use those samples to obtain the probability distribution of different genes at this time point. Since the measurement kills cells, one cell can only be measured once. Therefore, we cannot obtain the joint probability distribution of gene expression levels at different time points. When we measure at multiple time points, we can only obtain a marginal probability distribution of different genes for each time point.

If the gene expression is at stationary, measurement at multiple time points will produce the same distribution. If the gene expression is away from stationary, such as after adding drugs or during development, we can measure the gene expression at multiple time points and obtain several different probability distributions. For single-cell expression data of n genes at one time point, we can calculate the mean value of each gene (n independent values) and the $n \times n$ covariance matrix ($n(n+1)/2$ independent values, since the covariance matrix is symmetric), while higher-order statistics are not numerically stable due to limited cell number. Therefore, we have $n + n(n+1)/2$ independent known values, much smaller than what is needed to fully determine the $n \times n$ GRN [7]. Therefore, we prefer the non-stationary distributions measured at multiple time points, since they contain enough information to infer the GRN.

For such single-cell gene expression data measured at multiple time points, although it is the most informative data type under current technology, there are only a few GRN methods developed specifically for this data type [8]. We have developed WENDY method to infer the GRN with this data type [9]. It needs measurement at two time points, and solves the transformation between the covariance matrices of two probability distributions, where the transformation is determined by the GRN. However, it needs to solve a non-convex optimization problem, which has infinitely many solutions, and WENDY will output one solution, determined by the numerical solver.

In this paper, we present an improved version of WENDY that needs data measured at four time points, and it is also based on solving the transformation between covariance matrices. This new method is named QWENDY, where Q stands for quadruple. With the help of data from more time points, QWENDY can uniquely determine the GRN. Besides, QWENDY does not need to conduct non-convex optimizations, but just matrix decompositions.

WENDY method is derived from a simplified gene expression model, and the performance of WENDY is not satisfactory when the model does not fit with experiments. Instead of developing a more complicated gene expression model, we developed TRENDY method [10], which trains a transformer model, a deep learning architecture that can be applied to various fields [11], to transform the input data to better fit the gene expression model. Then the transformed data will produce more accurate GRNs by WENDY. Such more accurate GRNs will be further enhanced by another transformer model.

We use the idea of TRENDY to enhance QWENDY. The enhanced version of QWENDY that trains two new transformer models is named TEQWENDY. Besides, we also present another approach that fine-tunes a large language model (LLM) to replace the transformer model, since it has a large pre-trained transformer section. This LLM-enhanced version is named LEQWENDY. The training of TEQWENDY and LEQWENDY uses synthetic data.

We test the performance of QWENDY, TEQWENDY, and LEQWENDY on two experimental data sets and two synthetic data sets. The TRENDY paper [10] has tested 16 methods on the same data sets, meaning that we can compare them with our three new methods. In all 19 GRN inference methods, TEQWENDY has the best overall performance. If we restrict ourselves to two experimental data sets, QWENDY ranks first. Therefore, we recommend applying QWENDY or TEQWENDY.

Section 2 reviews other GRN inference methods, especially those related to LLMs. Section 3 introduces QWENDY method. We develop TEQWENDY and LEQWENDY methods in Section 4. Section 5 tests the performance of different methods on four data sets. We finish with some discussions in Section 6.

2 Literature review

There have been numerous non-deep learning GRN inference methods, and they can be roughly classified as information-based and model-based. Information-based methods [12, 13, 8] do not rely on models for gene expression, but treat GRN inference as a feature selection problem: for a target gene, select genes that can be used to predict the level of the target gene. A common obstacle for information-based methods is to distinguish between direct and indirect regulations.

Model-based methods construct concrete models for gene expression under regulation, and fit the expression data to the models to determine model parameters and reconstruct the GRN. Some methods [14] require measuring the same cell multiple times, which is not quite applicable for now. Some methods [15, 16] use average expression levels measured over many cells (bulk level), which do not utilize the rich information in the single-cell level measurements. Some methods [17] only work on single-cell data at one time point. For single-cell gene expression data measured at multiple time points, where each cell is measured only once, we only know one model-based method, WENDY.

Deep learning-based GRN inference methods [18, 19, 20, 21, 22] generally use neural networks as black boxes, without integrating them with gene expression models. Therefore, lacking of interpretability is a common problem.

Similar to TRENDY, there are some approaches to enhance GRNs inferred by other known methods[23, 24, 25, 26, 27], but they generally require extra knowledge of transcription factors or cannot fully determine the GRN.

Since WENDY paper and TRENDY paper contain more detailed summaries for traditional GRN inference methods and deep learning-based GRN inference methods, we present a review for GRN inference methods related to LLMs.

Some researchers directly used LLMs as an oracle machine to generate answers, without further training the model. Azam et al. [28] asked different GRNs whether one gene regulates another gene. Afonja et al. [29] provided GPT-4 with potential transcription factors and asked it to generate a GRN. Wu et al. [30] provided GPT-4 with related papers and asked it to summarize regulations and form a GRN.

Some researchers worked on pre-trained LLMs and fine-tuned them with new data. Weng et al. [31] trained GPT-3.5 with related papers to obtain GenePT, a new LLM that can provide a high-dimensional embedding (a vector of real numbers) of each gene. Then this embedding was used to train a neural network to output the GRN. Yang et al. [32] trained BERT with scRNAseq data to obtain scBERT, which can also provide gene embedding. Kommu et al. [33] used the embedding from scBERT to infer the GRN.

Some researchers trained new models from scratch. Cui et al. [34] trained a new model, scGPT, with expression data from many cells. The structure of scGPT is similar to other LLMs, although with a smaller size. This model can provide gene embedding, which was used to infer GRN.

For general applications of LLMs in bioinformatics, readers may refer to two reviews [35, 36].

3 QWENDY method

3.1 Setup

At four time points $T = 0$, $T = t$, $T = 2t$, $T = 3t$, we measure the single-cell gene expression levels. For the expression levels of n genes from m cells at each time point, we treat them as m samples of an n-dimensional probability distribution. Then we first calculate the average level of each gene over m cells, and use graphical lasso to calculate the $n \times n$ covariance matrix for different genes. The $1 \times n$ expected levels at four time points are denoted as $\mathbf{x}_0$, $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$. The covariance matrices at four time points are denoted as K_0, K_1, K_2, K_3. Notice that the process does not start from stationary, so that the probability distributions of n genes are different for different time points, and these $\mathbf{x}_i$ and K_i are not equal.

