Gene Regulatory Network Inference Enhanced by Transformer and LLM

Yue Wang

Irving Institute for Cancer Dynamics and
Department of Statistics, Columbia University

yw4241@columbia.edu

May 22, 2025

Outline

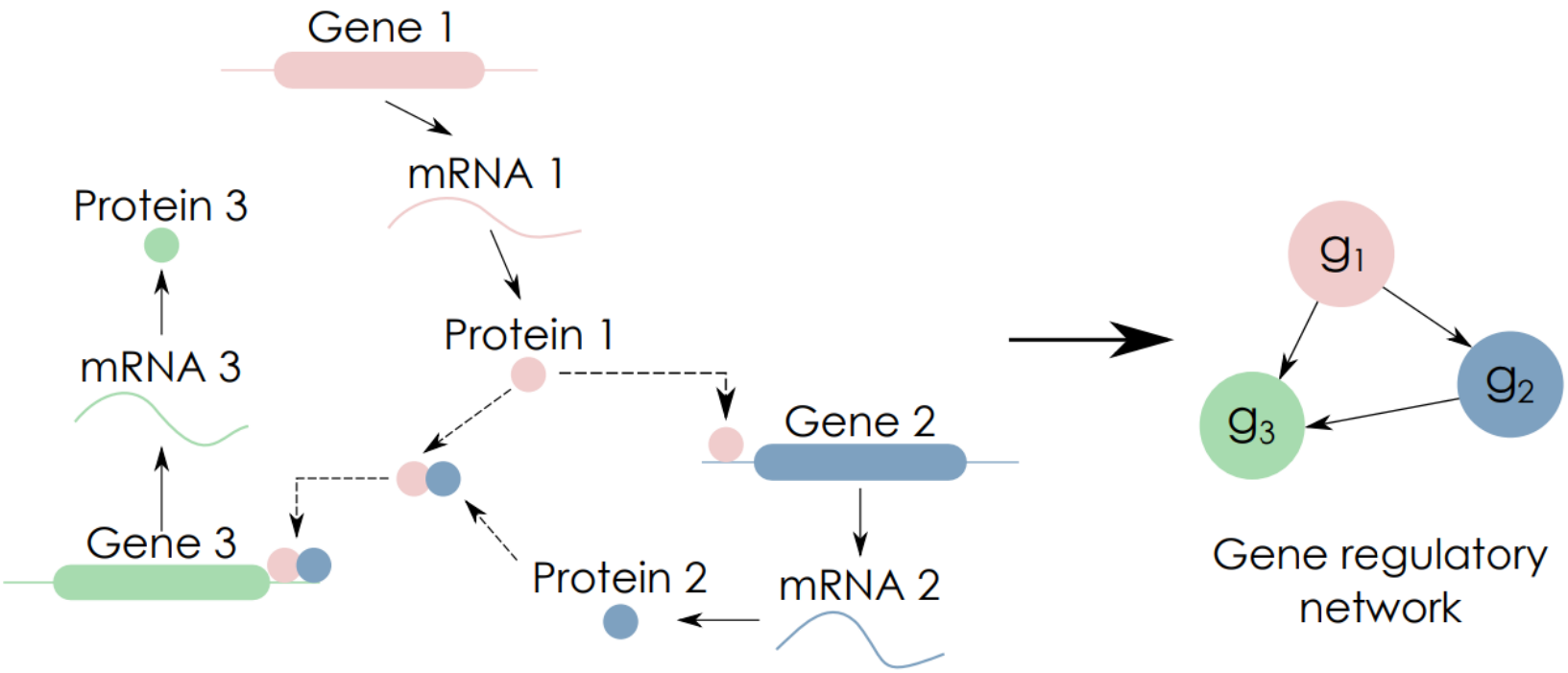
- Introduce gene regulatory network (GRN).
- Review the GRN inference method WENDY, and introduce its transformer-enhanced version, TRENDY.
- Introduce a novel GRN inference method, QWENDY, and its variants enhanced by transformers and large language models (LLMs).

Regulation of gene expression

- Gene expression: genes are transcribed to mRNAs and then translated to proteins.
- Various molecular regulators affect gene expression (change levels of mRNAs and/or proteins).
- Some regulators are small molecules, such as oxygen, sugars and vitamins. Some regulators are proteins. We focus on regulations between genes.

