A Multiobjective Evolutionary Approach for Solving Large-Scale Network Reconstruction Problems via Logistic Principal Component Analysis

Chaolong Ying, Jing Liu, *Senior Member, IEEE,* Kai Wu, and Chao Wang

***Abstract*—Currently, the problem of uncovering complex network structure and dynamics from time series is prominent in many fields. Despite the recent progress in this area, reconstructing large-scale networks from limited data remains a tough problem. Existing works treat connections of nodes as continuous values, leaving a challenge of setting a proper cut-off value to distinguish whether the connections exist or not. Besides, their performances on large-scale networks are far from satisfactory. Considering the reconstruction error and sparsity as two objectives, this paper proposes a subspace learning based evolutionary multiobjective network reconstruction algorithm, termed as SLEMO-NR, to solve the aforementioned problems. In the evolutionary process, we assume that binary-coded individuals obey the Bernoulli distribution and can use the probability and natural parameter as the alternative representations. Moreover, our approach utilizes the logistic principal component analysis (LPCA) to learn a subspace containing the features of network structure. The offspring solutions are generated in the learned subspace and then can be mapped back to the original space via LPCA. Benefitting from the alternative representations, a preference-based local search operator is proposed to concentrate on finding solutions approximate to the true sparsity. The experimental results on synthetic networks and six real-world networks demonstrate that, due to the well-learned network structure subspace and the preference-based strategy, our approach is effective in reconstructing large-scale networks compared to six existing methods.**

***Index Terms*—Complex network, evolutionary algorithm, logistic principal component analysis, network reconstruction.**

I. Introduction

THE study of complex networks is common in many fields. There exist unifying principles underlying the topology of networks, which are helpful to control collective dynamics [1]. However, directly accessing network structures and nodal dynamics is impossible in many cases, whereas only limited observable data are available [2]. It is thus imperative to propose network reconstruction approaches to uncover network structures from data extracted from observations or experiments. Potential applications exist in the fields of gene regulatory networks [3]–[5], catastrophes prediction [6], and brain function mapping [7].

The network reconstruction problem, the inverse problem, has become a heating topic in contemporary network science and engineering [8], [9]. Some works focus on reconstructing multilayer networks [10], [11], while in this paper we only consider the networks with a single layer. Since the structural information of the network is hidden under measurement data and the decision space is of extremely high dimension, accurate reconstruction of complex networks from limited data is challenging, especially for large-scale networks with hundreds and thousands of nodes. Usually, the network reconstruction problem (NRP) [12] is converted into sparse signal reconstruction problems. To recover the original signal, one needs to consider the following sparse optimization problem:

$$\min_{\boldsymbol{X}} \|\boldsymbol{X}\|_0 \quad \text{s.t.} \quad \boldsymbol{Y} = \boldsymbol{A}\boldsymbol{X} \tag{1}$$

where $\boldsymbol{X}$ in $\mathbb{R}^N$ is the neighboring vector, $\boldsymbol{Y}$ is an observation vector of length M in $\mathbb{R}^M$, and $\boldsymbol{A}$ is a sensing matrix in $\mathbb{R}^{M\times N}$. In most cases, the observation data are limited, which means $M \ll N$. Generally, $\boldsymbol{X}$ needs to be solved from known $\boldsymbol{Y}$ and $\boldsymbol{A}$. In NRPs, $x_{ij} \in \{0, 1\}$ is the element in $\boldsymbol{X}$, where 0 and 1 represent connection and disconnection of nodes i and j, respectively. The objective function $\|\boldsymbol{X}\|_0$ is the L_0-norm of $\boldsymbol{X}$, which is defined as the number of nonzero elements in $\boldsymbol{X}$. Problem (1) is usually converted into an unconstrained optimization problem with the following form:

$$\min_{\boldsymbol{X}} \left(\|\boldsymbol{Y} - \boldsymbol{A}\boldsymbol{X}\|_2^2 + \lambda\|\boldsymbol{X}\|_0\right) \tag{2}$$

where the L_2-norm $\|\boldsymbol{Y}-\boldsymbol{A}\boldsymbol{X}\|_2^2$ is the loss term, $\|\boldsymbol{X}\|_0$ is the regularization term, and λ is a regularization parameter with a positive value. Solving the equation above has been proven to be an NP-hard problem [13].