In the WENDY paper, the relationship for K_0, K_1 and the GRN A is derived after some approximations:

$$K_1 = (I + tA^{\mathrm{T}})K_0(I + tA),$$

where I is the $n \times n$ identity matrix. Define $B = I + tA$, we have

$$K_1 = B^{\mathrm{T}} K_0 B. \tag{1}$$

From this equation, WENDY directly solves B (and thus A) by minimizing $||K_1 - B^{\mathrm{T}} K_0 B||_{\mathrm{F}}^2$, where F is the Frobenius norm. This problem is non-convex and has infinitely many solutions. WENDY outputs one solution of them.

For K_2, K_3, similarly, we also have approximated equations

$$K_2 = B^{\mathrm{T}} K_1 B, \tag{2}$$

$$K_3 = B^{\mathrm{T}} K_2 B. \tag{3}$$

In this paper, we study whether we can better solve B with data from more time points, especially K_2, K_3. Since $B^{\mathrm{T}} K B = (-B)^{\mathrm{T}} K (-B)$, we cannot distinguish between B and $-B$ from K_0, K_1, K_2, K_3.

To solve this problem, we use $\mathbf{x}_0$, $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$. In the WENDY paper, it is derived that

$$\mathbf{x}_1 = \mathbf{x}_0 B + t\mathbf{c}, \tag{4}$$

where $\mathbf{c}$ is an unknown vector. Similarly,

$$\mathbf{x}_2 = \mathbf{x}_1 B + t\mathbf{c}, \tag{5}$$

$$\mathbf{x}_3 = \mathbf{x}_2 B + t\mathbf{c}. \tag{6}$$

Since Eqs. 1–6 are approximated, there are two problems: (1) if these equations hold accurately, can we solve B; (2) if these equations do not quite hold, can we find B to minimize the error. We will present the QWENDY method that provides positive answers to both problems.

For the first problem, Theorem 1 proves that given K_0, K_1, K_2, K_3, $\mathbf{x}_0$, $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$ that satisfy Eqs. 1–6 for some B_0, QWENDY can solve B_0 uniquely.

For the second problem, there are different interpretations.

One interpretation is to minimize

$$||K_1 - B^T K_0 B||_F^2 + ||K_2 - B^T K_1 B||_F^2 + ||K_3 - B^T K_2 B||_F^2$$

for any B. Unfortunately, this optimization problem is non-convex, making it difficult to solve.

Notice that for the gene expression data after interpretation, K_0 is measured earlier than K_3, making it farther from stationary, and more informative. Therefore, we want to emphasize more on K_0, K_1 than K_2, K_3. Our goal is to first find B that minimizes $K_1 - B^{\mathrm{T}} K_0 B$; for such B (not unique), we further determine which minimizes $K_2 - B^{\mathrm{T}} K_1 B$; for such B (still not unique), we finally determine which minimizes $K_3 - B^{\mathrm{T}} K_2 B$. This time we can solve B uniquely up to a $\pm$ sign, which can be determined by $\mathbf{x}_0$, $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$.

We will derive QWENDY by solving the second problem, and then prove that it also solves the first problem.

3.2 Algorithm details

In this section, given general covariance matrices K_0, K_1, K_2, K_3 that may not satisfy Eqs. 1–3 for any B_0, we introduce a procedure to calculate B that approximately solves Eqs. 1–3. Under some mild conditions (some matrices are invertible and have distinct eigenvalues), B can be uniquely determined up to

a $\pm$ sign. Then we use $\mathbf{x}_0$, $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$ to distinguish between B and $-B$. The whole procedure is named as QWENDY method.

Step (1): For any B, we want to minimize

$$||K_1 - B^{\mathrm{T}} K_0 B||_{\mathrm{F}}^2. \tag{7}$$

Assume K_0, K_1, K_2, K_3 are invertible. Consider Cholesky decompositions

$$K_1 = L_1 L_1^{\mathrm{T}},\ K_0 = L_0 L_0^{\mathrm{T}},$$

where L_1 and L_0 are lower-triangular and invertible. Define

$$O = L_1^{-1} B^{\mathrm{T}} L_0,$$

then the target Eq. 7 becomes

$$||L_1 L_1^{\mathrm{T}} - L_1 O O^{\mathrm{T}} L_1^{\mathrm{T}}||_{\mathrm{F}}^2.$$

Therefore, Eq. 7 is minimized to 0 if and only if O is orthonormal: $OO^{\mathrm{T}} = I$.

Since

$$B = L_0^{-\mathrm{T}} O^{\mathrm{T}} L_1^{\mathrm{T}},$$

we use K_0 and K_1 to restrict B to a space with the same dimension as the set of all orthonormal matrices.

Step (2): For such B that minimizes Eq. 7, we want to find B that makes $B^{\mathrm{T}} K_1 B$ close to K_2. Here we do not minimize

$$||K_2 - B^{\mathrm{T}} K_1 B||_{\mathrm{F}}^2$$

as it is difficult. Instead, we want to minimize

$$||L_1^{-1}(K_2 - B^{\mathrm{T}} K_1 B) L_1^{-\mathrm{T}}||_{\mathrm{F}}^2 \tag{8}$$

among B that minimizes Eq. 7.

Assume that $L_0^{-1} K_1 L_0^{-\mathrm{T}}$ does not have repeated eigenvalues. Consider the eigenvalue decomposition decomposition

$$L_0^{-1} K_1 L_0^{-\mathrm{T}} = P_1 D_1 P_1^{\mathrm{T}},$$

where P_1 is orthonormal, and D_1 is diagonal with strictly increasing positive diagonal elements (eigenvalues), since $L_0^{-1} K_1 L_0^{-\mathrm{T}}$ is positive definite and symmetric. Similarly, assume that $L_1^{-1} K_2 L_1^{-\mathrm{T}}$ does not have repeated eigenvalues, and we have

$$L_1^{-1} K_2 L_1^{-\mathrm{T}} = P_2 D_2 P_2^{\mathrm{T}}$$

with orthonormal P_2 and diagonal D_2 with strictly increasing positive diagonal elements (eigenvalues). Now the target Eq. 8 equals

$$||P_2 D_2 P_2^{\mathrm{T}} - O P_1 D_1 P_1^{\mathrm{T}} O^{\mathrm{T}}||_{\mathrm{F}}^2 = ||D_2 - P_2^{\mathrm{T}} O P_1 D_1 P_1^{\mathrm{T}} O^{\mathrm{T}} P_2||_{\mathrm{F}}^2,$$

since P_2 is orthonormal and does not affect Frobenius norm. Define

$$W = P_2^{\mathrm{T}} O P_1,$$

which is orthonormal. Then

$$O = P_2 W P_1^{\mathrm{T}},$$

and Eq. 8 equals

$$||D_2 - W D_1 W^{\mathrm{T}}||_{\mathrm{F}}^2. \tag{9}$$

We use the following lemma to handle Eq. 9:

