Estimating graphical models for count data with applications to single-cell gene network

Feiyi Xiao
School of Mathematical Sciences
Peking University
xiaofeiyi1217@pku.edu.cn

Junjie Tang
School of Mathematical Sciences
Peking University
junjie.tang@pku.edu.cn

Huaying Fang
Beijing Advanced Innovation Center for Imaging Theory and Technology
Capital Normal University
huayingfang@hotmail.com

Ruibin Xi
School of Mathematical Sciences
Center for Statistical Science
Peking University
ruibinxi@math.pku.edu.cn

Abstract
Graphical models such as Gaussian graphical models have been widely applied for direct interaction inference in many different areas. In many modern applications, such as single-cell RNA sequencing (scRNA-seq) studies, the observed data are counts and often contain many small counts. Traditional graphical models for continuous data are inappropriate for network inference of count data. We consider the Poisson log-normal (PLN) graphical model for count data and the precision matrix of the latent normal distribution represents the network. We propose a two-step method PLNet to estimate the precision matrix. PLNet first estimates the latent covariance matrix using the maximum marginal likelihood estimator (MMLE) and then estimates the precision matrix by minimizing the lasso-penalized D-trace loss function. We establish the convergence rate of the MMLE of the covariance matrix and further establish the convergence rate and the sign consistency of the proposed PLNet estimator of the precision matrix in the high dimensional setting. Importantly, although the PLN model is not sub-Gaussian, we show that the PLNet estimator is consistent even if the model dimension goes to infinity exponentially as the sample size increases. The performance of PLNet is evaluated and compared with available methods using simulation and gene regulatory network analysis of real scRNA-seq data.

1 Introduction
Gaussian graphical model (GGM) [15, 28] is widely used for understanding the complex interactions of the observed variables in various fields [26, 14, 23]. GGM assumes that each sample is drawn from a multivariate Gaussian distribution, in which the precision matrix (i.e., the inverse of the covariance matrix) represents the network. In GGM, two nodes are connected if the corresponding

*Equal contribution. Corresponding author: Ruibin Xi (ruibinxi@math.pku.edu.cn)

A few algorithms have been developed for estimating the precision matrix of the PLN model in high-dimensional settings. Compared with GGM, the likelihood of the PLN model is more complicated since it involves a high-dimensional integration and does not have a closed form. Maximizing the penalized log-likelihood of the PLN model is very difficult. Wu et al. [24] used Monte Carlo methods to approximate the log-likelihood and estimate the precision matrix by maximizing the penalized approximated log-likelihood of the PLN model. Chiquet et al. [5] developed a computationally more appealing method based on the variational approximation. However, these methods are all based on approximations of the log-likelihood and the accuracies of these approximations need to be further elaborated. More importantly, no convergence theory has been developed for these precision matrix estimators.

In this article, we propose a two-step approach named PLNet that first estimates the covariance matrix of the latent log-normal variables in the PLN model using the maximum marginal likelihood estimator (MMLE) and then estimates the sparse precision matrix by minimizing the lasso penalized D-trace loss [30]. One advantage of this two-step approach is that it avoids computing high-dimensional numerical integration that is involved in the log-likelihood of the multivariate PLN model. Minimizing the penalized D-trace loss is computationally cost-effective. Thus, this estimator
We estimate to avoid high dimensional integrations. Let \( Y_{ij} \) be the \( j \)th column of \( Y \). For \( 1 \leq j \leq p \), we estimate \( \mu^* \) and \( \Sigma^* \) by maximizing the marginal log-likelihood of \( Y_{ij} \) and denote the estimator as \( \hat{\mu}_j \) and \( \hat{\sigma}_{jj} \), and we estimate \( \sigma_{jk} \) by maximizing the marginal log-likelihood of \( (Y_{ij}, Y_{kj}) \) and

2.3 The maximum marginal likelihood estimator

We estimate \( \mu^* = [\mu^*_j]_{1 \leq j \leq p} \) and \( \Sigma^* = [\sigma^*_jk]_{1 \leq j, k \leq p} \) by maximizing the marginal log-likelihood to avoid high dimensional integrations. Let \( Y_j \) be the \( j \)th column of \( Y \). For \( 1 \leq j \leq p \), we estimate \( \mu^*_j \) and \( \sigma^*_jk \) by maximizing the marginal log-likelihood of \( Y_{ij} \) and denote the estimator as \( \hat{\mu}_j \) and \( \hat{\sigma}_{jj} \), and we estimate \( \sigma_{jk} \) by maximizing the marginal log-likelihood of \( (Y_{ij}, Y_{kj}) \) and