So far, a lot of approaches have been proposed to solve small-scale NRPs, where the regularization term in (2) is replaced by $\|\boldsymbol{X}\|_1$ in most cases [14], [15]. Napoletani *et al.* [16] proposed a constrained optimization technique based on chaotic time-series analysis to reconstruct nonlinear dynamical networks, which appeared to be limited to small-scale networks with sparse connections. Han *et al.* [17] pointed out

This work was supported in part by the Key Project of Science and Technology Innovation 2030 supported by the Ministry of Science and Technology of China under Grant 2018AAA0101302 and in part by the General Program of National Natural Science Foundation of China (NSFC) under Grant 61773300.

Chaolong Ying and Jing Liu are with the Guangzhou Institute of Technology, Xidian University, Guangzhou, China (e-mail: yingchaolong@126.com; neouma@163.com). For additional information regarding this paper, please contact Jing Liu, homepage: https://faculty.xidian.edu.cn/LJ22/zh_CN/index.htm.

Kai Wu and Chao Wang are with the School of Artificial Intelligence, Xidian University, Xi'an, China (e-mail: kwu@xidian.edu.cn; xiaofengxd@126.com).

that the vectors to be reconstructed were typically sparse due to the natural sparsity of realistic nonlinear dynamical networks and thus could be solved by a convex optimization algorithm named least absolute shrinkage and selection operator (LASSO) [18]. Wang *et al.* [19] proposed an efficient approach to reconstructing complex networks from evolutionary-game data via compressive sensing (CS) [20]. Compared to those associated with conventional reconstruction problems, the observation data requirements were relaxed considerably. Shen *et al.* [21] developed a CS-based framework to reconstruct complex networks with stochastic spreading dynamics from small amounts of polarized data, and this framework could be used to identify hidden sources. Unlike the above-mentioned approaches based on machine learning (ML), evolutionary algorithms (EAs) [22] are also introduced to solve NRPs. Wu *et al.* [23] decomposed NRP into a multiobjective optimization problem (MOP) and adopted the EA with LASSO initialization to solve it. To improve the accuracy of the reconstruction, Wu *et al.* [24] further utilized a memetic algorithm (MA) [25] to solve the non-convex problem (2) directly.

Despite the success in modeling and reconstructing complex networks, there are still two major limitations in previous approaches. First, for the unweighted networks, it is a natural idea to use binary-coded variables to represent the existence and inexistence of edges. However, no matter the approaches are ML-based or EA-based, they treat neighboring vector $\boldsymbol{X}$ as a real-coded variable. Utilizing the real-coded variable needs to face an inevitable problem of setting a cut-off value to distinguish the existent links from null connections. The elements with values smaller than the cut-off value correspond to zeros in the adjacency matrix, whereas those with values larger than the cut-off value are treated as corresponding to actual links. However, the reconstructed values are dispersed, leading to ambiguities in the identification of links. For example, if a solution $\boldsymbol{X}_i$ = (0.69, 0.27, 0.95, 0.03, 0.52, 0.38) is finally obtained and the true structure is $\boldsymbol{X}_{i,true}$ = (1, 0, 1, 0, 0, 0), then the cut-off value being 0.3, 0.5 or 0.7 will make a great difference. It is thus challenging to set a proper cut-off value to maximize the reconstruction performance. Moreover, due to the large search space, most previous approaches can only cope with small-scale NRPs with dozens of nodes. In addition to nodal dynamics information, the computational cost tends to increase dramatically with the size of networks [26]–[28]. For large-scale NRPs with thousands of nodes and edges, the reconstruction accuracy is far from satisfactory.

To overcome these shortcomings, we develop an approach to reconstructing large-scale networks from time series by a subspace learning based evolutionary multiobjective network reconstruction algorithm, termed as SLEMO-NR. So far, the multiobjective evolutionary algorithm (MOEA) has been successfully used to solve challenging optimization problems in ML [29]–[31] where the regularization term is converted into another objective function. In this way, MOEA avoids the choice of the regularization parameter in (2). However, existing MOEAs still can not handle large-scale NRPs properly. Specifically, the main contributions of our proposed method are summarized as follows,