Lemma 1. *For diagonal D_1, D_2 with strictly increasing diagonal elements and any orthonormal W, Eq. 9 is minimized when W is diagonal, and the diagonal elements are ± 1. (There are 2^n possibilities for such W.)*

Proof. Denote the diagonal elements of D_1 as $\mathbf{d}_1 = [d_{1,1}, \ldots, d_{1,n}]$ with $d_{1,1} < d_{1,2} < \cdots, < d_{1,n-1} < d_{1,n}$, and similarly $\mathbf{d}_2 = [d_{2,1}, \ldots, d_{2,n}]$ with $d_{2,1} < d_{2,2} < \cdots, < d_{2,n-1} < d_{2,n}$ for D_2. We need to minimize the following norm with an orthonormal W.

$$\begin{aligned}
||D_2 - W D_1 W^{\mathrm{T}}||_{\mathrm{F}}^2 &= ||D_2||_{\mathrm{F}}^2 + ||W D_1 W^{\mathrm{T}}||_{\mathrm{F}}^2 - 2\sum_{i=1}^{n} [D_2 \otimes (W D_1 W^{\mathrm{T}})]_{ii} \\
&= ||D_2||_{\mathrm{F}}^2 + ||D_1||_{\mathrm{F}}^2 - 2\sum_{i=1}^{n} d_{2,i} \sum_{j=1}^{n} W_{ij}^2 d_{1,j} \\
&= ||D_2||_{\mathrm{F}}^2 + ||D_1||_{\mathrm{F}}^2 - 2\mathbf{d}_1 (W \otimes W) \mathbf{d}_2^{\mathrm{T}},
\end{aligned}$$

where $\otimes$ is the element-wise product. Thus we just need to maximize $\mathbf{d}_1 (W \otimes W) \mathbf{d}_2^{\mathrm{T}}$. Notice that $W \otimes W$ is doubly-stochastic, meaning that it is non-negative, and each row or column has sum 1. By Birkhoff–von Neumann Theorem [37], $W \otimes W$ can be decomposed to

$$W \otimes W = \sum_{i=1}^{k} c_i Q_i,$$

where Q_i is a permutation matrix, $c_i > 0$, and $\sum_{i=1}^{k} c_i = 1$.

Due to the rearrangement inequality [38], for each permutation matrix Q_i,

$$\mathbf{d}_1 Q_i \mathbf{d}_2^{\mathrm{T}} \leq \mathbf{d}_1 I \mathbf{d}_2^{\mathrm{T}} = \mathbf{d}_1 \mathbf{d}_2^{\mathrm{T}},$$

where the equality holds if and only if $Q_i = I$.

Therefore,

$$\mathbf{d}_1 (W \otimes W) \mathbf{d}_2^{\mathrm{T}} = \sum_{i=1}^{k} c_i \mathbf{d}_1 Q_i \mathbf{d}_2^{\mathrm{T}} \leq \sum_{i=1}^{k} c_i \mathbf{d}_1 \mathbf{d}_2^{\mathrm{T}} = \mathbf{d}_1 \mathbf{d}_2^{\mathrm{T}},$$

where the equality holds if and only if

$$W \otimes W = I,$$

meaning that W is diagonal, and diagonal elements are ± 1. □

We now have

$$B = L_0^{-\mathrm{T}} P_1 W P_2^{\mathrm{T}} L_1^{\mathrm{T}},$$

meaning that given K_0, K_1, K_2, we can restrict B to 2^n possibilities.
Step (3): For such B that minimizes Eq. 8 among B that minimizes Eq. 7, we want to find B that makes $B^{\mathrm{T}} K_2 B$ close to K_3. Here we do not minimize

$$||K_3 - B^{\mathrm{T}} K_2 B||_{\mathrm{F}}^2$$

as it is difficult. Instead, we want to minimize

$$||L_1^{-1}(K_3 - B^{\mathrm{T}} K_2 B) L_1^{-\mathrm{T}}||_{\mathrm{F}}^2. \tag{10}$$

Define

$$G = P_2^{\mathrm{T}} L_1^{-1} K_3 L_1^{-\mathrm{T}} P_2,$$

and

$$H = P_1^{\mathrm{T}} L_0^{-1} K_2 L_0^{-\mathrm{T}} P_1.$$

Eq. 10 equals

$$\begin{aligned} &||P_2^{\mathrm{T}} L_1^{-1} K_3 L_1^{-\mathrm{T}} P_2 - W P_1^{\mathrm{T}} L_0^{-1} K_2 L_0^{-\mathrm{T}} P_1 W||_{\mathrm{F}}^2 \\ =&||G - WHW||_{\mathrm{F}}^2 = ||G||_{\mathrm{F}}^2 + ||WHW||_{\mathrm{F}}^2 - 2||G \otimes (WHW)||_{\mathrm{F}}^2 \\ =&||G||_{\mathrm{F}}^2 + ||H||_{\mathrm{F}}^2 - 2||G \otimes (WHW)||_{\mathrm{F}}^2. \end{aligned} \tag{11}$$

We want to find diagonal W with ± 1 that minimizes Eq. 10, which is equivalent to maximizing $||G \otimes (WHW)||_{\mathrm{F}}^2$. Define $C = G \otimes H$, which is still positive definite and symmetric by Schur product theorem [39]. Assume that C does not have repeated eigenvalues. Denote the diagonal elements of W by $\mathbf{w} = [w_1, \ldots, w_n]$. Then we need to maximize

$$||G \otimes (WHW)||_{\mathrm{F}}^2 = \mathbf{w} C \mathbf{w}^{\mathrm{T}}.$$

Now we relax this problem from $\mathbf{w}$ with $w_i = \pm 1$ to general $\mathbf{v}$ with $||\mathbf{v}||_2^2 = n$:

$$\max_{||\mathbf{v}||_2^2 = n} \mathbf{v} C \mathbf{v}^{\mathrm{T}}.$$

The unique solution (up to a $\pm$ sign) is the eigenvector that corresponds to the largest eigenvalue of C.

After obtaining $\mathbf{v}$, we just need to project it to $[w_1, \ldots, w_n]$ with $w_i = \pm 1$ by taking the sign of each term: $w_i = \mathrm{sign}(v_i)$. Then construct W by putting $[w_1, \ldots, w_n]$ on the diagonal.

This relaxation does not guarantee obtaining the optimal solution for Eq. 11, but we find that it produces the correct answer in almost all simulations. Alternatively, we can directly determine the optimal W for Eq. 11 by brute-force search, as there are finitely many (2^n) possibilities of W.