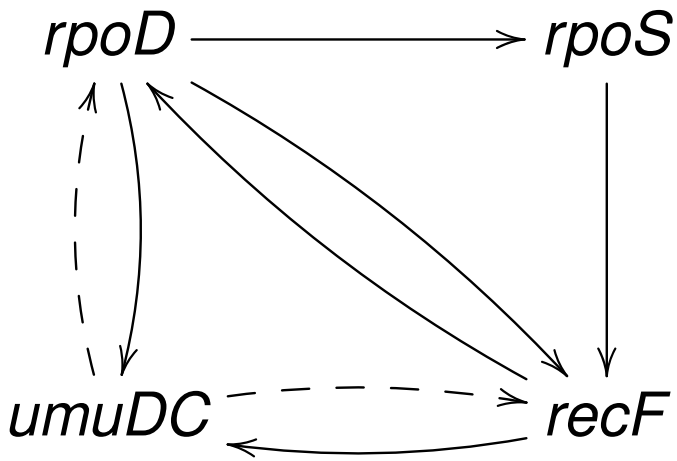
$$\text{gene} \xrightarrow{\text{expression}} \text{protein} \xrightarrow{\text{regulation}} \text{another gene}$$

Regulation of gene expression



Genes and their regulatory relations form a gene regulatory network (GRN). It is a directed graph: vertices are genes, and edges are regulatory relations.

Regulation of gene expression



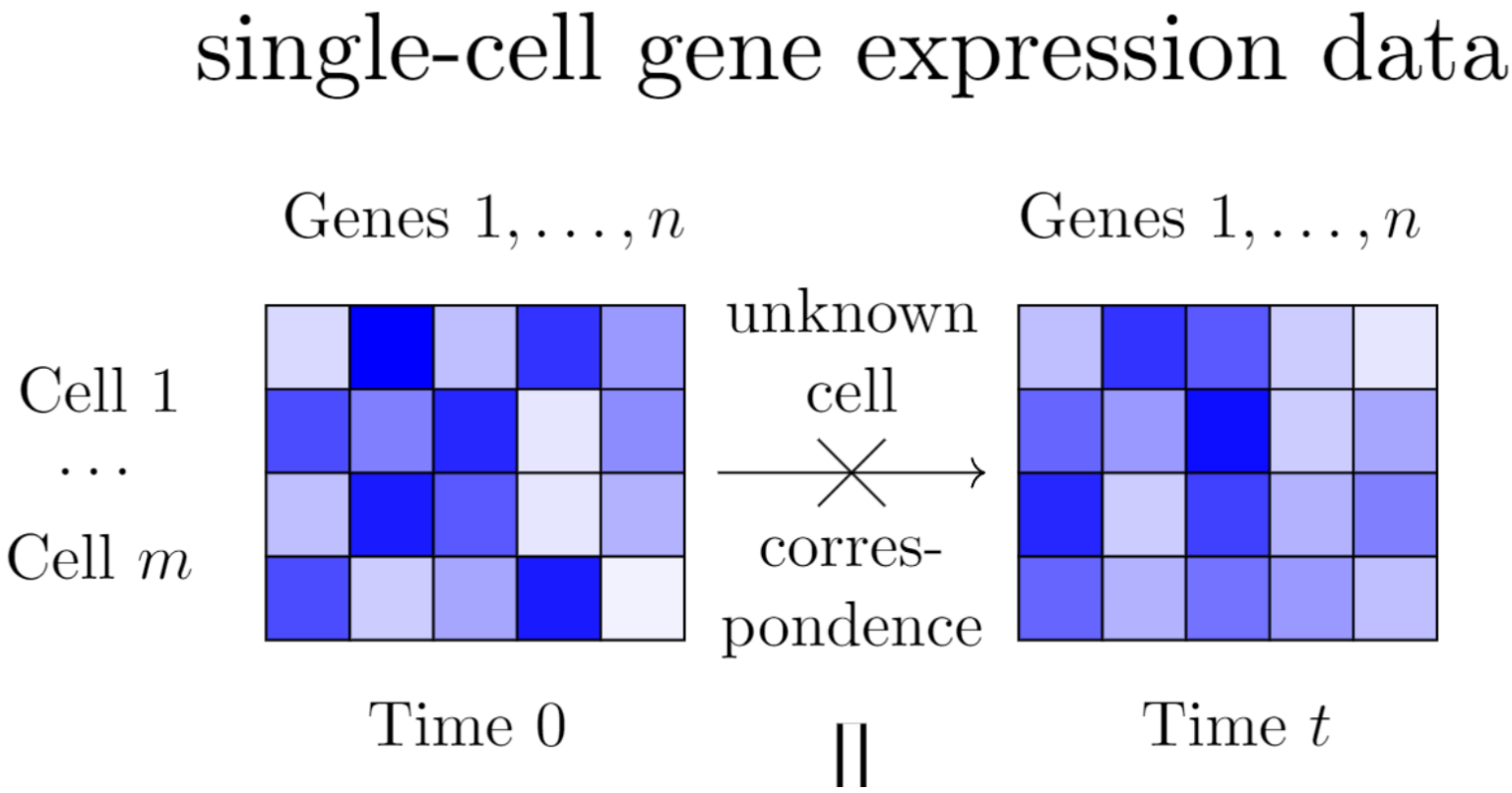
- An example of GRN in *E. coli*. Each vertex is a gene. Two types of regulations: solid arrow means activation, and dashed arrow means inhibition.
- GRN determines cell function. If the GRN structure is known, we know how the cell fate is determined, and how we can change the cell fate.