2 Methods

2.1 Notation

For a column vector \( a \) with the \( i \)-th entry \( a_i \), \( \|a\|_2 = (a^Ta)^{1/2} \) and \( \|a\|_1 = \sum |a_i| \). For a matrix \( A \) with the \((i, j)\)-th entry \( A_{ij} \), \( \|A\|_\infty = \max_{i, j} |A_{ij}| \) be the element-wise \( \ell_\infty \)-norm, \( \|A\|_1 = \sum_{i, j} |A_{ij}| \) be the \( l_1 \)-norm, \( \|A\|_1,\infty \) is the indicator function, \( \text{tr} (A) = \sum_i A_{ii} \) be the trace, and \( A \succeq 0 \) means \( A \) is positive semi-definite.

2.2 The PLN graphical model

Let \( Y = \{Y_{ij}\}_{1 \leq i \leq n, 1 \leq j \leq p} \) be the observed count matrix, \( Y_i = (Y_{i1}, \ldots, Y_{ip})^T \) be the observed count data of the \( i \)th sample, and \( X_i = (X_{i1}, \ldots, X_{ip})^T \) be the latent random vector. In scRNA-seq data, \( Y_{ij} \) and \( X_{ij} \) are the observed expression and the underlying "true" expression of the \( j \)th gene in the \( i \)th cell, respectively. We assume that conditional on \( X_i \), elements of \( Y_i \) are independent Poisson random variables with mean \( S_i X_{ij} (1 \leq j \leq p) \), where \( S_i \) is a scaling factor sampled from one distribution \( g(S) \) and we define \( S = (S_1, \ldots, S_n)^T \) as the scaling vector. In scRNA-seq data, \( S_i \) corresponds to the library size of the \( i \)th cell. The library size is related to the total sequencing reads and can be estimated by the sum of counts within each cell or other methods [12, 13, 21]. The\n
\[ \log(X_i) = (\log(X_{i1}), \ldots, \log(X_{ip}))^T \]

follows a multivariate normal random vector with mean \( \mu^* \) and covariance \( \Sigma^* \). The precision matrix \( \Theta^* = (\Sigma^*)^{-1} \) represents the network. In summary, we have the following graphical model for count data, for \( 1 \leq i \leq n \),

\[ Y_i | X_i \sim \prod_{j=1}^p \text{Poisson}(S_i X_{ij}), \]
\[ \log(X_i) \sim \mathcal{N}(\mu^*, (\Theta^*)^{-1}) \]

We denote this PLN distribution by \( Y_i \sim \text{PLN}(S_i; \mu^*, \Sigma^*) \). The above PLN model is a little different from the classical form [11], in which \( S_i = 1 \) for \( 1 \leq i \leq n \). We assume that the network \( \Theta^* \) is sparse and therefore we could maximize the penalized log-likelihood to estimate \( \Theta^* \). However, the likelihood function in the PLN model involves a \( p \)-dimensional integration and is difficult to be computed, especially when \( p \) is large. Chiquet et al. developed a variational algorithm called VPLN to maximize the penalized log-likelihood [5]. Although the variational method is computationally more feasible than directly maximizing the penalized log-likelihood, the estimator’s theoretical properties are difficult to be obtained.

We develop a two-step estimator called PLNet that is computationally efficient and has good theoretical properties. We first use the MMLE method to derive an estimator \( \hat{\Sigma} \) for \( \Sigma^* \). Then, we apply the D-trace method [30] with the covariance estimator \( \hat{\Sigma} \) to estimate the sparse precision matrix \( \Theta^* \). We show that the derived estimator \( \hat{\Theta} \) is a consistent estimator of \( \Theta^* \) even when the model dimension goes to infinity exponentially as the sample size increases.
denote the estimator as \( \hat{\sigma}_{jk} \). The specific forms of these marginal log-likelihoods and optimization problems are shown in the Supplemental Material (S2.1). Further, we define the MMLE for \( \mu^* \), \( \Sigma^* \) as \( \hat{\mu} = [\hat{\mu}_j]_{1 \leq j \leq p^*} \) and \( \hat{\Sigma} = [\hat{\sigma}_{jk}]_{1 \leq j,k \leq p^*} \). We apply the Newton-Raphson algorithm to maximize the marginal log-likelihood with the initial values taken as the moment estimator \( \hat{\mu}^m \) and \( \hat{\Sigma}^m \) in the Supplemental Material (S1.1). The explicit forms of the first and second-order partial derivatives of the marginal log-likelihood function are shown in the Supplemental Material (S2.1). The integrations involved in optimizations can be well approximated by the method described in [1][2].

