mathematics

MDPI

Article

Fitting Penalized Estimator for Sparse Covariance Matrix with Left-Censored Data by the EM Algorithm

Shanyi Lin [1], Qian-Zhen Zheng [2], Laixu Shang [2], Ping-Feng Xu [3,4,*] and Man-Lai Tang [5,*]

[1] School of Mathematics and Statistics, Changchun University of Technology, Changchun 130012, China; linshanyi@ccut.edu.cn
[2] College of Education, Zhejiang Normal University, Jinhua 321004, China; zhengqz@zjnu.edu.cn (Q.-Z.Z.); shanglx3107@zjnu.edu.cn (L.S.)
[3] Academy for Advanced Interdisciplinary Studies & Key Laboratory of Applied Statistics of MOE, Northeast Normal University, Changchun 130024, China
[4] Shanghai Zhangjiang Institute of Mathematics, Shanghai 201203, China
[5] Department of Physics, Astronomy and Mathematics, School of Physics, Engineering & Computer Science, University of Hertfordshire, Hertfordshire AL10 9AB, UK
* Correspondence: xupf900@nenu.edu.cn (P.-F.X.); m.l.tang@herts.ac.uk (M.-L.T.)

Abstract: Estimating the sparse covariance matrix can effectively identify important features and patterns, and traditional estimation methods require complete data vectors on all subjects. When data are left-censored due to detection limits, common strategies such as excluding censored individuals or replacing censored values with suitable constants may result in large biases. In this paper, we propose two penalized log-likelihood estimators, incorporating the L_1 penalty and SCAD penalty, for estimating the sparse covariance matrix of a multivariate normal distribution in the presence of left-censored data. However, the fitting of these penalized estimators poses challenges due to the observed log-likelihood involving high-dimensional integration over the censored variables. To address this issue, we treat censored data as a special case of incomplete data and employ the Expectation Maximization algorithm combined with the coordinate descent algorithm to efficiently fit the two penalized estimators. Through simulation studies, we demonstrate that both penalized estimators achieve greater estimation accuracy compared to methods that replace censored values with constants. Moreover, the SCAD penalized estimator generally outperforms the L_1 penalized estimator. Our method is used to analyze the proteomic datasets.

Keywords: sparse covariance matrix; Expectation Maximization algorithm; penalized estimator; left-censored data

MSC: 62H12; 62-08

Academic Editor: Manuel Alberto M. Ferreira

Received: 23 December 2024
Revised: 21 January 2025
Accepted: 25 January 2025
Published: 27 January 2025

Citation: Lin, S.; Zheng, Q.-Z.; Shang, L.; Xu, P.-F.; Tang, M.-L. Fitting Penalized Estimator for Sparse Covariance Matrix with Left-Censored Data by the EM Algorithm. *Mathematics* **2025**, *13*, 423. https://doi.org/10.3390/math13030423

1. Introduction

Estimating covariance matrices is a critical problem in modern multivariate data analysis, with broad applications in fields such as economics [1], finance [2], biology [3], and social networks [4], among others. The elements of these matrices provide valuable insights into variance, correlation, and covariance. However, it is challenging to estimate large covariance matrices, especially when the number of variables exceeds the sample size. The assumption of sparsity has been extensively developed in the context of large covariance matrix estimation. This assumption suggests that most of the off-diagonal elements in a covariance matrix are exactly zero, significantly reducing the number of free parameters in the covariance matrix.

There are many studies for sparse covariance matrix estimation. The methods based on banding [5], tapering [6,7] are proposed when a known ordering of variables is available. In the absence of information on natural ordering of variables, thresholding [8–10] is developed by setting smaller elements in the sample covariance matrix to zero. But there is no guarantee that the thresholding estimator is always positive definite. To enforce positive definiteness, Rothman [11] and Xue et al. [12] proposed the L_1 penalized Frobenius norm approaches, in which Rothman [11] included an additional logarithmic barrier term into the objective function and Xue et al. [12] used a direction-altering method. However, it is known that L_1 penalty induces bias in the estimators. To alleviate this bias effect, Wei and Zhao [13] and Wang et al. [14] imposed non-convex penalties such as smoothly clipped absolute deviation (SCAD) penalty, minimax concave penalty and an adaptive capped-L_1 penalty in the estimation of sparse covariance matrices.

Penalized likelihood techniques provide an alternative for estimating a sparse covariance matrix. Bien and Tibshirani [15] pioneered the covariance graphical lasso method, which penalized the likelihood with a lasso penalty on the elements of the covariance matrix and used a majorize–minimize approach to approximately minimize the L_1 penalized likelihood. To improve the computational efficiency and numerical stability, Wang [16] developed a coordinate descent algorithm to fit sparse covariance graphical lasso models. Xu and Lange [17] developed a likelihood-based method that regularizes the distance from the covariance estimate to a symmetric sparsity set and proposed a majorization–minimization-based algorithm to yield a sparse covariance matrix estimation. But the selection of penalty parameter in Bien and Tibshirani [15], Wang [16] and Xu and Lange [17] is carried out via computationally expensive cross-validation methods where the estimator is solved many times to choose the best-suited sparsity. To eliminate the selection of penalty parameter, Fatima et al. [18] presented a cyclic majorization–minimization-based technique to minimize the extended Bayesian information criteria with a L_0 penalty. Besides the above methods, Fatima et al. [19] proposed a two-stage procedure, which first finds zero elements in the target covariance matrix using false discovery rate multiple hypothesis testing and then estimates the non-zero elements by a block coordinate descent approach or a proximal distance approach. Sung and Lee [20] considered a spike and slab prior to introducing sparsity to the covariance matrix and proposed a sparse covariance estimation method using a block coordinate descent algorithm to determine the mode of the posterior density conditional on the structure of the covariance.

Up to this point, all these methodologies mentioned above were formulated assuming that all data are completely observed. However, in many real-world applications, particularly in fields like epidemiology [21], mass spectrometry-based metabolomics studies [22], environmental contaminant research [23] and hydrology [24], censored data due to detection limits are common, in which the value of an observation is not known exactly but rather is only known to be above or below a specific value. Therefore, the detection limits of measurement techniques may make it impossible to achieve data completeness.

A common and simple way to deal with censoring is to delete all study subjects that contain at least one censored value. However, ignoring these informative censored values often leads to severe bias and loss of precision due to effective sample size. Another common approach is to replace the left-censored value with a representative constant, e.g., the sample mean, half of the left-censored values, or the minimum of the observations, and then compute the sample covariance based on the complete data obtained. But previous studies have found all these alternative methods to be more or less biased. To reduce this estimation bias or loss of precision caused by censoring out-of-range values to the closest possible value, several algorithms were proposed for computing maximum likelihood estimation of covariance matrix under the assumption

of a normal distribution in Hoffman and Johnson [25], Jones et al. [26], Pesonen et al. [27]. Hoffman and Johnson [25] proposed a pseudo-likelihood approach by applying the MLE method to pairs of variables. Jones et al. [26] proposed a maximum pairwise pseudo-likelihood estimation of covariance matrices based on marginal likelihood. Pesonen et al. [27] proposed maximum likelihood estimation of finite-memory algorithms based on pairs of elements and estimation of covariance matrices using parallel pairs of elements. However, these methods were computationally intensive and unsuitable for the high-dimensional variable setting. Most importantly, these methods did not address the problem of sparse covariance estimation.

In this article, we focus on estimating the sparse covariance matrix in the multivariate normal model in the presence of censored data with detection limits. To obtain a sparse estimation of covariance matrix, we suggest penalizing the log-likelihood function of the censored data using the L_1 penalty and the SCAD penalty. However, the log-likelihood function involves high-dimensional integral, leading to high computational complexity. In this paper, we treat censored data as a special kind of incomplete data and we apply the Expectation Maximization (EM) algorithm [28] to optimize the penalized likelihood of censored data. In the Expectation step of the EM algorithm, we use an approximation method or Monte Carlo sampling method to calculate the Q function, i.e., the expectation of the log-likelihood function of complete data. In the maximization step of the EM algorithm, we apply the coordinate descent algorithm developed by Wang [16] to optimize the Q function with the L_1 penalty. For the SCAD penalty, we reformulate the Q-function as a series of penalized likelihood problems with a weighted L_1 penalty, using the local linear approximation (LLA) proposed by Zou and Li [29]. We then modify the coordinate descent algorithm of Wang [16] to efficiently optimize these penalized likelihood problems with the weighted L_1 penalty. We show the performance of the proposed estimators via simulation studies. We apply our method to analyze a proteomics dataset.

The remaining part of this paper is organized as follows. In Section 2, we propose two penalized estimators for sparse covariance matrix and then fit them by the EM algorithm combined with the coordinate descent algorithm. In Section 3, we give simulation studies to show the performance of the proposed method. In Section 4, we analyze a real dataset. Section 5 presents the conclusions and discusses directions for future research.

2. Sparse Covariance Matrix Estimation for Censored Data

2.1. Penalized Estimator for Censored Data

To incorporate the censoring mechanism in our framework, we follow Little and Rubin [30] and Augugliaro et al. [31]. Let $X = \left(X_1, X_2, ..., X_p\right)^\top$ be a p-dimensional random vector, that follows a multivariate normal distribution $N_p(\mu, \Sigma)$. The vector of known left-censoring values is denoted by $l = \left(l_1, \cdots l_p\right)$. We denote the censoring pattern by a p-dimensional random vector $R(X; l)$ with support set $\{0, 1\}^p$. Specifically, the jth element of $R(X; l)$ is defined as $R\left(X_j, l_j\right) = I\left(X_j < l_j\right)$, where $I(\cdot)$ denotes the indicator function. Thus, $R\left(X_j, l_j\right) = 0$ (i.e., X_j is observed) only if it is inside the interval $[l_j, +\infty)$, and $R\left(X_j, l_j\right) = 1$ (i.e., X_j is censored from below) if $X_j < l_j$.