Regulation of gene expression

- A central question in biology is to determine the GRN structure.
- For two genes V_i, V_j, does the expression of V_i regulate (activate or inhibit) the expression of V_j?
- Genes (DNAs), mRNAs and proteins are generally confined within living cells.
- It is extremely difficult or even impossible to directly determine whether one gene regulates another gene with biochemical methods.
- We have accumulated a large amount of gene expression data. Certain types of gene expression data can be used to infer the GRN structure.

Data type

- We consider a specific data type, which is common in experiment and very informative, but there are very few GRN inference methods developed for it specifically.
- For each cell, we can use single-cell RNA sequencing to measure the expression levels (mRNA counts) of different genes.
- Since single-cell level gene expression is stochastic, we repeat this for different cells. The data are represented as a matrix with m rows (cell number) and n columns (gene number), namely m samples of an n-dimensional random variable.
- We do this at multiple time points. If the cells are not at stationary, different time points have different distributions and more information.
- At different time points, we measure different cells, since measurement kills cells.



At time 0 and t, we obtain expression data $X(0)$ and $X(t)$, where each has m samples for an n-dimensional random variable.

WENDY method

- In a previous paper, we developed the WENDY method for this type of data.
- We can obtain a simplified relation for $X(0)$ and $X(t)$:

$$X(t) \approx X(0)(I + tA) + tb + X(0) \odot \epsilon(t).$$

- Here A is an $n \times n$ matrix, representing the GRN we want: $A_{i,j} > 0$ means gene i activates gene j; $A_{i,j} < 0$ means gene i inhibits gene j; and $A_{i,j} = 0$ means gene i does not regulate gene j directly.
- Besides, b is a $1 \times n$ vector, representing the base synthesis rate, $\epsilon(t)$ is random noise, and $\odot$ is the entrywise (Hadamard) product.

WENDY method

- For n-dimensional $X(0)$ or $X(t)$, we can calculate the $n \times n$ covariance matrix K_0 or K_t.
- When sample number m is small, we can use graphical lasso to avoid K_0 being degenerate.
- We can obtain the dynamics of K_t:

$$K_t \approx (I + tA^{\mathrm{T}})K_0(I + tA).$$

- Now we have an optimization problem:

$$\min_A f(A) := ||K_t - (I + tA^{\mathrm{T}})K_0(I + tA)||_{\mathrm{F}}^2.$$

- This non-convex optimization problem can be solved numerically with BFGS or other methods.
- WENDY method outputs the GRN A from K_0 and K_t.

TRENDY method

- In a recent paper, we developed a method to enhance WENDY.
- In practice, K_t does not accurately equal to $(I + tA^{\mathrm{T}})K_0(I + tA)$.
- Therefore, even if the numerical solver works perfectly, we might not obtain the accurate GRN A.
- Define

$$K_t^* = (I + tA^{\mathrm{T}})K_0(I + tA).$$

- If we feed K_0 and K_t^* to the solver, we could get a better A.
- We know K_0 and K_t from data, but A is unknown, so we cannot calculate K_t^*.