The above MMLE \( \hat{\Sigma} \) may not be positive semi-definite. However, the D-trace method requires the input covariance matrix estimator to be positive semi-definite to guarantee the convexity of the loss function. We project \( \hat{\Sigma} \) to the space of positive semi-definite matrices and identify \( \hat{\Sigma} \) that is closest to \( \hat{\Sigma} \) in the space, i.e.,

\[
\hat{\Sigma} = \arg \min_{\hat{\Sigma} \succ 0} \left\| A - \hat{\Sigma} \right\|_\infty.
\]

Then we add a small positive definite matrix to \( \hat{\Sigma} \) and get a positive-definite estimator \( \tilde{\Sigma} \) of \( \Sigma^* \),

\[
\tilde{\Sigma} = \hat{\Sigma} + \left\| \tilde{\Sigma} - \hat{\Sigma} \right\|_\infty I,
\]

where \( I \) is the identity matrix. The optimization problem \( \tilde{\Sigma} \) for \( \tilde{\Sigma} \) can be solved by a splitting conic solver [8].

With the covariance matrix estimator \( \tilde{\Sigma} \), we apply the D-trace method to estimate the precision matrix,

\[
\tilde{\Theta} = \arg \min_{\Theta \succ 0} \left\{ \frac{1}{2} \text{tr} \left( \tilde{\Sigma} \Theta^2 \right) - \text{tr} (\Theta) + \lambda_n \left\| \Theta \right\|_{1, \text{off}} \right\}.
\]

The above optimization problem \( \tilde{\Theta} \) can be solved by an alternating direction method of multipliers [30]. In this paper, we use a more efficient algorithm developed in [22] to calculate \( \tilde{\Theta} \). We can also incorporate the prior information about the gene regulatory relationship to improve the performance of network inference. Let \( E \) be the set of gene pairs that cannot have direct interactions from the prior information, we can incorporate the prior information \( E \) by considering the following constrained optimization problem,

\[
\tilde{\Theta} = \arg \min_{\Theta \succ 0} \left\{ \frac{1}{2} \text{tr} \left( \tilde{\Sigma} \Theta^2 \right) - \text{tr} (\Theta) + \lambda_n \left\| \Theta \right\|_{1, \text{off}} : \Theta_{ij} = 0 \text{ for } (i, j) \in E \right\}.
\]

The tuning parameter \( \lambda_n \) is selected by minimizing the following approximate Bayesian information criterion (BIC) [31],

\[
\left\| \frac{1}{2}(\tilde{\Theta} \tilde{\Sigma} + \tilde{\Sigma} \tilde{\Theta}) - I \right\|_F + \left( \left\| \tilde{\Theta} \right\|_0 \log n \right) / n.
\]

### 3 Theoretical properties

In this section, we establish the theoretical properties in the high dimensional setting. We first prove that \( \tilde{\Sigma} \) is a consistent estimator of \( \Sigma^* \). The convergence rate for \( \tilde{\Sigma} \) is similar to that of the sample covariance matrix for random variables with sub-Gaussian distribution. Then, under the same irrepresentability condition of the D-trace method in [30], we derive the edge recovery property and consistency for the PLNet estimator \( \tilde{\Theta} \).

#### 3.1 Notation

Let \( G = \{ (i, j) \mid \Theta_{ij}^* \neq 0 \} \) be positions of non-zero elements in \( \Theta^* \), \( G^c \) be the complement set of \( G \), \( d \) be the maximum node degree in \( \Theta^* \), \( s = \| \Theta^* \|_0 \), and \( \theta_{\min} = \min_{(i,j) \in G} \| \Theta_{ij}^* \| \) be the minimal absolute value of nonzero elements of \( \Theta^* \). Let \( \lambda_{\max}(A) \) and \( \lambda_{\min}(A) \) be the largest and smallest eigenvalues of a symmetric matrix \( A \). We define \( \Gamma^* = \Gamma(\Sigma^*) = (\Sigma^* \otimes I + I \otimes \Sigma^*) / 2 \), \( \Gamma = \Gamma(\tilde{\Sigma}) \), where \( A \otimes B \) is the Kronecker product. For a \( p \times p \) matrix \( A \), the row \( (i-1)p + j \) and column \( (k-1)p + l \) of \( \Gamma(A) \) is \( \Gamma(A)_{(i,j),(k,l)} = (A_{ik}1_{j=l} + A_{jl}1_{i=k}) / 2 \). For two subsets \( T_1 \) and \( T_2 \)
of \{ (i, j) | 1 \leq i, j \leq p \}, we define \( \Gamma(A)_{T_1, T_2} \) be the submatrix of \( \Gamma(A) \) whose rows and columns indexed by \( T_1 \) and \( T_2 \), respectively. Other notations are as follows,