Given a censoring pattern, we divide the set $\mathcal{V} = \{1, \ldots, p\}$ into two subsets $o = \left\{j \in \mathcal{V} \mid r_j = 0\right\}$ and $c = \left\{j \in \mathcal{V} \mid r_j = 1\right\}$. Then, x_o is the observed vector, and the observed data can be represented as (x_o, r). The density of the observed data (x_o, r) is denoted by

$$\psi(x_o, r \mid \theta) = \int_{D_c} \varphi(x_o, x_c \mid \theta) dx_c I(l_o \leq x_o), \tag{1}$$

where $\varphi(x_o, x_c \mid \theta)$ is the joint multivariate normal density depending on the parameter $\theta = (\mu, \Sigma)$ and $D_c = (-\infty, l_c)$ is the region of integration.

Consider n independent samples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ from a censored multivariate normal distribution, and we assume that l is known and fixed across the n observations. Let $\boldsymbol{r}_i$ be the ith realization of the random vector $R(\boldsymbol{x}_i; l)$, then the variables can be partitioned into two subsets $o_i = \{j \in \mathcal{V} | \boldsymbol{r}_{ij} = 0\}$, $c_i = \{j \in \mathcal{V} | \boldsymbol{r}_{ij} = 1\}$. The ith observation is the vector $(\boldsymbol{x}_{o_i}, \boldsymbol{r}_i)$, and the ith complete observation is $(\boldsymbol{x}_{o_i}, \boldsymbol{x}_{c_i})$, then we obtain the observed log-likelihood function

$$\ell_{obs}(\theta \mid \boldsymbol{x}_o, \boldsymbol{r}) = \sum_{i=1}^{n} \log \psi(\boldsymbol{x}_{o_i}, \boldsymbol{r}_i \mid \theta) = \sum_{i=1}^{n} \log \int_{D_{c_i}} \varphi(\boldsymbol{x}_{o_i}, \boldsymbol{x}_{c_i} \mid \theta) \mathrm{d}\boldsymbol{x}_{c_i} I(l_{o_i} < \boldsymbol{x}_{o_i}). \tag{2}$$

To obtain a sparse estimator of covariance matrix Σ, we suggest adding a penalty term to the log-likelihood function and the resulting penalized estimator is

$$\hat{\Sigma}_\lambda = \arg\min_{\Sigma \succ 0} -\ell_{obs}(\theta \mid \boldsymbol{x_o}, \boldsymbol{r}) + \sum_{i=1}^{p} \sum_{j=1}^{p} P_\lambda(|\Sigma_{ij}|), \tag{3}$$

where $\Sigma \succ 0$ means Σ is a positive definite matrix, Σ_{ij} is the (i, j)-element of matrix Σ for $i, j = 1, \ldots, p$, and $\lambda > 0$ is a penalty parameter. If the function $P_\lambda(|x|)$ is defined as $\lambda|x|$, the penalty term corresponds to the L_1 penalty $\lambda\|\Sigma\|_1 = \lambda \sum_{j,k} |\Sigma_{jk}|$, yielding the L_1 penalized estimator. The L_1 penalty can estimate a sparse covariance matrix by shrinking smaller elements toward zero. However, it introduces biases in the estimates for larger elements, as the penalty increases linearly with the element's magnitude. By contrast, the SCAD penalty alleviates the bias problem induced by the L_1 penalty. Similarly, the SCAD penalized estimator can be obtained by using the SCAD penalty, i.e.,

$$P_\lambda(|x|) = \begin{cases} \lambda|x|, & \text{if } |x| \leq \lambda \\ \frac{-|x|^2 + 2a\lambda|x| - \lambda^2}{2(a-1)}, & \text{if } \lambda < |x| \leq a\lambda \\ \frac{\lambda^2(a+1)}{2}, & \text{if } |x| > a\lambda, \end{cases} \tag{4}$$

whose first order derivative is given by

$$P'_\lambda(|x|) = \lambda \left\{ I(|x| \leq \lambda) + \frac{(a\lambda - |x|)_+}{(a-1)\lambda} I(|x| > \lambda) \right\},$$

where $a > 2$ is also a penalty parameter for SCAD, with Fan and Li [32] recommending the setting $a = 3.7$. We use this value in our simulation studies. Note that $P'_\lambda(|x|)$ is nonnegative since SCAD is monotonically nondecreasing over $[0, \infty)$.

There are two challenges to be addressed. First, the log-likelihood function (2) involves complex integration. Second, the optimization problem in (3) is not convex.

The minimization of the penalized observed likelihood is not easy since its analytical form is not available. Pesonen et al. [27] maximized the likelihood based on a finite-memory algorithm for boundary-constrained optimization, but with high complexity. Lee and Scott [33] used the EM algorithm, but the E-step requires the second-order moments of the truncated normal distribution, which makes the computation slow.

2.2. Fitting L_1 Penalized Estimator by the EM Algorithm

In this paper, we view censored data as a special kind of incomplete data and apply the Expectation Maximization (EM) algorithm [28] to solve the optimization problem in (3). The EM algorithm is an iterative method to find maximum likelihood estimates of parameters in statistical models in the presence of incomplete data. It alternately iterates

the expectation step (E-step) and maximization step (M-step) until convergence. E-step computes the Q-function, i.e., the conditional expectation of the complete log-likelihood based on the observed data and the current parameters. M-step minimizes the Q-function to update the parameters. In this paper, we suggest two strategies, i.e., approximate computation and the Monte Carlo integration, to speed up calculating the Q function in the E-step, and the resulting EM algorithms are denoted as ApEM and MCEM.

For the censored multivariate normal distribution, the ith complete observation is $\boldsymbol{x}_i = (\boldsymbol{x}_{io_i}, \boldsymbol{x}_{ic_i})$. The log-likelihood function of complete data is written as

$$l_{com}(\theta \mid \boldsymbol{x}_o, \boldsymbol{x}_c) \propto -\log|\Sigma| - \frac{1}{n}\mathrm{tr}\Big(\Sigma^{-1}T_2\Big) + \frac{2}{n}\mu^\top\Sigma^{-1}T_1 - \mu^\top\Sigma^{-1}\mu, \tag{5}$$

where $T_1 = \sum_{i=1}^{n} \boldsymbol{x}_i, T_2 = \sum_{i=1}^{n} \boldsymbol{x}_i\boldsymbol{x}_i^\top$. Let $\theta^{(t)} = (\mu^{(t)}, \Sigma^{(t)})$ be the estimate of the parameter $\theta = (\mu, \Sigma)$ in the tth iteration of the EM algorithm, then the E-step and M-step are implemented as follows.

2.2.1. E-Step

In E-step, the Q-function is

$$\begin{aligned} Q(\theta, \theta^{(t)}) &= \mathrm{E}(-l_{com}(\theta|\boldsymbol{x}_o, \boldsymbol{x}_c)|\boldsymbol{x}_o, \boldsymbol{r}, \theta^{(t)}) + \lambda||\Sigma||_1 \\ &\propto \log|\Sigma| + \frac{1}{n}\mathrm{tr}\Big(\Sigma^{-1}T_2^{(t+1)}\Big) - \frac{2}{n}\mu^\top\Sigma^{-1}T_1^{(t+1)} + \mu^\top\Sigma^{-1}\mu + \lambda||\Sigma||_1 \end{aligned} \tag{6}$$

where

$$T_1^{(t+1)} = \mathrm{E}\Big(T_1 \mid \boldsymbol{x}_o, \boldsymbol{r}, \theta^{(t)}\Big) = \sum_{i=1}^{n} \mathrm{E}\left(\begin{pmatrix} \boldsymbol{x}_{io_i} \\ \boldsymbol{x}_{ic_i} \end{pmatrix} \mid \boldsymbol{x}_{io_i}, \boldsymbol{r}_i, \theta^{(t)}\right), \tag{7}$$

$$T_2^{(t+1)} = \mathrm{E}\Big(T_2 \mid \boldsymbol{x}_o, \boldsymbol{r}, \theta^{(t)}\Big) = \sum_{i=1}^{n} \mathrm{E}\left(\begin{pmatrix} \boldsymbol{x}_{io_i}\boldsymbol{x}_{io_i}^\top & \boldsymbol{x}_{io_i}\boldsymbol{x}_{ic_i}^\top \\ \boldsymbol{x}_{ic_i}\boldsymbol{x}_{io_i}^\top & \boldsymbol{x}_{ic_i}\boldsymbol{x}_{ic_i}^\top \end{pmatrix} \mid \boldsymbol{x}_{io_i}, \boldsymbol{r}_i, \theta^{(t)}\right). \tag{8}$$

To compute the Q function, it suffices to compute conditional expectation $\mathrm{E}(\boldsymbol{x}_{ic_i} \mid \boldsymbol{x}_{io_i}, \boldsymbol{r}_i, \theta^{(t)})$ and $\mathrm{E}(\boldsymbol{x}_{ic_i}\boldsymbol{x}_{ic_i}^\top \mid \boldsymbol{x}_{io_i}, \boldsymbol{r}_i, \theta^{(t)})$. Note that, given $(\boldsymbol{x}_{io_i}, \boldsymbol{r}_i)$, $\boldsymbol{x}_{ic_i}$ follows a truncated multivariate normal distribution $TMN(\mu_{c_i|o_i}^{(t)}, \Sigma_{c_i|o_i}^{(t)}, l_{c_i})$, where $\mu_{c_i|o_i}^{(t)} = \mu_{c_i}^{(t)} - \Sigma_{c_io_i}^{(t)}(\Sigma_{o_io_i}^{(t)})^{-1}(\boldsymbol{x}_{io_i} - \mu_{o_i}^{(t)})$, and $\Sigma_{c_i|o_i}^{(t)} = \Sigma_{c_ic_i}^{(t)} - \Sigma_{c_io_i}^{(t)}(\Sigma_{o_io_i}^{(t)})^{-1}\Sigma_{o_ic_i}^{(t)}$. Thus, calculating the Q-function requires computing the first and second moments of a multivariate truncated normal distribution. Lee [34] utilized the distribution function of the multivariate normal distribution to compute the first and second moments, and Lee and Scott [33] applied the eigenfunctions to find the first and second moments. However, all of these methods involve multivariate numerical integration, which results in a heavy computational burden when dealing with observations that have multiple censored variables.