With

$$B = L_0^{-\mathrm{T}} P_1 W P_2^{\mathrm{T}} L_1^{\mathrm{T}},$$

given K_0, K_1, K_2, K_3, we can uniquely determine B up to a $\pm$ sign. Since $B^{\mathrm{T}}KB = (-B)^{\mathrm{T}}K(-B)$, more K_i cannot provide more information.
Step (4): For B and $-B$ from Step (3), we determine which satisfies Eqs. 4–6 better. Notice that Eqs. 4–6 share the same unknown $\mathbf{c}$. Define

$$\mathbf{c}_0 = (\mathbf{x}_1 - \mathbf{x}_0 B)/t,\ \mathbf{c}_1 = (\mathbf{x}_2 - \mathbf{x}_1 B)/t,\ \mathbf{c}_2 = (\mathbf{x}_3 - \mathbf{x}_2 B)/t,\ \bar{\mathbf{c}} = (\mathbf{c}_0 + \mathbf{c}_1 + \mathbf{c}_2)/3.$$

Then the error for fitting Eqs. 4–6 is

$$||\mathbf{c}_0 - \bar{\mathbf{c}}||_2^2 + ||\mathbf{c}_1 - \bar{\mathbf{c}}||_2^2 + ||\mathbf{c}_2 - \bar{\mathbf{c}}||_2^2 = ||\mathbf{c}_0||_2^2 + ||\mathbf{c}_1||_2^2 + ||\mathbf{c}_2||_2^2 - 3||\bar{\mathbf{c}}||_2^2.$$

We just need to compare the errors of B and $-B$ and choose the smaller one as the output.

With expression data from four time points, we uniquely determine B. See Algorithm 1 for the workflow of QWENDY method.

1. **Input** expression levels of n genes over m cells at time points 0, t, $2t$, $3t$
2. **Calculate** covariance matrices K_0, K_1, K_2, K_3, and mean levels $\mathbf{x}_0$, $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$
3. **Calculate** Cholesky decomposition

   $$K_1 = L_1 L_1^{\mathrm{T}},\ K_0 = L_0 L_0^{\mathrm{T}}$$

   Calculate eigenvalue decomposition

   $$L_0^{-1} K_1 L_0^{-\mathrm{T}} = P_1 D_1 P_1^{\mathrm{T}},\ L_1^{-1} K_2 L_1^{-\mathrm{T}} = P_2 D_2 P_2^{\mathrm{T}}$$

   Calculate

   $$C = (P_2^{\mathrm{T}} L_1^{-1} K_3 L_1^{-\mathrm{T}} P_2) \otimes (P_1^{\mathrm{T}} L_0^{-1} K_2 L_0^{-\mathrm{T}} P_1)$$

4. **Calculate** $\mathbf{v}$, the eigenvector that corresponds to the largest eigenvalue of C, and the projection $\mathbf{w}$ with $w_i = \mathrm{sign}(v_i)$

   Construct W with $\mathbf{w}$ on diagonal
5. **Calculate** $B = L_0^{-\mathrm{T}} P_1 W P_2^{\mathrm{T}} L_1^{\mathrm{T}}$

   Compare total squared errors of B and $-B$ for Eqs. 4–6
6. **Output** B or $-B$, the one that corresponds to the smaller error, and the GRN $A = (B - I)/t$

Algorithm 1: Workflow of QWENDY method.

3.3 Correctness of QWENDY method

Theorem 1. *If covariance matrices K_0, K_1, K_2, K_3 and mean levels $\mathbf{x}_0$, $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$ satisfy Eqs. 1–6 for some B_0, then QWENDY will output B_0.*

Proof. If $K_1 = B_0^{\mathrm{T}} K_0 B_0$, then

$$(L_1^{-1} B_0^{\mathrm{T}} L_0)(L_0^{\mathrm{T}} B_0 L_1^{-\mathrm{T}}) = I,$$

and $L_1^{-1} B_0^{\mathrm{T}} L_0 = O_0$ for some orthonormal O_0. Thus

$$B_0 = L_0^{-\mathrm{T}} O_0^{\mathrm{T}} L_1^{\mathrm{T}},$$

which is among the calculated B in Step (1).

If $K_2 = B_0^{\mathrm{T}} K_1 B_0$, then

$$D_2 = P_2^{\mathrm{T}} O_0 P_1 D_1 P_1^{\mathrm{T}} O_0^{\mathrm{T}} P_2.$$

Since $P_2^{\mathrm{T}} O_0 P_1$ is orthonormal, D_1 and D_2 are similar and have the same eigenvalues. Therefore, D_1 and D_2 as diagonal matrices with increasing diagonal values are equal. Assume that the diagonal elements $d_1, \ldots, d_n$ of D_1 are distinct. Define $W_0 = P_2^{\mathrm{T}} O_0 P_1$, then

$$D_1 W_0 = W_0 D_1,$$

and $W_0[i,j] d_i = W_0[i,j] d_j$. For $i \neq j$, we have $d_i \neq d_j$, meaning that $W_0[i,j] = 0$. Now W_0 is diagonal and orthonormal, implying that its diagonal elements are 1 or -1, and

$$B_0 = L_0^{-\mathrm{T}} P_1 W_0 P_2^{\mathrm{T}} L_1^{\mathrm{T}}$$

is among the calculated B in Step (2).

If $K_3 = B_0^{\mathrm{T}} K_2 B_0$, then

$$P_2^{\mathrm{T}} L_1^{-1} K_3 L_1^{-\mathrm{T}} P_2 = W_0 P_1^{\mathrm{T}} L_0^{-1} K_2 L_0^{-\mathrm{T}} P_1 W_0.$$

Define $\mathbf{w}_0 = [w_1, \ldots, w_n]$ to be the diagonal elements of W_0. Then from Step (3), W_0 minimizes Eq. 10, and $\mathbf{w}_0$ is the solution to

$$\max_{||\mathbf{v}||_2^2 = n} \mathbf{v} C \mathbf{v}^{\mathrm{T}},$$

namely the eigenvector of the largest eigenvalue. Since $w_i = \pm 1$, the projection $w_i = \mathrm{sign}(v_i)$ has no effect, and

$$B_0 = L_0^{-\mathrm{T}} P_1 W_0 P_2^{\mathrm{T}} L_1^{\mathrm{T}}$$

is the unique B (up to a $\pm$ sign) calculated in Step (3).