\[
\gamma = 1 - \max_{(i,j) \in G^c} \left\| \Gamma^*_{(i,j),G} (\Gamma^*_{G,G})^{-1} \right\|_1,
\]

\[
k_\Gamma = \left\| (\Gamma^*_{G,G})^{-1} \right\|_{1,\infty}, k_{\Sigma} = \left\| \Sigma^* \right\|_{1,\infty}.
\]

### 3.2 The irrepresentability condition and rate of convergence

We first present the necessary irrepresentability condition for establishing the rate of convergence for the estimator in PLNet. This irrepresentability condition is from the D-trace method in [30].

**Condition 1 (Irrepresentability condition)** There exists \( 0 < \alpha < 1 \), such that

\[
\max_{(i,j) \in G^c} \left\| \Gamma^*_{(i,j),G} (\Gamma^*_{G,G})^{-1} \right\|_1 \leq 1 - \alpha.
\]

The irrepresentability condition is equivalent to \( \gamma \geq \alpha > 0 \). We also need a boundedness condition in the PLN model.

**Condition 2 (Boundedness condition)** There exist positive constants \( M_1, M_2, M_3, M_4 \), and \( M_5 \), such that the distribution of \( S_i \) has bounded support \([M_1, M_2]\), \( \max_{1 \leq j, k \leq p} \{ |\mu_j^*|, |\sigma_{jk}^*| \} \leq M_3 \) and \( M_4 \leq \lambda_{\min}(\Sigma^*) \leq \lambda_{\max}(\Sigma^*) \leq M_5 \).

Based on the boundedness condition, we can establish the convergence rate for the covariance matrix estimator \( \hat{\Sigma} \).

**Theorem 1 (Rate of convergence for \( \hat{\Sigma} \))** Under Condition 2, there exist positive constants \( A, B \), such that for any \( \epsilon > 0 \),

\[
pr \left( \left\| \hat{\Sigma} - \Sigma^* \right\|_\infty \geq \epsilon \right) \leq p^2 A \exp(-B n^{\epsilon^2}).
\]

After plugging \( \hat{\Sigma} \) into the lasso penalized D-trace loss, we get a consistent estimator \( \hat{\Theta} \) that converges to \( \Theta^* \) in several matrix norms.

**Theorem 2 (Rate of convergence for \( \hat{\Theta} \))** Under the irrepresentability condition and the boundedness condition, there exist constants \( A, B \), such that for some \( \eta > 2 \), if

\[
n > B^{-1}(\eta \log p + \log A) \max \left\{ 12dk_\Gamma, 12\gamma^{-1}(k_\Sigma k_T^2 + k_\Gamma), \{ 12\gamma^{-1} (k_\Sigma k_T^2 + k_\Gamma) + 5dk_\Gamma^2 \}, \min \left\{ s^{1/2}, d + 1 \right\} \{ 12\gamma^{-1} (k_\Sigma k_T^2 + k_\Gamma) + 5dk_\Gamma^2 \} \lambda_{\min}(\Theta^*) \right\}^2,
\]

and

\[
\lambda_n = 12\gamma^{-1} (k_\Sigma k_T^2 + k_\Gamma) B^{-1/2} (\eta \log p + \log A)^{1/2} n^{-1/2},
\]

then with probability \( 1 - p^{2-n} \),

\[
\left\| \hat{\Theta} - \Theta^* \right\|_\infty \leq (12\gamma^{-1} (k_\Sigma k_T^2 + k_\Gamma) + 5dk_\Gamma^2) B^{-1/2} (\eta \log p + \log A)^{1/2} n^{-1/2},
\]

\[
\left\| \hat{\Theta} - \Theta^* \right\|_F \leq s^{1/2} \left\{ 12\gamma^{-1} (k_\Sigma k_T^2 + k_\Gamma) + 5dk_\Gamma^2 \right\} B^{-1/2} (\eta \log p + \log A)^{1/2} n^{-1/2},
\]

\[
\left\| \hat{\Theta} - \Theta^* \right\|_2 \leq \min \left\{ s^{1/2}, d + 1 \right\} \left\{ 12\gamma^{-1} (k_\Sigma k_T^2 + k_\Gamma) + 5dk_\Gamma^2 \right\} B^{-1/2} (\eta \log p + \log A)^{1/2} n^{-1/2}.
\]