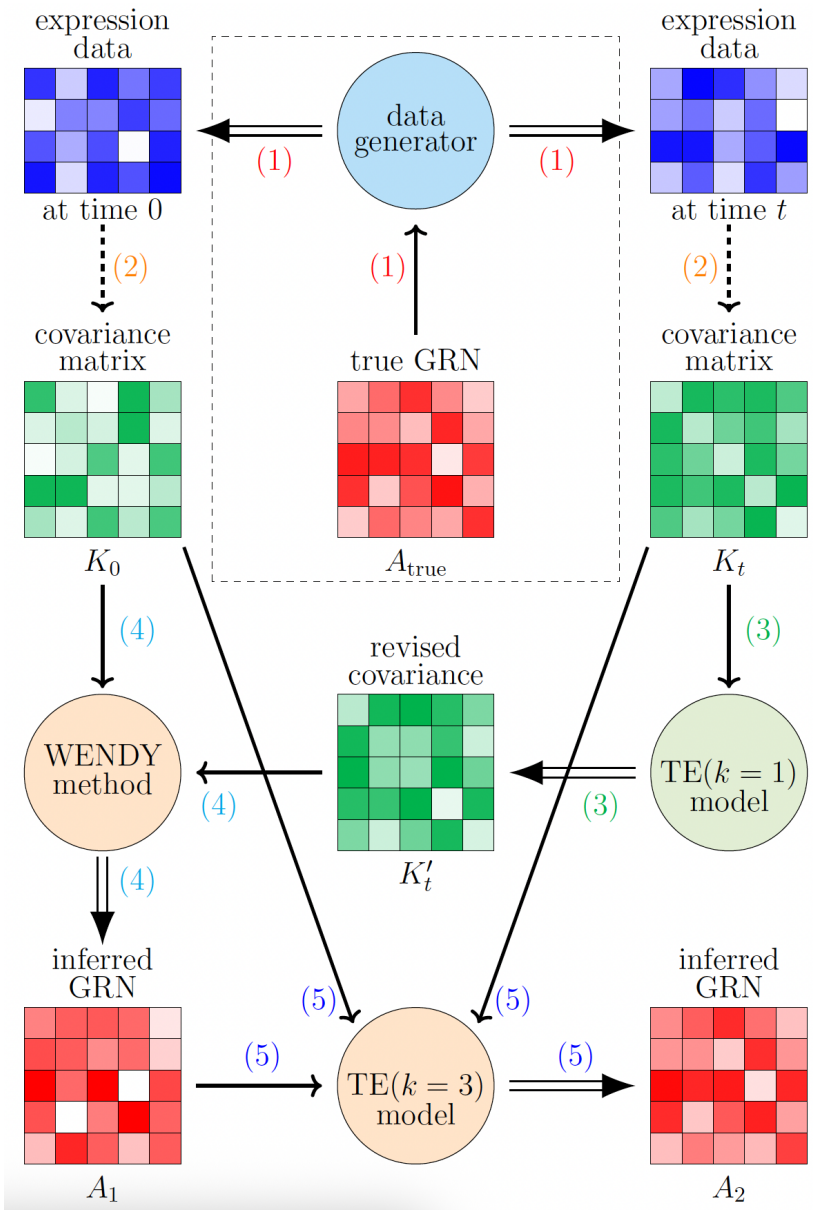
TRENDY method

- We can use some simulator to generate many gene expression data with random ground truth GRN A_{true}:

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

- Then train a supervised-learning model with input K_t and target K_t^* (which is known for training data), so the output K_t' is close to K_t^*.
- In reality, when we only have K_0 and K_t, use this model to output K_t', and feed K_0 and K_t' to the solver to obtain a better GRN, A_1.
- We can train another supervised-learning model with inputs A_1, K_0, K_t and target A_{true}. The output A_2 should be better than A_1.