1) Different from existing methods, SLEMO-NR uses binary-coded individuals to prevent the choice of a proper cut-off value, thus improving the ultimate performance. Thereafter, we propose to use the natural parameters and probability distribution as two alternative representations of binary-coded individuals. In the evolutionary process, the algorithm turns to search for the probability of the existence of an edge instead of searching for the existence directly. The natural parameters carrying the probability distribution information are engaged in generating new solutions instead of original binary-coded individuals. In the end, the newly generated natural parameters are transformed into binary offspring individuals.
2) For large-scale NRPs, the search space grows exponentially with the number of decision variables, causing stiff challenges for EAs to precisely and efficiently approximate the network structure [32]. Logistic principal component analysis (LPCA) [33], as an effective approach to extracting features from Bernoulli distributed data, is used to learn a subspace containing the features of network structure. In this way, SLEMO-NR is capable of searching for optimal solutions in the subspace, thus alleviating the impact of "curse of dimensionality".
3) In MOEAs, we can get a set of non-dominated solutions, called Pareto solutions. However, for NRPs, our ultimate goal is to find a so-called k-sparse solution, where k is the true sparsity in real networks. Benefitting from the alternative representations, a preference-based local search operator (PLSO) is proposed in SLEMO-NR to find a k-sparse solution, and no prior estimation of sparsity is required.

To validate the performance of SLEMO-NR, the evolutionary game (EG) [34] and resistor network (RN) [35] are employed as network models. At first, 16 problems based on four kinds of benchmark networks with different configurations are designed to evaluate the effectiveness of subspace learning and PLSO. After that, four representative problems are selected to analyze the sensitivity of parameters. In the end, extensive experiments on six real-world networks demonstrate that SLEMO-NR is superior to 6 state-of-the-art approaches in large-scale NRPs.

The rest of this paper is organized as follows. Section II gives basic formulations of the network reconstruction model and introduces two network models. Our proposed SLEMO-NR is described in detail in Section III. Section IV presents the experiments and comparisons with 6 existing methods. Finally, conclusions are given in Section V.

II. Preliminaries

A. Network Reconstruction Formulation

In a network, the interactions among N nodes are characterized as an $N \times N$ adjacent matrix. Particularly, reconstructing the whole network can be decomposed into learning local connections of nodes individually [17], [19], [24]. Therefore, it is a natural idea to decompose the entire problem into N independent subproblems. Besides, a node will not connect to itself, so we exclude the self-loops of the network. In a time period of length M, the states of nodes are recorded

as $\boldsymbol{A}_{net} = [\boldsymbol{A}_1, \boldsymbol{A}_2, \ldots, \boldsymbol{A}_N]$ and the outcomes of node interactions are recorded as $\boldsymbol{Y}_{net} = [\boldsymbol{Y}_1, \boldsymbol{Y}_2, \ldots, \boldsymbol{Y}_N]$, where the ith slice $\boldsymbol{A}_i \in \mathbb{R}^{M\times(N-1)}$ and $\boldsymbol{Y}_i \in \mathbb{R}^M$ are the time series of node i. For the ith subproblem, we have:

$$\boldsymbol{Y}_i = \boldsymbol{A}_i \boldsymbol{X}_i \tag{3}$$

where $\boldsymbol{X}_i \in \mathbb{R}^{(N-1)}$ is the connections of node i with other nodes and our goal is to reconstruct $\boldsymbol{X}_i$ from the recorded time series $\boldsymbol{A}_i$ and $\boldsymbol{Y}_i$. Considering the measurement error and the sparsity of variables as two objectives, the above subproblem can be transformed into an MOP:

$$\min_{\boldsymbol{X}_i} \left\{ f_1\left(\boldsymbol{X}_i\right) = \frac{1}{M}\left\|\boldsymbol{A}_i\boldsymbol{X}_i - \boldsymbol{Y}_i\right\|_2^2, f_2\left(\boldsymbol{X}_i\right) = \left\|\boldsymbol{X}_i\right\|_0 \right\} \tag{4}$$

B. Network Models

1) Evolutionary Game Model: Evolutionary game dynamics is a kind of game theory applying population dynamical methods, which is commonly used to model node-to-node interactions in various complex systems [36]. In each round of an EG, players have one of the two strategies ($\boldsymbol{S}$) to choose: cooperation (Co) or defection (De), expressed as $\boldsymbol{S}(Co) = (1,0)^T$ and $\boldsymbol{S}(De) = (0,1)^T$. The payoffs of two players are determined by the strategies they choose and the payoff matrix of the specific game they are playing. The prisoner's dilemma game (PDG) [37], a representative game model, is adopted in this paper. Its payoff matrix is:

$$\boldsymbol{P}_{PDG} = \begin{pmatrix} 1 & 0 \\ b & 0 \end{pmatrix} \tag{5}$$

where $b \in (1,2)$ is a parameter representing the temptation to defect. On the condition that both two players choose to cooperate (or defect), their payoffs are 1 (or 0). Whereas in the remaining cases when two players have different strategies, the defector gains payoff b but the cooperator gains the "sucker" payoff 0. In this paper, b is set to 1.2 [38].