In order to speed up the computation of the Q-function, we first consider an approximate computation via a univariate truncated normal distribution inspired by Guo et al. [35] and Augugliaro et al. [31]. Specifically, we approximate $E(x_{ij}|\boldsymbol{x}_{io_i}, \boldsymbol{r}_i, \theta^{(t)}) \approx E(x_{ij}|x_{io_i}, j)$, where the conditional expectation on the right side of the approximate equality is computed assuming that $x_{ij}|x_{io_i}, j$ follows a univariate truncated normal distribution $TN((\mu_{c_i|o_i}^{(t)})_j, (\Sigma_{c_i|o_i}^{(t)})_{jj}, l_j)$. Note that the above computation is approximate, because, for $j \in c_i$, the marginal conditional distribution of x_{ij} given $\boldsymbol{x}_{io_i}$ and $\boldsymbol{r}_i$ is generally not a univariate truncated normal distribution [36], although $\boldsymbol{x}_{ic_i}$ follows the truncated multivariate normal distribution $TMN(\mu_{c_i|o_i}^{(t)}, \Sigma_{c_i|o_i}^{(t)}, l_{c_i})$. One advantage of the approximate computa-

tion is that the moment of the univariate truncated normal distribution can be computed by exact formulas and requires only the evaluation of the cumulative distribution of the univariate Gaussian distribution [37].

Similarly, the second moment $E(\boldsymbol{x}_{ic_i}\boldsymbol{x}_{ic_i}^\top|\boldsymbol{x}_{io_i},\boldsymbol{r}_i)$ in $T_2^{(t+1)}$ can be computed approximately as follows. For $j,j' \in c_i$, when $j \neq j'$, we approximate $E(x_{ij}x_{ij'}|\boldsymbol{x}_{io_i},\boldsymbol{r}_i,\theta^{(t)}) \approx E(x_{ij}|\boldsymbol{x}_{io_i},j)E(x_{ij'}|\boldsymbol{x}_{io_i},j')$; when $j=j'$, $E(x_{ij}^2|\boldsymbol{x}_{io_i},\boldsymbol{r}_i,\theta^{(t)}) \approx E(x_{ij}^2|\boldsymbol{x}_{io_i},j)$. Note that the conditional expectation at the right end of the two approximate equal signs is computed assuming that $x_{ij}|\boldsymbol{x}_{io_i},j$ follows a univariate truncated normal distribution $TN((\mu_{c_i|o_i}^{(t)})_j,(\Sigma_{c_i|o_i}^{(t)})_{jj},l_j)$.

An alternative approach for accelerating the computation of Q-function is the Monte Carlo EM (MCEM) method of Wei and Tanner [38], in which a sequence of samples are generated from the conditional predictive distribution given the observed data and the current parameter to compute an approximation to the Q function. Specifically, for the ith observation $(\boldsymbol{x}_{o_i},\boldsymbol{r}_i)$, K random samples $\boldsymbol{x}_{c_i}^{(1)},\boldsymbol{x}_{c_i}^{(2)},...,\boldsymbol{x}_{c_i}^{(K)}$ are drawn from the conditional predictive distribution $p(\boldsymbol{x}_{c_i}|\boldsymbol{x}_{o_i},\boldsymbol{r}_i)$, i.e., the truncated multivariate normal distribution $TMN\left(\mu_{c_i|o_i}^{(t)},\Sigma_{c_i|o_i}^{(t)},l_{c_i}\right)$. Then, we obtain the complete data $\boldsymbol{x}_i^{(k)} = \left(\boldsymbol{x}_{o_i},\boldsymbol{x}_{c_i}^{(k)}\right)$, where $k=1,2,\ldots,K$, $i=1,2,\ldots,n$. Thus, the approximation can be obtained as $T_1^{(t+1)} \approx \frac{1}{nK}\sum_{i=1}^{n}\sum_{k=1}^{K}\boldsymbol{x}_i^{(k)}$ and $T_2^{(t+1)} \approx \frac{1}{nK}\sum_{i=1}^{n}\sum_{k=1}^{K}\boldsymbol{x}_i^{(k)}\boldsymbol{x}_i^{(k)\top}$.

2.2.2. M-Step

Next, we compute the updates $\theta^{(t+1)} = (\mu^{(t+1)},\Sigma^{(t+1)})$ as minimizer of $Q(\theta,\theta^{(t)})$ in (6). It is easily seen from (6) that $\mu^{(t+1)}$ and $\Sigma^{(t+1)}$ satisfy the following equations:

$$\mu^{(t+1)} = \frac{1}{n}T_1^{(t+1)}$$

and

$$\Sigma^{(t+1)} = \arg\min_{\Sigma\succ 0}\log|\Sigma| + tr(S^{(t+1)}\Sigma^{-1}) + \lambda||\Sigma||_1, \tag{9}$$

where $S^{(t+1)} = \frac{1}{n}T_2^{(t+1)} - \mu^{(t+1)}\mu^{(t+1)^\top}$. Therefore, the update in (9) reduces to a L_1 penalized log-likelihood problem for sparse covariance matrix estimation in the complete data setting, which can be solved by the coordinate descent algorithm of Wang [16].

2.3. Fitting SCAD Penalized Estimator by the EM Algorithm

Using the local linear approximation (LLA) proposed by Zou and Li [29] to the SCAD penalty, the sparse covariance matrix estimation problem can be reformulated as a sequence of penalized likelihood problems with a weighted L_1 penalty. This can be solved using an EM algorithm similar to that for the L_1 penalty, with only slight modifications to the coordinate descent algorithm for updating the matrix Σ in the M-step.

2.3.1. E-Step

By replacing the L_1 penalty with the SCAD penalty in Equation (6), the Q-function in the E-step is

$$\begin{aligned} Q(\theta,\theta^{(t)}) &= \mathrm{E}(-l_{com}(\theta|\boldsymbol{x_o},\boldsymbol{x_c})|\boldsymbol{x_o},\boldsymbol{r},\theta^{(t)}) + \sum_{i=1}^{p}\sum_{j=1}^{p}P_\lambda(|\Sigma_{ij}|) \\ &\propto \log|\Sigma| + \frac{1}{n}\mathrm{tr}(\Sigma^{-1}T_2^{(t+1)}) - \frac{2}{n}\mu^\top\Sigma^{-1}T_1^{(t+1)} + \mu^\top\Sigma^{-1}\mu + \sum_{i=1}^{p}\sum_{j=1}^{p}P_\lambda(|\Sigma_{ij}|), \end{aligned} \tag{10}$$

where $P_\lambda(\cdot)$ refers to the SCAD penalty (4). In each E-step, the LLA algorithm locally approximates the SCAD by a symmetric linear function. By the Taylor expansion, $P_\lambda(|\Sigma_{ij}|)$ is approximated in a neighborhood of $\Sigma_{ij}^{(t)}$ as follows,

$$P_\lambda(|\Sigma_{ij}|) \approx P_\lambda(|\Sigma_{ij}^{(t)}|) + P_\lambda'(|\Sigma_{ij}^{(t)}|)(|\Sigma_{ij}| - |\Sigma_{ij}^{(t)}|).$$

Thus, the penalty term can be regarded as a weighted version of the L_1 penalty up to a constant, i.e., $P_\lambda'(|\Sigma_{ij}^{(t)}|)|\Sigma_{ij}|$. The weighting scheme is determined by the first-order derivative of SCAD and the magnitude of the current estimate, with larger magnitudes corresponding to smaller weights.

Aligned with the E-step of the L_1 penalized likelihood method, the key to computing the Q-function lies in evaluating $T_1^{(t+1)}$ and $T_2^{(t+1)}$, which requires calculating the first and second moments of a multivariate truncated normal distribution. Two strategies including approximate computation and Monte Carlo integration are employed. For more details about these two strategies, refer to Section 2.2.1.

2.3.2. M-Step

The M-step minimizes $Q(\theta, \theta^{(t)})$ to obtain the updates $\theta^{(t+1)} = (\mu^{(t+1)}, \Sigma^{(t+1)})$, where $\mu^{(t+1)} = \frac{1}{n} T_1^{(t+1)}$ and

$$\Sigma^{(t+1)} = \arg\min_{\Sigma \succ 0} \log|\Sigma| + \mathrm{tr}(S^{(t+1)}\Sigma^{-1}) + \sum_{i=1}^{p}\sum_{j=1}^{p} P_\lambda'(|\Sigma_{ij}^{(t)}|)|\Sigma_{ij}|, \tag{11}$$

in which $S^{(t+1)} = \frac{1}{n} T_2^{(t+1)} - \mu^{(t+1)}\mu^{(t+1)^\top}$. If all $P_\lambda'(|\Sigma_{ij}^{(t)}|)$ are equal to λ, the optimization problem (11) will reduce to the L_1 penalized likelihood method. Due to the varying weights across elements, the coordinate descent algorithm proposed by Wang [16] cannot be directly applied to solve (11). Inspired by this algorithm, we update the matrix Σ one column and row at a time while holding the remaining elements fixed. The update for Σ closely follows the method in Wang [16]. For completeness, we provide a detailed derivation.