TRENDY method



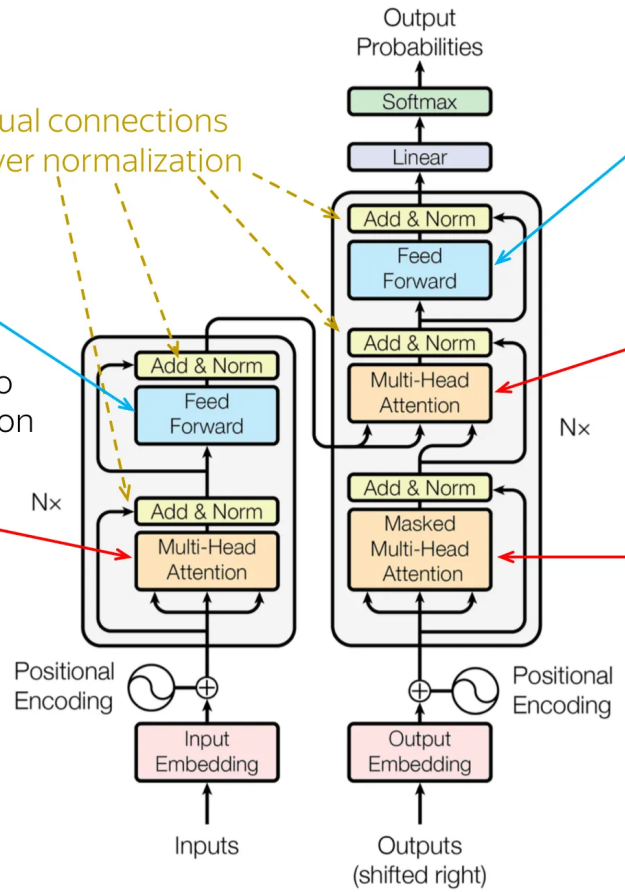
TRENDY method

- What supervised-learning model can be used for this task?
- The input is one or several matrices, and the target is a matrix. It is a highly nonlinear regression problem, and simple linear/nonlinear regression does not work.
- Since we have a data simulator, which leads to a large amount of data, we can train a deep learning model with many parameters.
- In TRENDY, the core of this model is the transformer encoder.

TRENDY method

- The transformer model, introduced by Vaswani et al. in 2017, is a neural network architecture designed for sequence modeling tasks such as translation and text generation.
- The transformer architecture has become the foundation of many modern models, including BERT, GPT, and T5.
- A transformer is composed of an encoder and a decoder:
  - The **encoder** consists of multiple identical layers that process the input sequence and generate context-rich embeddings.
  - The **decoder** also consists of multiple layers and generates the output sequence by attending to both previous decoder outputs and the encoder output.

TRENDY method



Residual connections and layer normalization

Feed-forward network: after taking information from other tokens, take a moment to think and process this information

Encoder self-attention: tokens look at each other

queries, keys, values are computed from encoder states

Feed-forward network: after taking information from other tokens, take a moment to think and process this information

Decoder-encoder attention: target token looks at the source

queries – from decoder states; keys and values from encoder states

Decoder self-attention (masked): tokens look at the previous tokens

queries, keys, values are computed from decoder states

TRENDY method

- For our highly nonlinear regression task, we can use the transformer encoder.
- Transformer encoder can be regarded as an oracle machine that learns the features of the input, which work well for downstream tasks.
- For transformer encoder, the input is m tokens, each as a d_{model}-dimensional vector, thus being a matrix of size $m \times d_{\text{model}}$. The output is also an $m \times d_{\text{model}}$ matrix. Here d_{model} is called the model dimension.

- (Optional): Details of transformer encoder and TRENDY.

TRENDY method

- The core component of transformer encoder, where X is the $m \times d_{\text{model}}$ input:

$$Q = XW^Q, \quad K = XW^K, \quad V = XW^V,$$

$$\text{Attention} = \text{softmax}\left(\frac{QK^{\text{T}}}{\sqrt{d_{\text{model}}}}\right) V.$$

  Here W^Q, W^K, W^V (query, key, value) are $d_{\text{model}} \times d_{\text{model}}$ trainable matrices.
- It is the self-attention mechanism, which allows each token in a sequence to attend to all other tokens, capturing long-range dependencies effectively.