From Eqs. 4–6,

$$\mathbf{x}_1 - \mathbf{x}_0 B_0 = \mathbf{x}_2 - \mathbf{x}_1 B_0.$$

Assume $\mathbf{x}_1 \neq \mathbf{x}_2$, then

$$\mathbf{x}_1 - \mathbf{x}_0(-B_0) \neq \mathbf{x}_2 - \mathbf{x}_1(-B_0).$$

Therefore, $-B_0$ does not satisfy Eqs. 4–6, and Step (4) chooses the correct B_0 from the two possibilities in Step (3). □

4 Enhanced QWENDY method

QWENDY method is derived from Eqs. 1–6, especially Eqs. 1–3, which are a linear approximation of the actual nonlinear gene expression dynamics. Therefore, real K_0, K_1, K_2, K_3 might not fit with Eqs. 1–3, and it might not be feasible to apply QWENDY directly to real K_0, K_1, K_2, K_3. This problem already exists for WENDY method, where the input matrices K_0, K_1 might not satisfy

$$K_1 = B^{\mathrm{T}} K_0 B.$$

For this problem, TRENDY method [10] proposes that we can construct $K_1^* = B^{\mathrm{T}} K_0 B$, and train a model with input K_1 and target K_1^*, so that the output K_1' is close to K_1^*. Then we can apply WENDY to K_0, K_1' to obtain a more accurate GRN A_1. After that, we can train another model with input K_0, K_1, A_1, and the target is the true GRN A_{true}. Then the final output A_2 is more similar to the true GRN than A_1.

Inspired by TRENDY, we propose a similar solution to enhance QWENDY. See Algorithms 2,3 for details. Since $\mathbf{x}_0$, $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$ are only used to distinguish between B and $-B$, it is not necessary to train another model for them.

1. **Repeat** generating random GRN A_{true} and corresponding gene expression data at time 0, t, $2t$, $3t$ from a gene expression simulator
2. **Calculate** covariance matrices K_0, K_1, K_2, K_3 and mean levels $\mathbf{x}_0$, $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$, and then calculate

   $$K_0^* = K_0,\ K_1^* = B^{\mathrm{T}} K_0^* B,\ K_2^* = B^{\mathrm{T}} K_1^* B,\ K_3^* = B^{\mathrm{T}} K_2^* B,$$

   where $B = I + tA_{\mathrm{true}}$
3. **Train** matrix-learning model 1 with inputs K_0, K_1, K_2, K_3 and target K_0^*, K_1^*, K_2^*, K_3^*

   Call trained matrix-learning model 1 to calculate K_0', K_1', K_2', K_3' from K_0, K_1, K_2, K_3
4. **Call** QWENDY to calculate A_1 from K_0', K_1', K_2', K_3', $\mathbf{x}_0$, $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$
5. **Train** matrix-learning model 2 with input A_1, K_0, K_1, K_2, K_3 and target A_{true}

Algorithm 2: Training workflow of LEQWENDY/TEQWENDY method. For those two matrix-learning models, LEQWENDY adopts LE structure, and TEQWENDY adopts TE structure.

For those two matrix-learning models in Algorithms 2,3, we can adopt the approach of TRENDY to construct a transformer structure with three sections: (1) Pre-process the inputs; (2) Use transformer encoder layers to learn high-

1. **Input**: gene expression data at four equally spaced time points
2. **Calculate** covariance matrices K_0, K_1, K_2, K_3 and mean levels $\mathbf{x}_0$, $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$
3. **Call** trained matrix-learning model 1 to calculate K_0', K_1', K_2', K_3' from K_0, K_1, K_2, K_3
4. **Call** QWENDY to calculate A_1 from K_0', K_1', K_2', K_3', $\mathbf{x}_0$, $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$
5. **Call** trained matrix-learning model 2 to calculate A_2 from A_1, K_0, K_1, K_2, K_3
6. **Output**: inferred GRN A_2

Algorithm 3: Testing workflow of LEQWENDY/TEQWENDY method. For those two matrix-learning models, LEQWENDY adopts LE structure, and TEQWENDY adopts TE structure.

dimensional representations of the inputs; (3) Construct outputs from the high-dimensional representations.

For training the transformer encoder layers in this structure, we present two approaches. One approach is to train new transformer encoder layers from scratch, the same as TRENDY. This structure is named "TE", meaning "transformer-enhanced".

The other approach is to integrate an LLM and fine-tune it. In general, an LLM has three sections: (1) Convert text to vector representations; (2) Use transformer (encoder layers, decoder layers, or both) to learn contextual relationships; (3) Convert model outputs back to natural language [40]. We can choose an LLM with pre-trained transformer encoder layers in the second section, and use them to replace the transformer encoder layers in the TE structure. In our practice, we use the encoder layers of the RoBERTa-large model [41], which have about 300 million parameters.

The pre-trained parameters will be kept frozen while we adopt a parameter-efficient fine-tuning method: LoRA [42]. Lower-rank matrices will be trained and added to the frozen encoding layers to obtain a new encoder in a cost-efficient way. The new structure with RoBERTa is named "LE", representing "LLM-enhanced".

We name Algorithms 2,3 with TE structure as TEQWENDY, and Algorithms 2,3 with LE structure as LEQWENDY. See Appendix for details of TE and LE structures. TEQWENDY has 4.7 million trainable parameters. LEQWENDY has 4.6 million trainable parameters, with 300 million non-trainable (frozen) parameters.

TEQWENDY and LEQWENDY are trained on synthetic data generated by

[43, 8, 9, 10]

$$\mathrm{d}X_j(t) = V\left\{\beta\prod_{i=1}^{n}\left[1+(A_{\mathrm{true}})_{i,j}\frac{X_i(t)}{X_i(t)+1}\right]-\theta X_j(t)\right\}\mathrm{d}t+\sigma X_j(t)\mathrm{d}W_j(t), \tag{12}$$

where $X_i(t)$ is the level of gene i at time t, $W_j(t)$ is a standard Brownian motion, and $V = 30$, $\beta = 1$, $\theta = 0.2$, $\sigma = 0.1$. There are 10^5 training samples, each with the expression levels of 10 genes for 100 cells, measured at four time points.

QWENDY requires that K_0, K_1, K_2, K_3 are symmetric and positive definite. Although the input K_i and the target K_i^* are naturally symmetric and positive definite, the learned output K_i' in Step (3) of Algorithms 2,3 might not be positive definite, or even symmetric. Therefore, in the implementation of QWENDY, we add two extra steps to adjust the input covariance matrices:
1. If K_i is asymmetric, replace K_i by $(K_i + K_i^{\mathrm{T}})/2$.
2. If K_i is not positive definite, in the eigenvalue decomposition $K_i = O\Lambda O^{\mathrm{T}}$, where O is orthonormal, and Λ is diagonal, replace negative values of Λ by small positive values to obtain Λ', and replace K_i by $O\Lambda' O^{\mathrm{T}}$.

5 Performance of different methods

The same as in the WENDY paper and the TRENDY paper [9, 10], we test different methods on two synthetic data sets and two experimental data sets. SINC data set [10] and DREAM4 data set [44] are generated by simulating stochastic differential equation systems. THP-1 data set is from monocytic THP-1 human myeloid leukemia cells [45]. hESC data set is from human embryonic stem cell-derived progenitor cells[46]. THP-1 and hESC data sets each has only one group of data (one ground truth GRN and the corresponding expression levels); DREAM4 data set has five groups of data; SINC data set has 1000 groups of data.