Without loss of generality, we take the last column and row as an example. For this purpose, we partition Σ and $S = S^{(t+1)}$ as

$$\Sigma = \begin{pmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\sigma}_{12} \\ \boldsymbol{\sigma}_{12}^\top & \sigma_{22} \end{pmatrix}, \qquad S = \begin{pmatrix} \boldsymbol{S}_{11} & \boldsymbol{s}_{12} \\ \boldsymbol{s}_{12}^\top & s_{22} \end{pmatrix},$$

where (i) $\boldsymbol{\Sigma}_{11}$ and $\boldsymbol{S}_{11}$ are the covariance matrix and the sample covariance matrix of the first $p-1$ variables, respectively; (ii) $\boldsymbol{\sigma}_{12}$ and $\boldsymbol{s}_{12}$ are the covariances and the sample covariances between the first $p-1$ variables and the last variable, respectively; and (iii) σ_{22} and s_{22} are the variance and the sample variance of the last variable, respectively. Let

$$\boldsymbol{\beta} = \boldsymbol{\sigma}_{12}, \qquad \gamma = \sigma_{22} - \boldsymbol{\sigma}_{12}^\top \boldsymbol{\Sigma}_{11}^{-1} \boldsymbol{\sigma}_{12}.$$

Then, the three terms in (11) can be expressed as a function of $(\boldsymbol{\beta}, \gamma)$,

$$\begin{aligned}
&\log|\Sigma| = \log(\gamma) + c_1, \\
&\mathrm{tr}(S\Sigma^{-1}) = \boldsymbol{\beta}^\top \boldsymbol{\Sigma}_{11}^{-1} \boldsymbol{S}_{11} \boldsymbol{\Sigma}_{11}^{-1} \boldsymbol{\beta}\gamma^{-1} - 2\boldsymbol{s}_{12}^\top \boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\beta}\gamma^{-1} + s_{22}\gamma^{-1} + c_2, \\
&\sum_{i=1}^{p}\sum_{j=1}^{p} P_\lambda'(|\Sigma_{ij}^{(t)}|)|\Sigma_{ij}| = 2\sum_{j=1}^{p-1} P_\lambda'(|\beta_j^{(t)}|)|\beta_j| + P_\lambda'(|\sigma_{22}^{(t)}|)(\gamma + \boldsymbol{\beta}^\top \boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\beta}) + c_3,
\end{aligned}$$

where c_1, c_2 and c_3 are constants. Thus, the optimization problem (11) can be turned into a new one with respect to $(\boldsymbol{\beta}, \gamma)$,

$$\min_{\boldsymbol{\beta},\gamma} \log(\gamma) + \boldsymbol{\beta}^\top \boldsymbol{\Sigma}_{11}^{-1} \boldsymbol{S}_{11} \boldsymbol{\Sigma}_{11}^{-1} \boldsymbol{\beta} \gamma^{-1} - 2\boldsymbol{s}_{12}^\top \boldsymbol{\Sigma}_{11}^{-1} \boldsymbol{\beta} \gamma^{-1} + s_{22}\gamma^{-1} + 2\sum_{j=1}^{p-1} P_\lambda'(|\beta_j^{(t)}|)|\beta_j| + P_\lambda'(|\sigma_{22}^{(t)}|)(\gamma + \boldsymbol{\beta}^\top \boldsymbol{\Sigma}_{11}^{-1} \boldsymbol{\beta}).$$

For γ, the minimization problem is

$$\min_{\gamma} \log(\gamma) + b\gamma^{-1} + P_\lambda'(|\sigma_{22}^{(t)}|)\gamma,$$

where $b = \boldsymbol{\beta}^\top \boldsymbol{\Sigma}_{11}^{-1} \boldsymbol{S}_{11} \boldsymbol{\Sigma}_{11}^{-1} \boldsymbol{\beta} - 2\boldsymbol{s}_{12}^\top \boldsymbol{\Sigma}_{11}^{-1} \boldsymbol{\beta} + s_{22}$. The solution of γ can be easily obtained as

$$\hat{\gamma} = \begin{cases} b, & \text{if } P_\lambda'(|\sigma_{22}^{(t)}|) = 0 \\ (-1 + \sqrt{1 + 4bP_\lambda'(|\sigma_{22}^{(t)}|)})/(2P_\lambda'(|\sigma_{22}^{(t)}|)), & \text{if } P_\lambda'(|\sigma_{22}^{(t)}|) \neq 0. \end{cases}$$

For $\boldsymbol{\beta}$, the minimization problem is

$$\min_{\boldsymbol{\beta}} \boldsymbol{\beta}^\top \boldsymbol{V} \boldsymbol{\beta} - 2\boldsymbol{u}^\top \boldsymbol{\beta} + 2\sum_{j=1}^{p-1} P_\lambda'(|\beta_j^{(t)}|)|\beta_j|,$$

where $\boldsymbol{V} = (v_{ij}) = \boldsymbol{\Sigma}_{11}^{-1} \boldsymbol{S}_{11} \boldsymbol{\Sigma}_{11}^{-1} \gamma^{-1} + P_\lambda'(|\beta_j^{(t)}|)\boldsymbol{\Sigma}_{11}^{-1}$ and $\boldsymbol{u} = (u_j) = \boldsymbol{\Sigma}_{11}^{-1} \boldsymbol{s}_{12} \gamma^{-1}$. This problem can be solved by another coordinate descent algorithm. For each $j \in \{1, \ldots, p-1\}$, the solution of β_j is

$$\hat{\beta}_j = \mathcal{S}(u_j - \sum_{k \neq j} v_{kj}\hat{\beta}_k, P_\lambda'(|\beta_j^{(t)}|))/v_{jj},$$

where $\mathcal{S}(x, t) = \text{sign}(x)(|x| - t)_+$ is the soft-threshold operator. The β_j for $j = 1, \ldots, p-1$ is updated iteratively until the inner coordinate descent algorithm converges. We then update the column as $\boldsymbol{\sigma}_{12} = \boldsymbol{\beta}, \sigma_{22} = \gamma + \boldsymbol{\beta}^\top \boldsymbol{\Sigma}_{11}^{-1} \boldsymbol{\beta}$ followed by cycling through all columns until convergence.

2.4. Selection of the Shrinkage Parameter

As different shrinkage parameters may lead to estimators with different sparsity levels, one will usually need to select an appropriate value of λ. Let $\hat{\theta}_\lambda = (\hat{\mu}_\lambda, \hat{\Sigma}_\lambda)$ denote the estimate of $\theta = (\mu, \Sigma)$ by using the shrinkage parameter λ. We would like to choose a value of λ to balance goodness-of-fit and model complexity by minimizing the Bayesian Information Criterion (BIC)

$$BIC(\hat{\theta}_\lambda) = -2\ell_{obs}(\hat{\theta}_\lambda | \boldsymbol{x}_o, \boldsymbol{r}) + \text{df}_\lambda \cdot \log(n),$$

over a grid of candidate values for λ, where df_λ denotes the number of nonzero off-diagonal estimates of $\hat{\Sigma}_\lambda$, and n is the sample size. However, the computational burden related to the evaluation of the observed log-likelihood function is heavy. For this reason, we choose the value of λ by minimizing the following approximate measure:

$$\begin{aligned} \overline{BIC}(\hat{\theta}_\lambda) &= -n\text{E}(\ell_{com}(\hat{\theta}_\lambda | \boldsymbol{x}_o, \boldsymbol{x}_c)) + \text{df}_\lambda \cdot \log(n) \\ &= n\log|\hat{\Sigma}_\lambda| + \text{tr}(\hat{\Sigma}_\lambda^{-1} T_2^{(t+1)}) - 2\hat{\mu}_\lambda^\top \hat{\Sigma}_\lambda^{-1} T_1^{(t+1)} + n\hat{\mu}_\lambda^\top \hat{\Sigma}_\lambda^{-1} \hat{\mu}_\lambda + \text{df}_\lambda \cdot \log(n), \end{aligned}$$

where $T_1^{(t+1)}$ and $T_2^{(t+1)}$ are computed as in (7) and (8), respectively.

3. Simulation

3.1. Simulation Settings

To verify the effectiveness of ApEM and MCEM with both L_1 and SCAD penalties, a simulation study is conducted using R with varying sample sizes n, left-censoring ratios c, and numbers of variables p. The data are generated from a p-dimensional censored multivariate normal distribution with a mean vector $\mu = (0, \ldots, 0)^T$ and two kinds of covariance matrix Σ described as follows:

- Cliques model: We take $\Sigma = diag(C_1, \ldots, C_5)$, where each C_i is a dense matrix of size $p/5$. The off-diagonal elements of C_i are ± 0.4 with random sign and the diagonal element $\Sigma_{i,i} = \delta$, where δ is chosen such that the condition number of Σ equals p.
- First-order moving average model: Diagonal element $\Sigma_{i,i} = 1$, and $\Sigma_{j,j-1} = \Sigma_{j-1,j} = 0.4$ for $j = 2, ..., p$. The remaining elements are equal to zero.