TRENDY method

1. **Input**: k matrices (k groups of $n \times n$)
2. Linear embedding layer with dimension d (k groups of $n \times n \times d$)
3. Segment embedding layer (omitted when $k = 1$) (k groups of $n \times n \times d$)
4. 2-D positional encoding layer (k groups of $n \times n \times d$)
5. Flattening and concatenation $((n^2) \times (dk))$
6. l layers of transformer encoder $((n^2) \times (dk))$
7. Linear embedding layer $(n^2 \times 1)$
8. **Output**: reshaping into a matrix $(n \times n)$

Structure of the $\mathrm{TE}(k)$ model used in TRENDY

TRENDY method

- Since the transformer encoder does not have a built-in notion of sequence order, **positional encodings** are added to input embeddings before they are fed into the model, so the position information of each token is known to the model.

$$\mathrm{PE}_{\mathrm{pos},2i} = \sin\left(\frac{\mathrm{pos}}{10000^{\frac{2i}{d_{\mathrm{model}}}}}\right)$$

$$\mathrm{PE}_{\mathrm{pos},2i+1} = \cos\left(\frac{\mathrm{pos}}{10000^{\frac{2i}{d_{\mathrm{model}}}}}\right)$$

TRENDY method

Since our input is 2-D matrices, we need a 2-D positional encoding.

$$\mathrm{PE}[x, y, 2j-1] = \mathrm{PE}[x, y, 2j-1+d/2] = \sin[(y-1) \times 10^{-32(j-1)/d}],$$

$$\mathrm{PE}[x, y, 2j] = \mathrm{PE}[x, y, 2j+d/2] = \cos[(y-1) \times 10^{-32(j-1)/d}],$$

$$\mathrm{PE}[x, y, 2j-1+d/4] = \mathrm{PE}[x, y, 2j-1+3d/4] = \sin[(x-1) \times 10^{-32(j-1)/d}],$$

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

TRENDY method

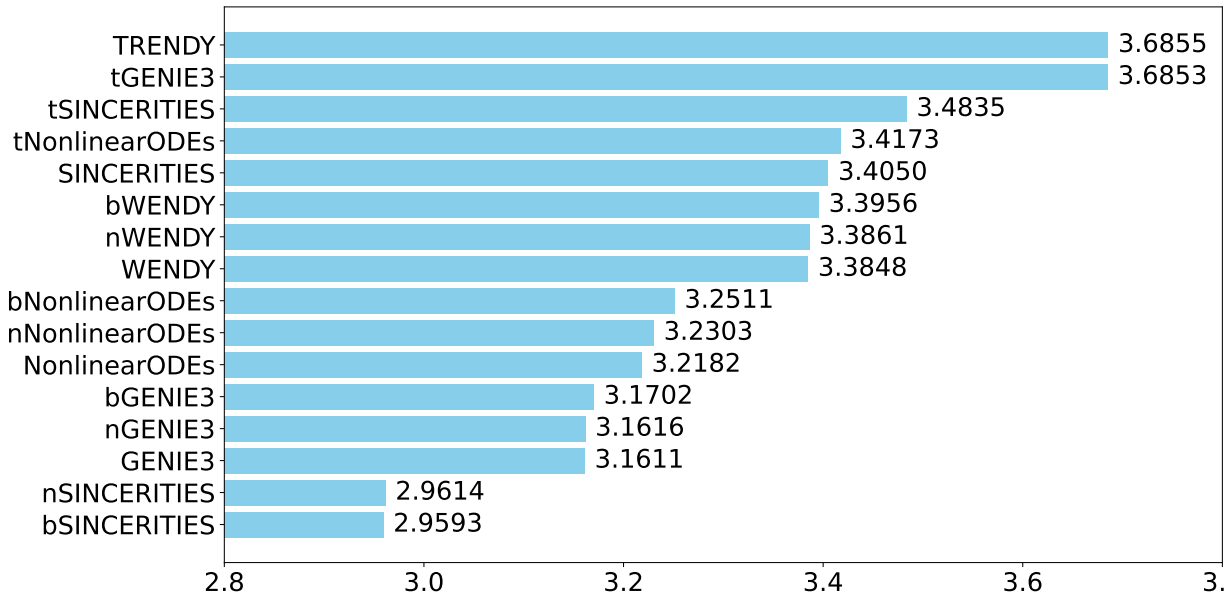
- Since the model does not distinguish between different input matrices, we need to add different **segment embedding** vectors to different input matrices.
- The segment embedding vector is trainable with length d_{model}, and it is the same for all tokens in the same input matrix.