For SINC data set, we use data from all four time points. For DREAM4 data set, we use data from any four consecutive time points and take average. For THP-1 data set and hESC data set, we choose any four time points with equal difference. This is due to the limitation that QWENDY (and thus TEQWENDY and LEQWENDY) is derived for four evenly spaced time points.

We test the performance of QWENDY, TEQWENDY, and LEQWENDY methods on these four data sets. Since TRENDY ranks the first among 16 methods on these four data sets [10], we also present the performance of TRENDY. For each method, we compare the inferred GRN and the ground truth GRN by calculating the AUROC and AUPRC scores [9]. These two scores, both between 0 and 1, evaluate the level of matching, where 1 means perfect match, and 0 means perfect mismatch.

See Table 1 for the scores. We can see that TEQWENDY is slightly better than TRENDY, meaning that TEQWENDY has the best performance in all 19 methods (16 methods tested in TRENDY paper, QWENDY, TEQWENDY, and LEQWENDY). Besides, QWENDY is worse than TRENDY, and LEQWENDY

is even worse than QWENDY, showing that LLMs trained with natural language inputs might not directly help with the numerical task in GRN inference.

Since TEQWENDY and TRENDY are trained on synthetic data, a natural concern is whether the training makes these methods overfitting to certain types of synthetic data, but less suitable for experimental data. Therefore, we also compare the performance of different methods only for two experimental data sets. For all 16 methods tested in TRENDY paper, bWENDY ranks the first on THP-1 and hESC data sets, with total score 1.5783 [10]. In comparison, QWENDY has total score 1.6272, making it the best in all 19 methods on two experimental data sets.

In sum, TEQWENDY method has the best overall performance, while QWENDY ranks the first on experimental data sets. LEQWENDY is worse than QWENDY in most cases, showing the failure of training with an LLM.

		QWENDY	LEQ-WENDY	TEQ-WENDY	TRENDY
SINC	AUROC	0.5314	0.4972	0.9587	0.8941
	AUPRC	0.5720	0.5185	0.9142	0.8314
DREAM4	AUROC	0.4987	0.5164	0.5372	0.5341
	AUPRC	0.1844	0.1823	0.2203	0.2177
THP-1	AUROC	0.5524	0.5543	0.5415	0.5557
	AUPRC	0.4294	0.3632	0.3801	0.3669
hESC	AUROC	0.6019	0.5905	0.4815	0.5311
	AUPRC	0.0435	0.0367	0.0317	0.0376
Total		3.4137	3.2591	4.0652	3.9686

Table 1: AUROC and AUPRC scores of QWENDY, LEQWENDY, TEQWENDY, TRENDY methods on four data sets

6 Discussion

In this paper, we present QWENDY, a GRN inference method that requires single-cell gene expression data measured at four time points. Then we train the transformer-based variant TEQWENDY from scratch and fine-tune the LLM-based variant LEQWENDY based on pre-trained weights. TEQWENDY has the best overall performance, while QWENDY ranks the first on experimental data sets. Notice that each experimental data set only has one group of data, meaning that the performance of each method has a large uncertainty level.

The performance of LEQWENDY is not satisfactory, meaning that the pre-trained LLMs, even after task-specific fine-tuning, do not fit with this task. We need to explore other approaches of applying LLMs in GRN inference, and utilize the fact that LLMs are trained with natural language material.

QWENDY method and its variants require measurements at four **evenly spaced** time points t_0, t_1, t_2, t_3, meaning that $t_1 - t_0 = t_2 - t_1 = t_3 - t_2$. Otherwise, the dynamics of covariance matrices is much more complicated, and we do not have an explicit solution. One possible strategy is to use the approximation

$$I + (t_2 - t_1)A \approx \frac{t_2 - t_1}{t_1 - t_0} \left[I + (t_1 - t_0)A \right],$$

so that different B in Eqs. 1-3 only differ by a constant factor. This means that we can relocate this factor to K_i, and B is the same for all equations.

Since QWENDY and its variants need four time points, the duration of the whole experiment might be too long, so that the dynamics of gene expression under regulation might have changed during this time. QWENDY is based on a time-homogeneous model, which might fail in this situation.

Training TEQWENDY and LEQWENDY on data generated by Eq. 12 does not necessarily increase their performance on experimental data sets. This means that Eq. 12 might not faithfully reflect the gene expression dynamics. To better integrate deep learning techniques, we need better gene expression data generators.

Although QWENDY and its variants can fully determine the GRN, the diagonal elements that represent the regulation of one gene to itself (autoregulation) can be mixed with natural mRNA synthesis and degradation [47]. Therefore, it is not recommended to infer the autoregulation with QWENDY or its variants.

Given new GRN inference methods, we can study how cells maintain homeostasis [48, 49] or be driven away from homeostasis and cause diseases [50, 51].

Acknowledgments

Y.W. would like to thank Dr. Mingda Zhang for a helpful discussion.

Data and code availability

Main function of QWENDY, TEQWENDY, and LEQWENDY methods, 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/QWENDY`

Appendix: details of TEQWENDY and LEQWENDY

For all models in TEQWENDY and LEQWENDY, the loss function is mean squared error; the optimizer is Adam with learning rate 0.001; the number of

training epochs is 100. After each epoch, we evaluate the model performance on a validation set with 1000 samples. The training stops early and rolls back to the best status if there is no improvement for consecutive 10 epochs.

See Algorithm A1 for the structure of the first half of TEQWENDY. The inputs are four covariance matrices K_0, K_1, K_2, K_3. The targets are four revised covariance matrices K_0^*, K_1^*, K_2^*, K_3^*. Notice that the first input matrix (K_0) is not processed through these layers, since the target K_0^* equals K_0. Besides, K_1, K_2, K_3 are processed separately.

1. **Input**: four covariance matrices (4 groups of $n \times n$)
2. Linear embedding layer with dimension 1 to $d = 64$ (4 groups of $n \times n \times d$)
3. ReLU activation function (4 groups of $n \times n \times d$)
4. Linear embedding layer with dimension d to d (4 groups of $n \times n \times d$)
5. 2-D positional encoding layer (4 groups of $n \times n \times d$)
6. Flattening and concatenation (4 groups of $(n^2) \times d$)
7. 7 layers of transformer encoder with 4 heads, dimension d, feedforward dimension $4d$, dropout rate 0.1 (4 groups of $(n^2) \times d$)
8. Linear embedding layer with dimension d to d (4 groups of $(n^2) \times d$)
9. LeakyReLU activation function with $\alpha = 0.1$ (4 groups of $n \times n \times d$)
10. Linear embedding layer with dimension d to 1 (4 groups of $(n^2) \times 1$)
11. **Output**: reshaping into four matrices (4 groups of $n \times n$)

Algorithm A1: Structure of TEQWENDY method, first half. The shape of data after each layer is in the brackets.