For each model, we explored all possible configurations with $p = 30, 100$, $n = 200, 500, 1000$, and $c = 0.1, 0.3$. Additionally, we examined high-dimensional settings for the cliques models with L_1 penalty, using parameter settings $(p, n, c) = (100, 50, 0.1), (100, 100, 0.1)$, and $(200, 100, 0.1)$.

Given each configuration (p, n, c), we first draw a Gaussian sample from multivariate Gaussian distribution $N(\mu, \Sigma)$ and then cut it by a censoring value vector l to obtain left-censored data, where l is determined by the mean, the square root of variance of each variable and the percentage of left censoring. We generated $N = 50$ datasets for comparison.

In the simulation studies, we compared ApEM and MCEM to the coordinate descent (CD) algorithm of Wang [16] under both L_1 and SCAD penalties. Since the CD algorithm is only suitable for completely observed data, we applied it to both the complete data without censoring and the dataset where the censored values were substituted with the censoring value vector. The resulting CD algorithms are referred to as com_CD and sub_CD, respectively. The penalty type is prefixed to the algorithm name to clearly distinguish the results produced by different penalty–algorithm pairings, such as L1_ApEM and SCAD_MCEM. All codes are available at https://github.com/Shanyi0106/cen_scov.

3.2. Simulation Results

We evaluate each method by L_1 norm $||\hat{\Sigma} - \Sigma||_1$, Frobenius norm $||\Sigma - \hat{\Sigma}||_F$, and KL divergence, calculated by $-\log|\Sigma\hat{\Sigma}^{-1}| + \mathrm{Tr}(\Sigma\hat{\Sigma}^{-1}) - p$ used in [18], where Σ is the true covariance matrix and $\hat{\Sigma}$ denotes the estimated covariance matrix obtained by each penalty–algorithm combination. Both the L_1 norm and the Frobenius norm measure the distance between the true Σ and the estimated $\hat{\Sigma}$, while the KL divergence evaluates the distance between the two distributions corresponding to Σ and $\hat{\Sigma}$, respectively. The results are presented in line charts in Figures 1–6, showing the average L_1 norm, average Frobenius norm and average KL divergence, all computed over 50 datasets. Note that MCEM was explored with different sample sizes $K = 20, 50, 100, 200$ in the E-step, and similar results were obtained across these values. Therefore, we only present the results obtained with $K = 20$.

For low-dimensional settings with $n > p$, Figures 1 and 2 display the average results for cliques models, and Figures 3 and 4 show the average results for moving average models. The three lines, marked with circles, triangles and plus signs, correspond to the settings $n = 200, 500, 1000$, respectively. The method com_CD provides an ideal situation where the data are completely observed and serves as a benchmark for comparison purposes.

From these figures, it is evident that com_CD performs the best under both L_1 and SCAD penalties as expected, except for the cliques model with $n = 500, p = 100, c = 0.1$ under the SCAD penalty. Unsurprisingly, sub_CD performs the worst in almost all settings, apart from the cliques model with $n = 200, 1000, p = 100, c = 0.1$ and $n = 200, p = 100, c = 0.3$ under the SCAD penalty. Compared to ApEM, MCEM yields slightly better results, with the exception of the cliques model with $p = 100$. In general, the SCAD penalty outperforms the L_1 penalty except in the case when $n = 200, p = 100$, and the performance of all methods improves as the sample size increases.

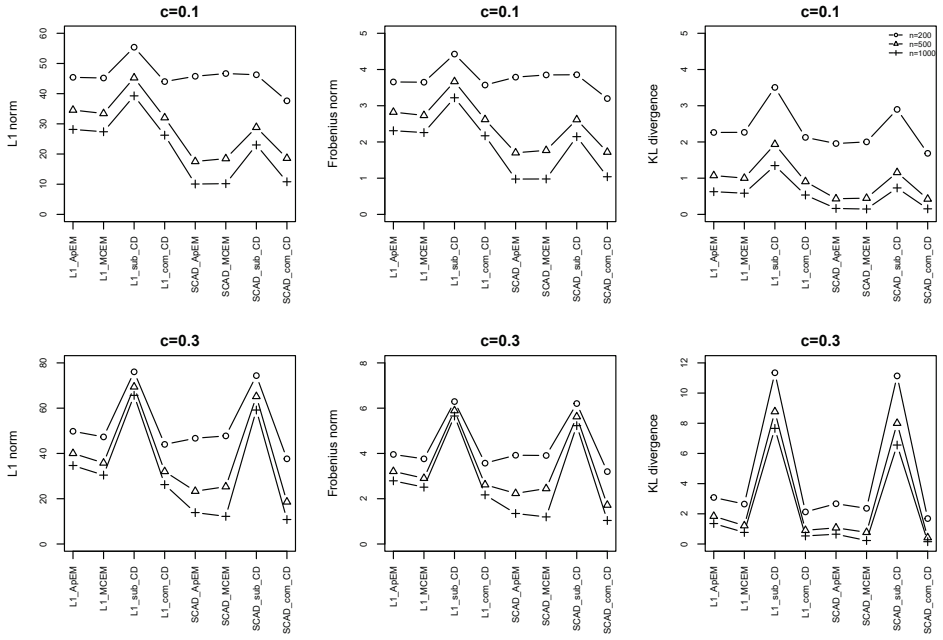

Figure 1. Line charts of average L_1 norm, Frobenius norm, KL divergence for cliques model with $p = 30$ under both L_1 and SCAD penalties.

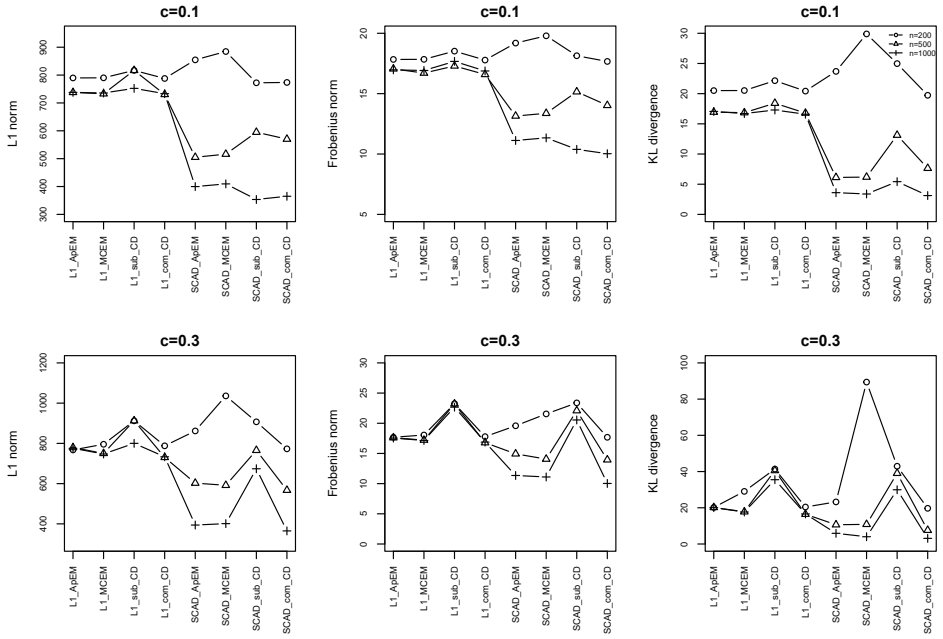

Figure 2. Line charts of average L_1 norm, Frobenius norm, KL divergence for cliques model with $p = 100$ under both L_1 and SCAD penalties.

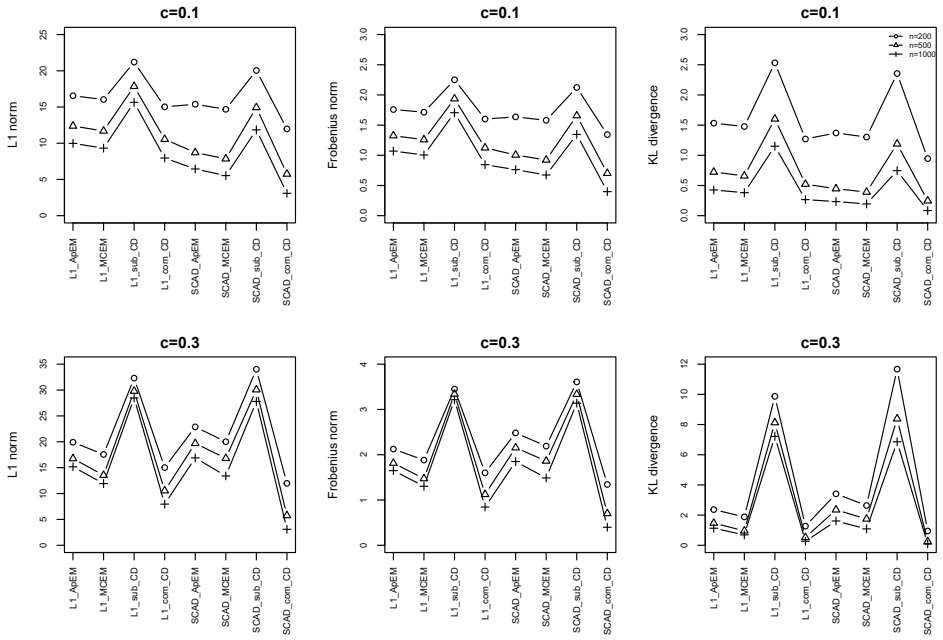

Figure 3. Line charts of average L_1 norm, Frobenius norm, KL divergence for the moving average model with $p = 30$ under both L_1 and SCAD penalties.