Meanwhile, with a high probability, the sign of the sparse precision matrix \( \Theta^* \) can be recovered by \( \hat{\Theta} \). We have the following theorem about the sign consistency of \( \hat{\Theta} \).
Theorem 3 (Sign consistency for $\hat{\Theta}$) Under the conditions in Theorem 2, for some $\eta > 2$, choosing the same $n$ and $\lambda_n$ in Theorem 2 then with probability $1 - p^{2-\eta}$, $\hat{\Theta}$ recovers all zeros and nonzeros in $\Theta^*$. 

The rate-of-convergence and sign consistency results in Theorem 2 and 3 are closely related to the tail probability of the MMLE estimator $\hat{\Sigma}$ in Theorem 1. The boundedness condition 2 is assumed to guarantee the convergence of the MMLE. Ignoring the complicated constants in theorems, for any $\eta > 2$, if we have $n > C\eta \log p$, or in other words, if $p$ tends to infinity not faster than exponential of the sample size $n$, $\hat{\Theta}$ is a consistent estimator of $\Theta^*$. Especially, the rate of convergence for $\hat{\Theta}$ is $O\left(\left[\eta (\log p) / n\right]^{1/2}\right)$ under $l_\infty$-norm.

4 Simulation studies

4.1 Simulation settings

We conduct simulations to evaluate the performance of PLNet and compare it with the available network inference methods including VPLN, glasso, and an estimator called PLNet-MOM by plugging the moment estimator $\hat{\Sigma}^m$ into D-trace loss (See Supplemental Material (S1.1)). Both PLNet and VPLN are designed to estimate the precision matrix for count data in the PLN model. The glasso algorithm is a classical approach for continuous data in GGM and we apply glasso to the logarithmic transformation of the normalized data, which is defined as $\tilde{Y}_{ij} = (Y_{ij} + 1) / \hat{S}_i$, $1 \leq i \leq n$, where $\hat{S}_i$ is the estimated library size of $i$th cell. In all simulations, we estimate the library size by the total sum scaling (i.e. $\hat{S}_i = \sum_{j=1}^p Y_{ij}$), which is a classical normalization method for scRNA-seq and is defined as the sum of counts within each cell.

We consider 48 different simulation scenarios, which are 2 sample size setups ($n = 500, 2000$) × 3 dimension setups ($p = 100, 300, 500$) × 2 dropout levels (low and high) × 4 graph structures and simulate count data from the PLN model, where the dropout level represents the proportion of zeros in the count matrix. Details of the simulation settings are shown in the Supplementary Material (S3.1). In each scenario, we independently repeat simulations 100 times. The four graph structures are as follows.

1. Banded Graph: Pairs $(i, j)$ of nodes are connected if $|i - j| \leq 2$, $i \neq j$. All nonzero edges are set as 0.3.
2. Random Graph: Pairs of nodes are connected with probability 0.1. The nonzero edges are set as 0.3 with probability 0.8 and as −0.3 with probability 0.2.
3. Scale-free Graph: The Barabasi-Albert model is used to generate a scale-free graph with power 1. The nonzero edges are set as 0.3.
4. Blocked Graph: The nodes are divided into 5 blocks of equal sizes. Pairs of nodes in the same block are connected with probability 0.1 and the nonzero edges are set as 0.3. Different blocks have no edge connection.

The diagonal elements of precision matrices are set as 1 plus a positive number to guarantee positive definiteness.

4.2 Performance comparison

Table 1 and Table 2 show the area under the precision and recall curve (AUPR) of each estimator. AUPRs are calculated by varying the tuning parameters (i.e. the penalty parameters of the four estimators). As expected, the AUPR decreases as the number of genes increases. AUPRs in the high-dropout cases are generally smaller than that in the low-dropout cases. PLNet is the most robust estimation among these four estimators and outperforms PLNet-MOM, VPLN, and glasso in most simulation settings in AUPR, especially for the settings with high dropouts. For example, for $n = 2000$, $p = 100$, PLNet achieves an AUPR of 0.88 for the random graph under the scenario of the high dropout rate, while PLNet-MOM, VPLN, and glasso only have AUPRs of 0.82, 0.69, and 0.18, respectively. Furthermore, PLNet-MOM outperforms glasso in most settings and outperforms
Table 1: Comparisons of PLNet, VPLN, glasso, and PLNet-MOM in terms of the area under precision and recall curve (AUPR) on simulation results for $n = 500$. The results are averages over 100 replicates with standard deviations in brackets.