The 2-D positional encoding layer incorporates spatial information of matrix to the input. It generates an $n \times n \times d$ array PE and adds it to the embedded input: For x and y in $1, 2, \ldots, n$ and j in $1, \ldots, d/4$,

$$\begin{aligned}
&\mathrm{PE}[x, y, 2j-1] = \cos[(x-1) \times 10^{-16(j-1)/d}],\\
&\mathrm{PE}[x, y, 2j] = \sin[(x-1) \times 10^{-16(j-1)/d}],\\
&\mathrm{PE}[x, y, 2j-1+d/2] = \cos[(y-1) \times 10^{-16(j-1)/d}],\\
&\mathrm{PE}[x, y, 2j+d/2] = \sin[(y-1) \times 10^{-16(j-1)/d}].
\end{aligned}$$

See Algorithm A2 for the structure of the second half of TEQWENDY. After obtaining the outputs K_0', K_1', K_2', K_3' from K_0, K_1, K_2, K_3 by the first half of TEQWENDY, call the QWENDY method to calculate the inferred GRN A_1

from K_0', K_1', K_2', K_3'. The inputs of the second half of TEQWENDY are four covariance matrices K_0, K_1, K_2, K_3, and the inferred GRN A_1. The target is the ground truth GRN A_{true}.

1. **Input**: four covariance matrices and one inferred GRN (5 groups of $n \times n$)
2. Linear embedding layer with dimension 1 to $d = 64$ (5 groups of $n \times n \times d$)
3. Segment embedding layer (5 groups of $n \times n \times d$)
4. 2-D positional encoding layer (5 groups of $n \times n \times d$)
5. Flattening and concatenation $((n^2) \times (5d))$
6. 3 layers of transformer encoder with 4 heads, dimension $5d$, feedforward dimension $20d$, dropout rate 0.1 $((n^2) \times (5d))$
7. Linear embedding layer with dimension $5d$ to 1 $((n^2) \times 1)$
8. **Output**: reshaping into one matrix $(n \times n)$

Algorithm A2: Structure of TEQWENDY method, second half. The shape of data after each layer is in the brackets.

The segment embedding layer generates different trainable d-dimensional vectors for all five inputs. Then each vector is copied into dimension $n \times n \times d$, and added to the embedded inputs. This layer incorporates the source of inputs. The final input of the transformer encoder layers is a matrix, with shape $k \times D$, where k is the total number of input values, and D is the representation dimension. For each location in $1, 2, \ldots, k$, the segment embedding marks which input matrix it is from, and the position encoding marks which position in the matrix it is from. These two layers solve the problem that the inputs are multiple matrices, but the inputs of transformer are representations of a 1-D sequence.

See Algorithm A3 for the structure of the first half of LEQWENDY. The inputs are four covariance matrices K_0, K_1, K_2, K_3. The targets are four revised covariance matrices K_0^*, K_1^*, K_2^*, K_3^*. Notice that the first input matrix (K_0) is not processed through these layers, since the target K_0^* equals K_0. Besides, K_1, K_2, K_3 are processed separately.

For each large pre-trained weight matrix W with size $p \times q$, LoRA freezes W and replace it by $W + \Delta W$. Here $\Delta W = (\text{LoRA-}\alpha/r)AB$, where the trainable A has size $p \times r$, and the trainable B has size $r \times q$. The total number of trainable parameters decreases from pq to $(p + q)r$, since $r \ll p, q$. The scaling factor LoRA-α controls the update rate.

See Algorithm A4 for the structure of the second half of LEQWENDY. After obtaining the outputs K_0', K_1', K_2', K_3' from K_0, K_1, K_2, K_3 by the first half

1. **Input**: four covariance matrices (4 groups of $n \times n$)
2. Linear embedding layer with dimension 1 to $d = 256$ (4 groups of $n \times n \times d$)
3. ReLU activation function (4 groups of $n \times n \times d$)
4. Linear embedding layer with dimension d to d (4 groups of $n \times n \times d$)
5. Segment embedding layer (4 groups of $n \times n \times d$)
6. 2-D positional encoding layer (4 groups of $n \times n \times d$)
7. Flattening and concatenation $((n^2) \times (4d))$
8. Transformer encoder part of the RoBERTa-large model, frozen: 24 layers of transformer encoder with 16 heads, dimension $4d$, feedforward dimension $16d$, dropout rate 0.1;

   Trainable LoRA layers with rank $r = 8$ and LoRA-$\alpha = 16$, added to each transformer encoder layer $((n^2) \times (4d))$
9. Four different linear embedding layers with dimension $4d$ to $2d$ (4 groups of $(n^2) \times (2d)$)
10. LeakyReLU activation function with $\alpha = 0.1$ (4 groups of $(n^2) \times (2d)$)
11. Four different linear embedding layers with dimension $2d$ to 1 (4 groups of $(n^2) \times 1$)
12. **Output**: reshaping into four matrices (4 groups of $n \times n$)

Algorithm A3: Structure of LEQWENDY method, first half. The shape of data after each layer is in the brackets.

of LEQWENDY, call the QWENDY method to calculate the inferred GRN A_1 from K_0', K_1', K_2', K_3'. The inputs of the second half of LEQWENDY are four covariance matrices K_0, K_1, K_2, K_3, and the inferred GRN A_1. The target is the ground truth GRN A_{true}. Since the transformer encoder part of the RoBERTa-large model has a fixed dimension 1024, we need to apply $d_1 = 192$ for each covariance matrix input, and $d_2 = 256$ for the GRN input, so that the total dimension is $d = 4d_1 + d_2 = 1024$.

1. **Input**: four covariance matrices and one inferred GRN (5 groups of $n \times n$)
2. Linear embedding layer with dimension 1 to $d_1 = 192$ or $d_2 = 256$ (4 groups of $n \times n \times d_1$ and 1 group of $n \times n \times d_2$)
3. Segment embedding layer (4 groups of $n \times n \times d_1$ and 1 group of $n \times n \times d_2$)
4. 2-D positional encoding layer (4 groups of $n \times n \times d_1$ and 1 group of $n \times n \times d_2$)
5. Flattening and concatenation $((n^2) \times (d = 4d_1 + d_2 = 1024))$
6. Transformer encoder part of the RoBERTa-large model, frozen: 24 layers of transformer encoder with 16 heads, dimension $4d$, feedforward dimension $16d$, dropout rate 0.1;

   Trainable LoRA layers with rank $r = 16$ and LoRA-$\alpha = 32$, added to each transformer encoder layer $((n^2) \times d)$
7. Linear embedding layer with dimension d to d $((n^2) \times d)$
8. LeakyReLU activation function with $\alpha = 0.1$ $((n^2) \times d)$
9. Linear embedding layer with dimension d to 1 $((n^2) \times 1)$
10. **Output**: reshaping into one matrix $(n \times n)$

Algorithm A4: Structure of LEQWENDY method, second half. The shape of data after each layer is in the brackets.