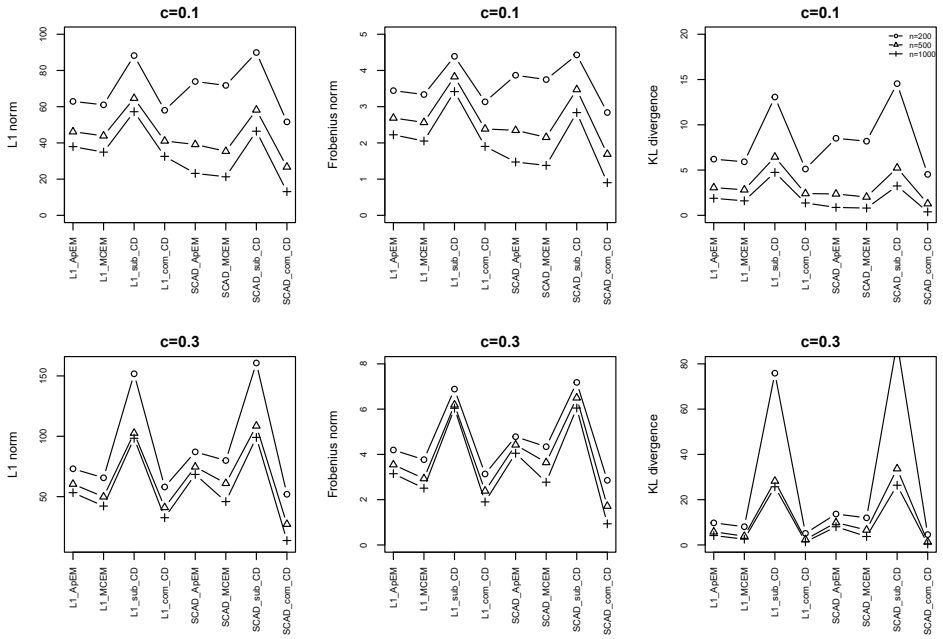

Figure 4. Line charts of average L_1 norm, Frobenius norm, KL divergence for the moving average model with $p = 100$ under both L_1 and SCAD penalties.

Figures 5 and 6 present the line charts of the average L_1 norm, the average Frobenius norm and average KL divergence, for high-dimensional settings with $n \leq p$ under the L_1 penalty. When $p = 100, n = 50$, MCEM runs slightly better than ApEM. For the settings with $p = 100, n = 100$ and $p = 200, n = 100$, MCEM and ApEM perform similarly.

Furthermore, the average CPU times in seconds and the average number of iterations for the cliques model and moving average model with the L_1 penalty under low-dimensional settings are presented in Tables A1 and A2, respectively, highlighting a significant advantage of MCEM over ApEM in terms of the computational efficiency. From Table A3, we observe that for the cliques model with the L_1 penalty under high-dimensional settings, all algorithms, particularly ApEM and MCEM, are computationally expensive.

The reason is that the number of EM iterations required for the given smallest shrinkage parameter λ is high, causing the EM algorithm to call the CD algorithm many times, which leads to a substantial computational burden.

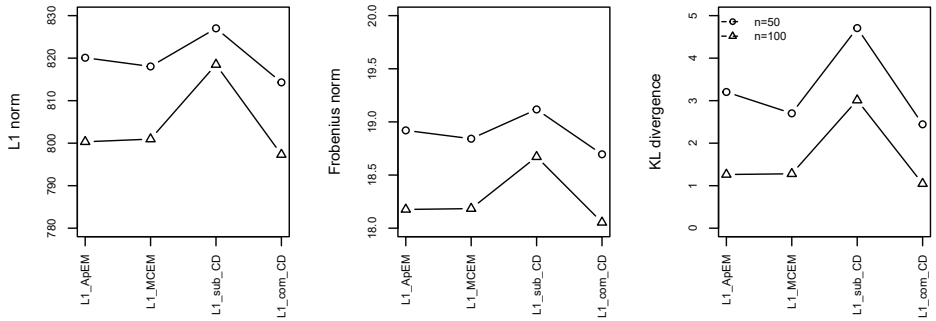

Figure 5. Line charts of average L_1 norm, Frobenius norm, KL divergence for the cliques model with $c = 0.1, p = 100, n = 50, 100$ under the L_1 penalty.

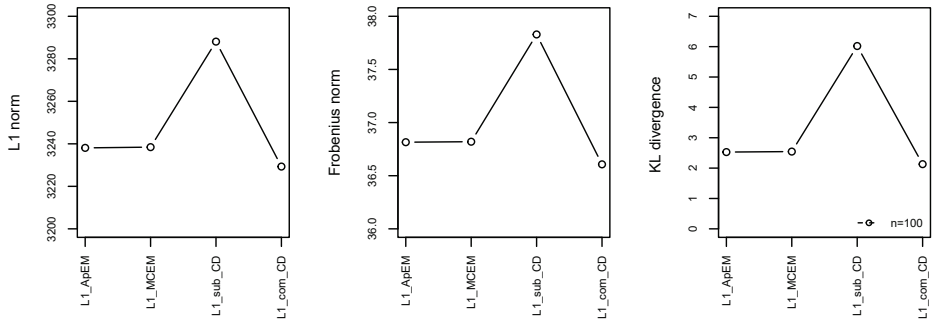

Figure 6. Line charts of average L_1 norm, Frobenius norm, KL divergence for cliques model with $c = 0.1, p = 200, n = 100$ under the L_1 penalty.

3.3. Cell Signalling Dataset Under Artificial Censoring

In this subsection, we assess the performance of L1_ApEM and L1_MCEM using real biological data that were used in [15] for sparse covariance matrix estimation. The data consist of flow cytometry measurements of the concentrations of $p = 11$ proteins in $n = 7466$ cells [39]. Despite the data being fully observed, we apply our methods to the datasets where observations are made artificially left-censored, similarly to [31]. Two datasets are generated with varying degrees of left censoring by designating the top 10% and 30% of the lowest values as missing.

To assess L1_ApEM and L1_MCEM, we compare them with several imputation methods, such as k-nearest neighbor imputation (denoted as L1_missknn), random forest imputation (denoted as L1_missforest), half of censored value imputation (denoted as L1_half), column mean imputation (denoted as L1_colmean) and lower limit of detection as censored value (denoted as L1_sub_CD). All these techniques start by imputing censored values and then utilize the CD algorithm to obtain the L_1 penalized estimation of the covariance matrix. The obtained estimated covariance matrix is denoted by $\hat{\Sigma}_{cen}$ by some imputation method, L1_ApEM or L1_MCEM. We also use the CD algorithm on the complete data without censoring and obtain $\hat{\Sigma}_{com}$. We compute the metric L_1 norm $= ||\hat{\Sigma}_{cen} - \hat{\Sigma}_{com}||_1$, Frobenius norm $= ||\hat{\Sigma}_{cen} - \hat{\Sigma}_{com}||_F$, and KL divergence $= -\log|\hat{\Sigma}_{com}\hat{\Sigma}_{cen}^{-1}| + \mathrm{Tr}(\hat{\Sigma}_{com}\hat{\Sigma}_{cen}^{-1}) - p$.

Table 1 shows average metrics for L1_ApEM, L1_MCEM, L1_sub_CD, L1_half, L1_colmean, L1_missknn and L1_missforest under varying degrees of censoring. It can be seen from Table 1 that L1_ApEM and L1_MCEM outperform all imputation methods. The performance of all methods decreases as degree of censoring increases.

Table 1. Average L_1 norm, Frobenius norm and KL divergence of L1_ApEM, L1_MCEM, L1_sub_CD, L1_half, L1_colmean, L1_missforest and L1_missknn under various degrees of censoring (0.1 and 0.3).

	c = 0.1			c = 0.3		
	L_1 Norm	Frobenius Norm	KL Divergence	L_1 Norm	Frobenius Norm	KL Divergence
L1_ApEM	3.03	0.59	0.64	6.28	1.22	1.97
L1_MCEM	2.44	0.49	0.62	4.19	0.84	2.70
L1_sub_CD	3.96	0.78	1.41	6.51	1.31	8.88
L1_half	3.53	1.19	1.60	9.92	3.10	4.20
L1_colmean	6.72	1.27	3.18	10.67	2.10	10.96
L1_missknn	5.86	1.12	3.45	8.96	1.77	11.43
L1_missforest	5.67	1.11	4.62	8.78	1.79	21.11

4. Application to Proteomic Datasets

In this section, we will apply L1_ApEM and L1_MCEM to analyze the proteomics dataset in the R package 'imputeLCMD' [40]. The data contain three biological replicates and three technical replicates each of Conditioned Medium (CM) and Whole Cell Lysate (WCL) for the C8-D1A cell line. The dataset was processed using MaxQuant for iBAQ protein intensities. It includes 18 iBAQ protein intensities for 7396 proteins, where each iBAQ protein intensity has a left-censored loss ratio of more than 10%. The data were log-transformed in our analysis.

We compare L1_ApEM and L1_MCEM with the imputation methods used in the previous section. The imputation methods start by estimating censored values and then utilize the CD algorithm to obtain the L_1 penalized estimator of the covariance matrix. Then, we calculate the correlation coefficient matrices from the estimated covariance matrices, and sample correlation matrix using sample data. We present heat maps of correlation matrices in Figure 7. From Figure 7, we can see that both L1_ApEM and L1_MCEM result in a sensible sparsity pattern, as they not only preserve the relationships between variables with strong correlations but also yield sparse estimates for variables with weak correlations. The heat map obtained by our method is clustered into two blocks, indicating a stronger correlation between the first nine variables, which are the iBAQ values of proteins secreted by the cell into its external environment, and the last nine variables, which are the iBAQ values of all proteins inside the cell.