<table>
<thead>
<tr>
<th>Sample size $n$</th>
<th>Dimension $p$</th>
<th>Dropout</th>
<th>PLNet</th>
<th>VPLN</th>
<th>glasso</th>
<th>PLNet-MOM</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n = 500$</td>
<td>$p = 100$</td>
<td>Low</td>
<td>0.79(0.03)</td>
<td>0.68(0.06)</td>
<td>0.34(0.04)</td>
<td>0.62(0.03)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>High</td>
<td>0.51(0.04)</td>
<td>0.44(0.04)</td>
<td>0.05(0.01)</td>
<td>0.43(0.03)</td>
</tr>
<tr>
<td>$n = 500$</td>
<td>$p = 300$</td>
<td>Low</td>
<td>0.6(0.06)</td>
<td>0.6(0.02)</td>
<td>0.44(0.03)</td>
<td>0.33(0.02)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>High</td>
<td>0.31(0.04)</td>
<td>0.27(0.02)</td>
<td>0.04(0.01)</td>
<td>0.21(0.01)</td>
</tr>
<tr>
<td>$n = 500$</td>
<td>$p = 500$</td>
<td>Low</td>
<td>0.44(0.08)</td>
<td>0.22(0.03)</td>
<td>0.22(0.03)</td>
<td>0.21(0.02)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>High</td>
<td>0.4(0.07)</td>
<td>0.35(0.01)</td>
<td>0.2(0.05)</td>
<td>0.16(0.04)</td>
</tr>
</tbody>
</table>

Table 2: Comparisons of PLNet, VPLN, glasso, and PLNet-MOM in terms of the area under precision and recall curve (AUPR) on simulation results for $n = 2000$. The results are averages over 100 replicates with standard deviations in brackets.

<table>
<thead>
<tr>
<th>Sample size $n$</th>
<th>Dimension $p$</th>
<th>Dropout</th>
<th>PLNet</th>
<th>VPLN</th>
<th>glasso</th>
<th>PLNet-MOM</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n = 2000$</td>
<td>$p = 100$</td>
<td>Low</td>
<td>0.99(0.01)</td>
<td>0.78(0.08)</td>
<td>0.55(0.06)</td>
<td>0.94(0.03)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>High</td>
<td>0.96(0.01)</td>
<td>0.69(0.07)</td>
<td>0.45(0.07)</td>
<td>0.82(0.05)</td>
</tr>
<tr>
<td>$n = 2000$</td>
<td>$p = 300$</td>
<td>Low</td>
<td>0.99(0.01)</td>
<td>0.78(0.08)</td>
<td>0.55(0.06)</td>
<td>0.94(0.03)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>High</td>
<td>0.94(0.01)</td>
<td>0.67(0.1)</td>
<td>0.4(0.03)</td>
<td>0.8(0.03)</td>
</tr>
<tr>
<td>$n = 2000$</td>
<td>$p = 500$</td>
<td>Low</td>
<td>0.98(0.01)</td>
<td>0.78(0.08)</td>
<td>0.55(0.06)</td>
<td>0.94(0.03)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>High</td>
<td>0.88(0.04)</td>
<td>0.67(0.1)</td>
<td>0.4(0.03)</td>
<td>0.8(0.03)</td>
</tr>
</tbody>
</table>

7
VPLN under the scenario of large sample sizes. The results of the area under the receiver operating characteristic curve (AUC) are similar and shown in Supplementary Table S1-S2.

To further demonstrate the performance of PLNet, we visualize the mean networks predicted by the four methods tuned by BIC criterion for the banded graph with \( n = 2000, p = 100 \) over the 100 simulations (Figure 1). More specifically, we calculate the relative frequency \( F_{ij} \) that an algorithm reports edges over the 100 simulations. For positions \((i, j)\) with \( \Theta_{ij} \neq 0 \), \( F_{ij} \) is the proportion that an algorithm correctly recovers the edge in the 100 simulations, while for positions \((i, j)\) with \( \Theta_{ij} = 0 \), \( F_{ij} \) is the proportion that an algorithm falsely predicts edges between nodes \( i \) and \( j \) in the 100 simulations. We plot the relative frequency matrices of the four methods in Figure 1. We clearly see that PLNet is able to detect more true positives while having fewer false positives than other methods, especially for the high dropout case. The results of mean predicted networks for the other three graphs with \( n = 2000, p = 100 \) are similar and shown in Supplementary Figure S1-S3.