TRENDY method

- WENDY method uses two covariance matrices to infer the GRN.
- WENDY relies on a linear approximation, which leads to an error.
- TRENDY uses the transformer encoder to learn this error and decrease it.
- This idea can be used to enhance other GRN inference methods: use the inferred GRN as input and true GRN as target, and train a similar transformer encoder.
- There are other methods to enhance GRN inference, but they cannot determine the direction of regulation.

TRENDY method

- We test 16 methods: four base methods, their transformer-enhanced versions and two other enhanced version.
- There are two synthetic data sets, DREAM4 and SINC, and two experimental data sets, hESC and THP-1.
- We use two quantities to measure the performance: AUROC and AUPRC. They are between 0 (perfect mismatch) and 1 (perfect match).
- Here is the sum of AUROC and AUPRC on four data sets.



QWENDY method

- We introduce TRENDY, which uses the transformer encoder to enhance WENDY.
- With covariance matrices from two time points, WENDY solves a non-convex optimization problem with infinitely many solutions.
- With covariance matrices from more time points, can we infer the GRN easier?
- With data from four time points, we can uniquely determine the GRN just by matrix factorization.
- This new method is called QWENDY, where Q means quadruple.

QWENDY method

- We have data at time points 0, t, $2t$, $3t$, each as m samples for n genes. Their $n \times n$ covariance matrices are K_0, K_1, K_2, K_3. For the GRN matrix A, define $B = I + tA$.
- We have approximated relations:

$$K_1 = B^{\mathrm{T}} K_0 B, \quad K_2 = B^{\mathrm{T}} K_1 B, \quad K_3 = B^{\mathrm{T}} K_2 B. \tag{1}$$

- Define the $1 \times n$ expected levels at four time points as $\mathbf{x}_0$, $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$. They have approximated relations:

$$\mathbf{x}_1 = \mathbf{x}_0 B + t\mathbf{c}, \quad \mathbf{x}_2 = \mathbf{x}_1 B + t\mathbf{c}, \quad \mathbf{x}_3 = \mathbf{x}_2 B + t\mathbf{c}, \tag{2}$$

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

QWENDY method

- Given K_0, K_1, K_2, K_3, $\mathbf{x}_0$, $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$, we will solve B from Eq. 1 and Eq. 2.
- **Step** (1): For any B, we want to minimize

$$||K_1 - B^{\mathrm{T}} K_0 B||_{\mathrm{F}}^2. \tag{3}$$

- 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

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

then the target Eq. 3 becomes

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

Therefore, Eq. 3 is minimized to 0 if and only if O is orthonormal: $OO^{\mathrm{T}} = I$.

- 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. 3, 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

$$||L_1^{-1}(K_2 - B^{\mathrm{T}} K_1 B)L_1^{-\mathrm{T}}||_{\mathrm{F}}^2 \tag{4}$$

among B that minimizes Eq. 3.

QWENDY method

- Consider the eigenvalue 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)

- We also 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. 4 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.

QWENDY method

- Define

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

  where W is orthonormal.
- Then Eq. 4 equals

$$||D_2 - W D_1 W^{\mathrm{T}}||_{\mathrm{F}}^2. \tag{5}$$

- We use the following lemma to handle Eq. 5:

Lemma

For diagonal D_1, D_2 with strictly increasing diagonal elements and any orthonormal W, Eq. 5 is minimized when W is diagonal, and the diagonal elements are ± 1. (There are 2^n possibilities for such W.)

- Given K_0, K_1, K_2, we can restrict B to 2^n possibilities.

- **Step** (3): For such B that minimizes Eq. 4 among B that minimizes Eq. 3, 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

$$||L_1^{-1}(K_3 - B^{\mathrm{T}} K_2 B)L_1^{-\mathrm{T}}||_{\mathrm{F}}^2. \tag{6}$$

- 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. 6 equals

$$
\begin{aligned}
&||P_2^{\mathrm{T}} L_1^{-1} K_3 L_1^{-\mathrm{T}} P_2 - W P_1^{\mathrm{T}} L_0^{-1} K_2 L_0^{-\mathrm{T}} P_1 W||_{\mathrm{F}}^2 \\
=&||G - WHW||_{\mathrm{F}}^2 = ||G||_{\mathrm{F}}^2 + ||WHW||_{\mathrm{F}}^2 - 2||G \otimes (WHW)||_{\mathrm{F}}^2 \\
=&||G||_{\mathrm{F}}^2 + ||H||_{\mathrm{F}}^2 - 2||G \otimes (WHW)||_{\mathrm{F}}^2,
\end{aligned} \tag{7}
$$

  where $\otimes$ is element-wise product.