References

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] 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.

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

[8] 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.

[9] Yue Wang, Peng Zheng, Yu-Chen Cheng, Zikun Wang, and Aleksandr Aravkin. Wendy: Covariance dynamics based gene regulatory network inference. *Mathematical Biosciences*, 377:109284, 2024.

[10] Xueying Tian, Yash Patel, and Yue Wang. Trendy: Gene regulatory network inference enhanced by transformer. *arXiv preprint arXiv:2410.21295*, 2024.

[11] A Waswani, N Shazeer, N Parmar, J Uszkoreit, L Jones, A Gomez, L Kaiser, and I Polosukhin. Attention is all you need. In *NIPS*, 2017.

[12] 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.

[13] 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.

[14] Vân Anh Huynh-Thu and Guido Sanguinetti. Combining tree-based and dynamical systems for the inference of gene regulatory networks. *Bioinformatics*, 31(10):1614–1622, 2015.

[15] Bruno-Edouard Perrin, Liva Ralaivola, Aurelien Mazurie, Samuele Bottani, Jacques Mallet, and Florence d'Alché Buc. Gene networks inference using dynamic bayesian networks. *Bioinformatics-Oxford*, 19(2):138–148, 2003.

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

[17] Hao-Chih Lee, Matteo Danieletto, Riccardo Miotto, Sarah T Cherng, and Joel T Dudley. Scaling structural learning with no-bears to infer causal transcriptome networks. In *Pacific Symposium on Biocomputing 2020*, pages 391–402. World Scientific, 2019.

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

[19] 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.

[20] 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.

[21] 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.

[22] 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.

[23] Soheil Feizi, Daniel Marbach, Muriel Médard, and Manolis Kellis. Network deconvolution as a general method to distinguish direct dependencies in networks. *Nature biotechnology*, 31(8):726–733, 2013.

[24] Mengfei Cao, Hao Zhang, Jisoo Park, Noah M Daniels, Mark E Crovella, Lenore J Cowen, and Benjamin Hescott. Going the distance for protein function prediction: a new distance metric for protein interaction networks. *PloS one*, 8(10):e76339, 2013.

[25] Aurélie Pirayre, Camille Couprie, Frédérique Bidard, Laurent Duval, and Jean-Christophe Pesquet. Brane cut: biologically-related a priori network enhancement with graph cuts for gene regulatory network inference. *BMC bioinformatics*, 16:1–12, 2015.

[26] Aurélie Pirayre, Camille Couprie, Laurent Duval, and Jean-Christophe Pesquet. Brane clust: Cluster-assisted gene regulatory network inference refinement. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 15(3):850–860, 2017.

[27] Bo Wang, Armin Pourshafeie, Marinka Zitnik, Junjie Zhu, Carlos D Bustamante, Serafim Batzoglou, and Jure Leskovec. Network enhancement as a general method to denoise weighted biological networks. *Nature communications*, 9(1):3108, 2018.

[28] Muhammad Azam, Yibo Chen, Micheal Olaolu Arowolo, Haowang Liu, Mihail Popescu, and Dong Xu. A comprehensive evaluation of large language models in mining gene interactions and pathway knowledge. *bioRxiv*, 2024.

[29] Tejumade Afonja, Ivaxi Sheth, Ruta Binkyte, Waqar Hanif, Thomas Ulas, Matthias Becker, and Mario Fritz. Llm4grn: Discovering causal gene regulatory networks with llms–evaluation through synthetic data generation. *arXiv preprint arXiv:2410.15828*, 2024.

[30] Xidong Wu, Sumin Jo, Yiming Zeng, Arun Das, Tinghe Zhang, Parth Patel, Yuanjing Wei, Lei Li, Shou-Jiang Gao, Jianqiu Zhang, et al. regulogpt: Harnessing gpt for end-to-end knowledge graph construction of molecular regulatory pathways. In *IEEE-EMBS International Conference on Biomedical and Health Informatics*, 2024.

[31] Guangzheng Weng, Patrick Martin, Hyobin Kim, and Kyoung Jae Won. Integrating prior knowledge using transformer for gene regulatory network inference. *Advanced Science*, 12(3):2409990, 2025.

[32] Fan Yang, Wenchuan Wang, Fang Wang, Yuan Fang, Duyu Tang, Junzhou Huang, Hui Lu, and Jianhua Yao. scbert as a large-scale pretrained deep language model for cell type annotation of single-cell rna-seq data. *Nature Machine Intelligence*, 4(10):852–866, 2022.

[33] Sindhura Kommu, Yizhi Wang, Yue Wang, and Xuan Wang. Gene regulatory network inference from pre-trained single-cell transcriptomics transformer with joint graph learning. *arXiv preprint arXiv:2407.18181*, 2024.

[34] Haotian Cui, Chloe Wang, Hassaan Maan, Kuan Pang, Fengning Luo, Nan Duan, and Bo Wang. scgpt: toward building a foundation model for single-cell multi-omics using generative ai. *Nature Methods*, pages 1–11, 2024.

[35] Jinge Wang, Zien Cheng, Qiuming Yao, Li Liu, Dong Xu, and Gangqing Hu. Bioinformatics and biomedical informatics with chatgpt: Year one review. *Quantitative Biology*, 12(4):345–359, 2024.

[36] Wei Ruan, Yanjun Lyu, Jing Zhang, Jiazhang Cai, Peng Shu, Yang Ge, Yao Lu, Shang Gao, Yue Wang, Peilong Wang, et al. Large language models for bioinformatics. *arXiv preprint arXiv:2501.06271*, 2025.

[37] Wolfgang B Jurkat and Herbert John Ryser. Term ranks and permanents of nonnegative matrices. *Journal of algebra*, 5(3):342–357, 1967.

[38] Godfrey H Hardy, John E Littlewood, and George Pólya. *Inequalities (Cambridge mathematical library)*. Cambridge university press, 1934.

[39] Dipa Choudhury. The schur product theorem in the block case. *Proceedings of the American Mathematical Society*, 108(4):879–886, 1990.

[40] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.

[41] Yinhan Liu. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 364, 2019.

[42] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[43] 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.

[44] 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.

[45] 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.

[46] 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.

[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.

[48] 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.

[49] Zikun Wang. *Identification of Gene Expression Changes in Sleep Mutants Associated With Reduced Longevity in Drosophila*. PhD thesis, The Rockefeller University, 59–74, 2020.

[50] 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.

[51] 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.