![Figure 1: The mean networks predicted by PLNet, VPLN, glasso, and PLNet-MOM for the banded graph with 100 nodes and \( n = 2000 \). False edges are colored in red and true edges are in blue. The left panel is the true network matrix for reference.](image)

Supplementary Table S3-S4 show the computational time of the four methods. PLNet is computationally more efficient than VPLN in most simulation settings. Also, the computational efficiency of PLNet is roughly the same for different sample sizes, while the computational complexity of VPLN maintains a linear relationship with sample size. The detailed comparisons between the computational efficiency of the four methods are shown in the Supplementary Material (S3.2.2).

## 5 Application to a scRNA-seq dataset

We apply PLNet and VPLN to infer the gene regulatory network of CD14+ Monocytes profiled in Kang et al. [10]. The single cells are profiled in two different conditions, IFN-\( \beta \)-treated and controlled. IFN-\( \beta \) is a cytokine in the interferon family that influences the transcriptional profiles for many genes, especially those in the JAK/STAT pathway [16]. We focus on the IFN-\( \beta \)-treated cells (2147 cells) and use the top 200 highly variable genes that are used in [20] for network analysis.

We compare networks of PLNet and VPLN with parameters tuned such that the network densities are around 5%. Gene Ontology (GO) analysis [11] shows that the 200 genes mainly involve in 4 biological processes, including "Cytokine-mediated signaling pathway" (Module \( M_1 \)), "Neutrophil-mediated immunity" (Module \( M_2 \)), "Cellular protein metabolic process" (Module \( M_3 \)), and "Proteolysis" (Module \( M_4 \)). Figure 2 shows the predicted networks of the genes in the 4 modules by PLNet and VPLN, where the colors represent the partial correlations between genes. The partial correlation given by PLNet between genes \( i \) and \( j \) is defined as \(-\hat{\Theta}_{ij}/(\hat{\Theta}_{ii} \hat{\Theta}_{ij})^{1/2}\). The partial correlation given by VPLN is defined similarly. We clearly see that the network given by PLNet tends to have more connections within the modules than VPLN. To see this more clearly, for each module \( M_k \), we calculate the ratio between within-module and between-module connections \( R(M_k) = \Sigma_{i,j \in M_k} W_{ij}/\Sigma_{i \in M_k, j \notin M_k} W_{ij} \), where the weights \( W_{ij} \) are set as the absolute partial correlation between genes \( i \) and \( j \) (weighted within-between connection ratio) or are set as 1 and 0 depending on whether genes \( i \) and \( j \) are connected (unweighted within-between connection ratio). The within-
between connection ratios of PLNet are much larger than that of VPLN in most cases (Table 3). Similar results also hold for networks of other different densities that are shown in the Supplementary Material (S4.1).

Table 3: The within-between connection ratios of the 4 modules in the networks estimated by PLNet and VPLN tuned such that the network densities are around 5%.

<table>
<thead>
<tr>
<th>Type</th>
<th>Method</th>
<th>Module M1</th>
<th>Module M2</th>
<th>Module M3</th>
<th>Module M4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weighted</td>
<td>PLNet</td>
<td>0.681</td>
<td>0.231</td>
<td>0.582</td>
<td>0.387</td>
</tr>
<tr>
<td></td>
<td>VPLN</td>
<td>0.621</td>
<td>0.381</td>
<td>0.459</td>
<td>0.259</td>
</tr>
<tr>
<td>Unweighted</td>
<td>PLNet</td>
<td>0.533</td>
<td>0.129</td>
<td>0.419</td>
<td>0.383</td>
</tr>
<tr>
<td></td>
<td>VPLN</td>
<td>0.515</td>
<td>0.200</td>
<td>0.170</td>
<td>0.213</td>
</tr>
</tbody>
</table>

To further compare the performance of PLNet and VPLN on the real scRNA-seq data analysis, we construct a silver standard based on an available regulatory network database obtained from ChIP-seq experiments (the hTFtarget database [29]). The silver standard consists of edges between the 200 highly variable genes and 36 transcription factors (TF) from the top 500 highly variable genes. We consider 26 additional TFs from the top 500 highly variable genes because the top 200 highly variable genes only contain 10 TFs and the silver standard constructed only using these genes is too sparse to be a reliable silver standard. We compare networks estimated by PLNet and VPLN with their parameters tuned such that the network densities are at the same levels (1% to 10%). Table 4 shows the number of true edges in the silver standard detected by PLNet and VPLN at different network densities. Clearly, we see that PLNet detects more true edges at the same density levels than VPLN, suggesting that PLNet has a higher true discovery rate than VPLN.