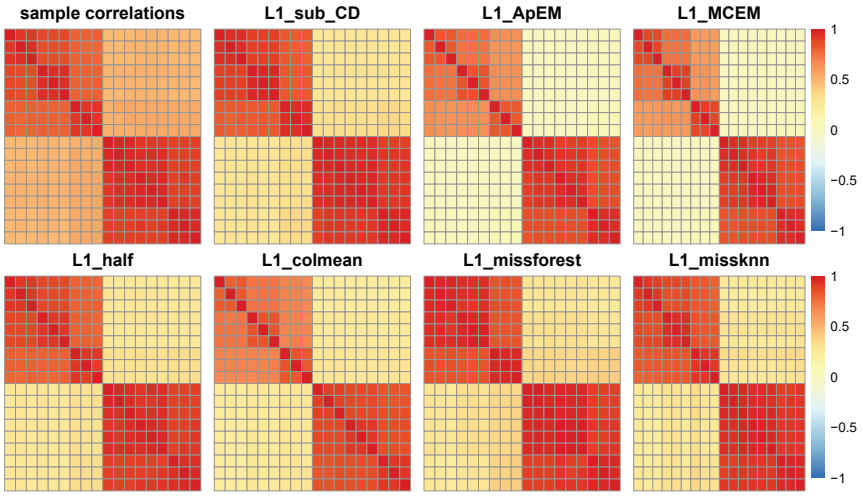

Figure 7. Heat maps of correlation coefficient matrices estimated by L1_ApEM, L1_MCEM, L1_missforest, L1_missknn, L1_half, L1_colmean, L1_sub_CD and sample correlations for proteomic dataset.

5. Conclusions

In this paper, we proposed ApEM and MCEM under the L_1 penalty and the SCAD penalty for estimating sparse covariance matrices in the multivariate normal model with left-censored data due to detection limits. Simulation studies reveal that ApEM and MCEM outperform imputation methods, regardless of whether the L_1 penalty or the SCAD penalty is used, and that the estimates obtained by SCAD penalization generally outperform those obtained by the L_1 penalty in most cases. Furthermore, we analyze cell signaling and proteomic datasets, demonstrating the advantages of ApEM and MCEM over imputation methods for estimating sparse covariance matrices.

As shown in Wu [41] and Chapter 3 of McLachlan and Krishnan [42], under mild conditions, every limit point of the sequences generated by EM converges to a local maximum rather than a global maximum. Thus, the global convergence of ApEM and MCEM cannot be guaranteed, as they are proposed based on the EM framework. Since different initial values may result in different estimated covariance matrices Σ, we compared the performance of two initialization strategies. One strategy uses the sample covariance matrix calculated from the dataset, where censored values are substituted by the censoring value vector, while the other uses a diagonal matrix with diagonal elements derived from the first strategy. The simulation results display that these two strategies yield similar average L_1 norm, average Frobenius norm and average divergence across all four methods under the $L1$ penalty, including ApEM, MCEM, sub_CD and com_CD. This suggests that the two initialization strategies have little impact on the performance. Due to time constraints, further simulations related to initialization will be conducted in future research.

The current study can also be extended in the following directions. First, the M-steps of the proposed ApEM and MCEM algorithms are computationally expensive for high-dimensional settings, especially the settings with small sample size, so we will try a more efficient algorithm, such as the proximal distance algorithm proposed by Xu and Lange [17], to reduce the computational burden. We also will consider the L_0 penalty using the extended Bayesian information criteria as in [18] to eliminate the need for hyper-parameter tuning to speed up computation. Second, we will try a non-convex penalty such as minimax concave penalty (MCP) to reduce the estimation bias of L_1 penalty, as discussed by Wei and Zhao [13]. Third, it is of interest to further study the consistency for the proposed penalized log-likelihood estimator under technical assumptions. Fourth, estimating a sparse covariance matrix without assuming a normal distribution is interesting, since the normal distribution assumption may not be satisfied for some censored datasets.

Author Contributions: Conceptualization, P.-F.X., S.L. and M.-L.T.; methodology, S.L., Q.-Z.Z., L.S. and P.-F.X.; software, S.L. and L.S.; writing—original draft preparation, P.-F.X. and S.L.; writing—review and editing, S.L., Q.-Z.Z., L.S., P.-F.X. and M.-L.T.; funding acquisition, P.-F.X. All authors have read and agreed to the published version of the manuscript.

Funding: The research of Ping-Feng Xu was supported by the National Social Science Fund of China (No. 23BTJ062).

Data Availability Statement: Data are available upon request to the authors.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A

Tables A1–A3 show the average CPU times (in seconds) for the E-step and M-step, the total average CPU time (in seconds), and the average number of iterations for both the L1_ApEM and L1_MCEM methods and the average CPU times (in seconds) for the L1_com_CD and L1_sub_CD algorithms. Overall, the CPU time increases with the number

of variables p, the sample size n, and the percentage of censoring c. L1_MCEM runs faster than L1_ApEM in E-step in all the settings.

Table A1. Average CPU time (in seconds) of E-step and M-step, average CPU time in total and average number of iterations for cliques model with L_1 penalty under low-dimensional settings.

			c = 0.1				c = 0.3			
			E-Step	**M-Step**	**Total**	**iter**	**E-Step**	**M-Step**	**Total**	**iter**
p = 30	n = 200	L1_ApEM	18.58	0.26	18.85	3.72	71.78	0.41	72.21	7.00
		L1_MCEM	6.16	0.25	6.45	3.58	20.36	0.43	20.84	7.15
		L1_sub_CD			0.35				0.33	
		L1_com_CD			0.33				0.33	
	n = 500	L1_ApEM	44.42	0.22	44.66	3.02	172.96	0.33	173.32	6.50
		L1_MCEM	14.31	0.22	14.70	3.20	46.28	0.39	46.84	5.95
		L1_sub_CD			0.34				0.33	
		L1_com_CD			0.33				0.32	
	n = 1000	L1_ApEM	88.54	0.24	88.81	3.00	343.98	0.33	344.35	6.35
		L1_MCEM	27.55	0.23	28.09	3.02	89.53	0.40	90.21	6.35
		L1_sub_CD			0.33				0.30	
		L1_com_CD			0.33				0.31	
p = 100	n = 200	L1_ApEM	43.34	17.56	61.01	4.00	187.14	31.11	218.46	11.00
		L1_MCEM	21.01	17.24	38.64	3.98	60.08	25.88	86.39	6.75
		L1_sub_CD			18.48				24.61	
		L1_com_CD			23.93				23.49	
	n = 500	L1_ApEM	107.28	9.26	116.69	4.00	440.50	16.14	456.86	7.00
		L1_MCEM	49.24	8.92	59.17	3.72	121.94	16.51	139.50	7.25
		L1_sub_CD			16.46				14.27	
		L1_com_CD			12.25				11.94	
	n = 1000	L1_ApEM	212.09	9.01	221.33	3.98	860.64	14.79	875.71	7.00
		L1_MCEM	86.89	7.57	95.87	3.14	228.27	15.79	245.57	6.75
		L1_sub_CD			12.71				13.15	
		L1_com_CD			11.94				11.92	

Table A2. Average CPU time (in seconds) of E-step and M-step, average CPU time in total and average number of iterations for moving average model with L_1 penalty under low-dimensional settings.

			c = 0.1				c = 0.3			
			E-Step	**M-Step**	**Total**	**iter**	**E-Step**	**M-Step**	**Total**	**iter**
p = 30	n = 200	L1_ApEM	14.59	0.18	14.79	3.08	62.09	0.31	62.42	6.25
		L1_MCEM	5.54	0.21	5.79	3.54	21.81	0.39	22.24	6.45
		L1_sub_CD			0.29				0.30	
		L1_com_CD			0.28				0.28	
	n = 500	L1_ApEM	36.09	0.17	36.27	3.00	149.25	0.31	149.57	6.00
		L1_MCEM	12.98	0.18	13.31	3.10	50.66	0.34	51.15	6.10
		L1_sub_CD			0.25				0.29	
		L1_com_CD			0.25				0.28	
	n = 1000	L1_ApEM	70.72	0.16	70.90	3.00	296.08	0.28	296.38	6.00
		L1_MCEM	24.62	0.17	25.08	3.10	99.67	0.34	100.30	6.00
		L1_sub_CD			0.26				0.26	
		L1_com_CD			0.25				0.25	
p = 100	n = 200	L1_ApEM	43.32	45.07	88.50	4.00	187.35	66.16	253.69	7.40
		L1_MCEM	20.03	49.82	70.22	4.18	106.78	92.13	199.34	10.40
		L1_sub_CD			64.71				85.31	
		L1_com_CD			59.13				58.58	
	n = 500	L1_ApEM	85.26	14.31	99.72	3.08	379.60	27.41	407.22	6.00
		L1_MCEM	43.73	16.41	61.13	3.30	184.79	30.94	216.82	6.25
		L1_sub_CD			23.80				31.77	
		L1_com_CD			23.56				23.28	
	n = 1000	L1_ApEM	160.32	13.98	174.51	3.00	766.09	25.91	792.26	6.00
		L1_MCEM	78.11	14.89	94.38	3.04	343.23	29.75	374.44	6.10
		L1_sub_CD			23.89				24.15	
		L1_com_CD			23.49				23.57	

Table A3. Average CPU time (in seconds) of E-step and M-step, average CPU time in total and average number of iterations for cliques model with L_1 penalty under high-dimensional settings.