- We want to maximize $||G \otimes (WHW)||_{\mathrm{F}}^2$. Define $C = G \otimes H$, which is still positive definite and symmetric.
- Denote the diagonal elements of W by $\mathbf{w} = [w_1, \ldots, w_n]$. Then we need to maximize

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

QWENDY method

- With general vector $\mathbf{v}$ with $||\mathbf{v}||_2^2 = n$, $\mathbf{v}C\mathbf{v}^{\mathrm{T}}$ is maximized by the eigenvector that corresponds to the largest eigenvalue of C (unique up to a $\pm$ sign).
- 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 we have W, and finally

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

- Given K_0, K_1, K_2, K_3, we can uniquely determine B up to a $\pm$ sign.
- Since $B^{\mathrm{T}} K B = (-B)^{\mathrm{T}} K (-B)$, more K_i cannot provide more information.

QWENDY method

- **Step** (4): For B and $-B$ from Step (3), we can use $\mathbf{x}_0$, $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$ to determine which is better, since they cannot both satisfy Eq. 2.
- We just need to compare the errors of B and $-B$ and choose the smaller one as the output.

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 $A = (B - I)/t$

Workflow of QWENDY method.

QWENDY method

- Since Eq. 1 and Eq. 2 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.
- The derivation of QWENDY solves problem (2). The following theorem shows that QWENDY also solves problem (1).

Theorem

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 Eq. 1 and Eq. 2 for some B_0, then QWENDY will output B_0.

QWENDY method

- Similar to TRENDY, we can try to enhance QWENDY.
- One idea is to train a transformer encoder to calculate K_1', K_2', K_3' that better fit Eq. 1. The corresponding enhanced method with 3M trainable parameters is named TEQWENDY.
- We can also try large language models (LLMs). 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.
- LLMs are trained with massive natural language data.
- We can choose an LLM with pre-trained encoder layers and fine-tune it.

QWENDY method

- We use the RoBERTa Large model with 24 transformer encoder layers and 300M parameters.
- Training all parameters is not feasible. We use LoRA method to fine-tune it.
- If the original pre-trained weight matrix W (frozen) has size $p \times q$, LoRA trains two low rank matrices A and B, with size $p \times r$ and $r \times q$, where $r \ll p, q$. The final weight matrix is $W + AB$.
- In our case, the number of trainable parameters decreases from 300M to 3M.
- We use the fine-tuned RoBERTa Large model to replace the transformer part in TEQWENDY, and name it LEQWENDY.

QWENDY method

- The dimension of RoBERTa Large model is fixed (1024). We need to process the input to match this dimension.
- For both TEQWENDY and LEQWENDY, the training takes a few hours on a single PC with a good graphics card.
- The most time-consuming step is to generate enough training data (about two weeks).

QWENDY method

- Since TRENDY ranks the first in 16 methods, we only test new methods against TRENDY.
- We use the same data sets and measurements.
- TEQWENDY is better than TRENDY. LEQWENDY totally fails, although with the same amount of trainable parameters with TEQWENDY.
- QWENDY does not perform well on synthetic data, but it is the best in all 19 methods on two experimental data sets.

		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

Summary

- Review WENDY method and introduce its transformer-enhanced version, TRENDY.
- Introduce a novel GRN inference method, QWENDY, and its variants enhanced by transformers and large language models (LLMs).
- Transformer encoder is a powerful tool for general tasks. LLMs are trained with natural language, not very helpful to science. We should train different large foundation models for different subjects and fine-tune on specific tasks.

References

Xueying Tian, Yash Patel, and Yue Wang. (2024). "TRENDY: gene regulatory network inference enhanced by transformer." bioRxiv, 2024.10.14.618189, accepted by Bioinformatics.
Yue Wang, and Xueying Tian. (2025). "QWENDY: gene regulatory network inference enhanced by large language model and transformer." bioRxiv, 2025.02.22.639640.

Code files: https://github.com/YueWangMathbio/TRENDY
https://github.com/YueWangMathbio/QWENDY