Table 4: The number of true edges estimated by two methods with different density levels.

<table>
<thead>
<tr>
<th>Density</th>
<th>0.01</th>
<th>0.02</th>
<th>0.03</th>
<th>0.04</th>
<th>0.05</th>
<th>0.06</th>
<th>0.07</th>
<th>0.08</th>
<th>0.09</th>
<th>0.10</th>
</tr>
</thead>
<tbody>
<tr>
<td>PLNet</td>
<td>8</td>
<td>16</td>
<td>23</td>
<td>35</td>
<td>41</td>
<td>44</td>
<td>62</td>
<td>73</td>
<td>81</td>
<td>92</td>
</tr>
<tr>
<td>VPLN</td>
<td>2</td>
<td>5</td>
<td>7</td>
<td>12</td>
<td>20</td>
<td>27</td>
<td>36</td>
<td>48</td>
<td>62</td>
<td>62</td>
</tr>
</tbody>
</table>

6 Discussion

In this paper, we consider the PLN graphical model for count data. This model has an intuitive explanation for single-cell gene regulatory network analysis. The Poisson layer is to capture the
technical noises and the log-normal layer is to model the biological fluctuations in single cells. Gene regulatory network is represented by the precision matrix of the latent log-normal model. To estimate the precision matrix, we propose a two-step estimator PLNet, using the MMLE to estimate the covariance matrix and then minimizing the penalized D-trace loss to estimate the precision matrix. The simplicity of this estimation procedure allows us to establish the consistency theory for the proposed PLNet estimator for the high dimensional setting. The numerical analysis also shows that the PLNet method outperforms available methods.

The proposed method can be generalized in several ways. First, we may consider the zero-inflated PLN (ZIPLN) model to account for the excessed number of zeros that cannot be modeled by PLN models. We could also use the MMLE method to estimate the covariance matrix of the ZIPLN model, but the convergence theory is more difficult to develop. Secondly, a straightforward generalization is the differential network analysis. We could use the same method to estimate covariance matrices in two different conditions and use the D-trace loss developed in earlier work [27] for differential network analysis in single cells. Thirdly, the mean parameters of the latent log-normal random variable can depend on covariates. In such model, we could first use regression to estimate the mean parameters and derive the corresponding moment estimator of the covariance matrix. Another generalization is gene regulatory network analysis of mixtures of cell populations. Different cell populations may have different gene regulatory networks and we could jointly model the mixture and infer the gene regulatory networks for all cell populations. Finally, the current method is based on parametric inference. If the model is mis-specified, network inference could be misleading. To mitigate this problem, we can consider a semi-parametric model by replacing the latent log-normal model with a Gaussian copula graphical model, but how to do network inference with this semi-parametric model is still a challenge that needs to be overcome in the future.

**Acknowledgments and Disclosure of Funding**

We thank Changhu Wang and Siyuan Huang for helpful discussions. This work was supported by the National Natural Science Foundation of China [No. 11971039 to R.X.], the National Key Basic Research Program of China [2020YFE0204000 to R.X.], and Sino-Russian Mathematics Center.

**References**


Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 6.
   (c) Did you discuss any potential negative societal impacts of your work? [No] Our work do not have any negative societal impact.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Conditions 1 and 2 and Theorems.
   (b) Did you include complete proofs of all theoretical results? [Yes] See Supplemental Material for proofs.

3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See Supplemental Material for codes.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 4.1 for Simulation settings and equation 5 for hyperparameter chosen.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See the standard deviations in Table 1 and 2 for error bars.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Supplemental Material (S3.2.2).

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
(a) If your work uses existing assets, did you cite the creators? [Yes] See lines 105, 114, 128, 135, 138, 177, 195, 230 for citations.

(b) Did you mention the license of the assets? [Yes] All the assets are open available.

(c) Did you include any new assets either in the supplemental material or as a URL? [No]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No]

5. If you used crowdsourcing or conducted research with human subjects...

   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]