			c = 0.1			
			E-Step	**M-Step**	**Total**	**iter**
p = 100	n = 50	L1_ApEM	92.20	230,702.07	230,795.24	5.70
		L1_MCEM	18.74	18,491.17	18,510.35	4.70
		L1_sub_CD			1916.47	
		L1_com_CD			832.07	
	n = 100	L1_ApEM	23.26	187.76	211.14	4.00
		L1_MCEM	12.04	348.34	360.73	4.10
		L1_sub_CD			465.46	
		L1_com_CD			369.12	
p = 200	n = 100	L1_ApEM	86.25	370,653.01	370,740.91	4.10
		L1_MCEM	28.25	22,842.59	22,872.19	4.10
		L1_sub_CD			2010.41	
		L1_com_CD			525.93	

References

1. Li, D. Estimation of large dynamic covariance matrices: A selective review. *Econom. Stat.* **2024**, *29*, 16–30. [CrossRef]
2. Johansson, K.; Ogut, M.G.; Pelger, M.; Schmelzer, T.; Boyd, S. A simple method for predicting covariance matrices of financial returns. *Found. Trends® Econom.* **2023**, *12*, 324–407. [CrossRef]
3. Kuismin, M.O.; Sillanpää, M.J. Estimation of covariance and precision matrix, network structure, and a view toward systems biology. *Wiley Interdiscip. Rev. Comput. Stat.* **2017**, *9*, e1415. [CrossRef]
4. Shen, H.W.; Cheng, X.Q.; Fang, B.X. Covariance, correlation matrix, and the multiscale community structure of networks. *Phys. Rev. E—Stat. Nonlinear Soft Matter Phys.* **2010**, *82*, 016114. [CrossRef]
5. Bickel, P.J.; Levina, E. Regularized estimation of large covariance matrices. *Ann. Stat.* **2008**, *36*, 199–227. [CrossRef]
6. Cai, T.T.; Zhang, C.H.; Zhou, H.H. Optimal rates of convergence for covariance matrix estimation. *Ann. Stat.* **2010**, *38*, 2118–2144. [CrossRef]
7. Ollila, E.; Breloy, A. Regularized tapered sample covariance matrix. *IEEE Trans. Signal Process.* **2022**, *70*, 2306–2320. [CrossRef]
8. Bickel, P.J.; Levina, E. Covariance regularization by thresholding. *Ann. Stat.* **2008**, *36*, 2577–2604. [CrossRef] [PubMed]
9. Rothman, A.J.; Levina, E.; Zhu, J. Generalized thresholding of large covariance matrices. *J. Am. Stat. Assoc.* **2009**, *104*, 177–186. [CrossRef]
10. Cai, T.; Liu, W. Adaptive thresholding for sparse covariance matrix estimation. *J. Am. Stat. Assoc.* **2011**, *106*, 672–684. [CrossRef]
11. Rothman, A.J. Positive definite estimators of large covariance matrices. *Biometrika* **2012**, *99*, 733–740. [CrossRef]
12. Xue, L.; Ma, S.; Zou, H. Positive-definite l_1-penalized estimation of large covariance matrices. *J. Am. Stat. Assoc.* **2012**, *107*, 1480–1491. [CrossRef]
13. Wei, Q.; Zhao, Z. Large covariance matrix estimation with oracle statistical rate via majorization-minimization. *IEEE Trans. Signal Process.* **2023**, *71*, 3328–3342. [CrossRef]
14. Wang, X.; Kong, L.; Wang, L. Estimation of sparse covariance matrix via non-convex regularization. *J. Multivar. Anal.* **2024**, *202*, 105294. [CrossRef]
15. Bien, J.; Tibshirani, R.J. Sparse estimation of a covariance matrix. *Biometrika* **2011**, *98*, 807–820. [CrossRef]
16. Wang, H. Coordinate descent algorithm for covariance graphical lasso. *Stat. Comput.* **2014**, *24*, 521–529. [CrossRef]
17. Xu, J.; Lange, K. A proximal distance algorithm for likelihood-based sparse covariance estimation. *Biometrika* **2022**, *109*, 1047–1066. [CrossRef]
18. Fatima, G.; Stoica, P.; Babu, P. L_0 penalized maximum likelihood estimation ofsparse covariance matrices. *IEEE Signal Process. Lett.* **2024**, *32*, 66–70. [CrossRef]
19. Fatima, G.; Babu, P.; Stoica, P. Two new algorithms for maximum likelihood estimation of sparse covariance matrices with applications to graphical modeling. *IEEE Trans. Signal Process.* **2024**, *72*, 958–971. [CrossRef]
20. Sung, B.; Lee, J. Covariance structure estimation with Laplace approximation. *J. Multivar. Anal.* **2023**, *198*, 105225. [CrossRef]
21. Lubin, J.H.; Colt, J.S.; Camann, D.; Davis, S.; Cerhan, J.R.; Severson, R.K.; Bernstein, L.; Hartge, P. Epidemiologic evaluation of measurement data in the presence of detection limits. *Environ. Health Perspect.* **2004**, *112*, 1691–1696. [CrossRef] [PubMed]
22. Wei, R.; Wang, J.; Jia, E.; Chen, T.; Ni, Y.; Jia, W. GSimp: A Gibbs sampler based left-censored missing value imputation approach for metabolomics studies. *PLoS Comput. Biol.* **2018**, *14*, e1005973. [CrossRef]
23. Shoari, N.; Dube, J.S. Estimating the mean and standard deviation of environmental data with below detection limit observations: Considering highly skewed data and model misspecification. *Chemosphere* **2015**, *138*, 599–608. [CrossRef] [PubMed]

24. Mohammed, R.A.M.; Brooks, S.C.; Tsai, C.H.; Ahmed, T.; Rucker, D.F.; Ulery, A.; Pierce, E.; Carroll, K.C. Geostatistical interpolation of streambed hydrologic attributes with addition of left censored data and anisotropy. *J. Hydrol.* **2021**, *599*, 126474. [CrossRef]
25. Hoffman, H.J.; Johnson, R.E. Pseudo-likelihood estimation of multivariate normal parameters in the presence of left-censored data. *J. Agric. Biol. Environ. Stat.* **2015**, *20*, 156–171. [CrossRef]
26. Jones, M.P.; Perry, S.S.; Thorne, P.S. Maximum pairwise pseudo-likelihood estimation of the covariance matrix from left-censored data. *J. Agric. Biol. Environ. Stat.* **2015**, *20*, 83–99. [CrossRef]
27. Pesonen, M.; Pesonen, H.; Nevalainen, J. Covariance matrix estimation for left-censored data. *Comput. Stat. Data Anal.* **2015**, *92*, 13–25. [CrossRef]
28. Dempster, A.P.; Laird, N.M.; Rubin, D.B. Maximum likelihood from incomplete data via the EM algorithm. *J. R. Stat. Soc. Ser. B (Methodol.)* **1977**, *39*, 1–22. [CrossRef]
29. Zou, H.; Li, R. One-step sparse estimates in nonconcave penalized likelihood models. *Ann. Stat.* **2008**, *36*, 1509. [PubMed]
30. Little, R.J.A.; Rubin, D.B. *Statistical Analysis with Missing Data*, 3rd ed.; John Wiley & Sons: Hoboken, NJ, USA, 2019.
31. Augugliaro, L.; Abbruzzo, A.; Vinciotti, V. ℓ_1-penalized censored Gaussian graphical model. *Biostatistics* **2020**, *21*, e1–e16.
32. Fan, J.; Li, R. Variable selection via nonconcave penalized likelihood and its oracle properties. *J. Am. Stat. Assoc.* **2001**, *96*, 1348–1360. [CrossRef]
33. Lee, G.; Scott, C. EM algorithms for multivariate Gaussian mixture models with truncated and censored data. *Comput. Stat. Data Anal.* **2012**, *56*, 2816–2829. [CrossRef]
34. Lee, L.F. On the first and second moments of the truncated multi-normal distribution and a simple estimator. *Econ. Lett.* **1979**, *3*, 165–169. [CrossRef]
35. Guo, J.; Levina, E.; Michailidis, G.; Zhu, J. Graphical models for ordinal data. *J. Comput. Graph. Stat.* **2015**, *1*, 183–204. [CrossRef]
36. Horrace, W.C. Some results on the multivariate truncated normal distribution. *J. Multivar. Anal.* **2005**, *94*, 209–221. [CrossRef]
37. Johnson, N.L.; Kotz, S.; Balakrishnan, N. *Continuous Univariate Distributions*, 2nd ed.; John Wiley & Sons: Hoboken, NJ, USA, 1994.
38. Wei, G.C.; Tanner, M.A. A Monte Carlo implementation of the EM algorithm and the poor man's data augmentation algorithms. *J. Am. Stat. Assoc.* **1990**, *85*, 699–704. [CrossRef]
39. Sachs, K.; Perez, O.; Pe'er, D.; Lauffenburger, D.A.; Nolan, G.P. Causal protein-signaling networks derived from multiparameter single-cell data. *Science* **2005**, *308*, 523–529. [CrossRef] [PubMed]
40. Lazar, C.; Burger, T. imputeLCMD: A Collection of Methods for Left-Censored Missing Data Imputation. R Package Version 2.1. 2022. Available online: https://cran.r-project.org/web/packages/imputeLCMD/imputeLCMD.pdf (accessed on 10 June 2022).
41. Wu, C.F.J. On the convergence properties of the EM algorithm. *Ann. Stat.* **1983**, *11*, 95–103. [CrossRef]
42. McLachlan, G.J.; Krishnan, T. *The EM Algorithm and Extensions*; John Wiley & Sons: Hoboken, NJ, USA, 2007.