For player i, its neighboring vector $\boldsymbol{X}_i$ stands for the connections with other players, with elements $x_{ij} = 1$ if players i and j are connected and $x_{ij} = 0$ otherwise. At each round of the game, all players play the game with their neighbors. For player i at time step t, its payoff $\boldsymbol{Y}_i(t)$ is denoted as:

$$\boldsymbol{Y}_i(t) = \sum_{j=1}^{N} \boldsymbol{S}_i^T(t)\boldsymbol{P}_{PDG}\boldsymbol{S}_j(t)x_{ij} \tag{6}$$

where $\boldsymbol{S}_i(t)$ is the strategy of player i. After each round of the game, a player will adjust its strategy according to both its own and its neighbors' payoffs, with the purpose of maximizing its payoff at the next round. In our simulations of the EG model, the Fermi rule is adopted to update the strategy, which is defined as follows:

$$W\left(\boldsymbol{S}_i \leftarrow \boldsymbol{S}_j\right) = \frac{1}{1+\exp\left[\left(\boldsymbol{Y}_i - \boldsymbol{Y}_j\right)/\kappa\right]} \tag{7}$$

where κ is a parameter characterizing the stochastic uncertainties in EG dynamics.

According to (6), the NRP can be formulated as:

$$\boldsymbol{Y}_i = \left(\boldsymbol{Y}_i(1), \boldsymbol{Y}_i(2), \ldots, \boldsymbol{Y}_i(M)\right)^T \tag{8}$$

$$\boldsymbol{X}_i = \left(x_{i1}, \ldots, x_{i,i-1}, x_{i,i+1}, \ldots, x_{iN}\right)^T \tag{9}$$

$$\boldsymbol{A}_i = \begin{pmatrix} J_{i1}(1) & \cdots & J_{i,i-1}(1) & J_{i,i+1}(1) & \cdots & J_{iN}(1) \\ J_{i1}(2) & \cdots & J_{i,i-1}(2) & J_{i,i+1}(2) & \cdots & J_{iN}(2) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ J_{i1}(M) & \cdots & J_{i,i-1}(M) & J_{i,i+1}(M) & \cdots & J_{iN}(M) \end{pmatrix} \tag{10}$$

where $J_{ij}(t) = \boldsymbol{S}_i^T(t)\boldsymbol{P}_{PDG}\boldsymbol{S}_j(t)$. It should be noted that $\boldsymbol{Y}_i$ is derived from the payoff data, $\boldsymbol{A}_i$ is obtained through the calculation of strategy data, and $\boldsymbol{X}_i$ containing all possible connections need to be reconstructed. The matrix $\boldsymbol{A}_i$ and vectors $\boldsymbol{Y}_i$, $\boldsymbol{X}_i$ satisfy (3).

The numerical simulation process of EG is described as follows. At the beginning of the game, all players choose to cooperate or defend. At each round of the game, the payoff of each player is calculated according to (6). All the payoffs and strategies in the process are recorded as time series to reconstruct the network.

2) Resistor network model: Resistor network dynamics is a kind of circuit system considering current transportation in resistors [35]. For two nodes i and j, the resistance of a resistor between them is denoted as R_{ij}. If i and j are directly connected by a resistor, $R_{ij} = 1$; otherwise, $R_{ij} = \infty$. According to the Kirchhoff's law, we have:

$$\sum_{j=1}^{N} \frac{1}{R_{ij}}\left(U_i - U_j\right) = C_i \tag{11}$$

where U_i and U_j are the voltages at nodes i and j respectively, and C_i is the total current flowing through node i. To better approximate real situations, alternating current is taken into consideration. To this end, the voltage is expressed as:

$$U_i = \bar{U}\sin\left[\left(\omega + \Delta\omega_i\right)t\right] \tag{12}$$

where $\bar{U} = 1$ is the peak of voltage, $\omega = 10^3$ is the frequency, and $\Delta\omega_i \in [0, 20]$ is the perturbation. The RN model can be written as the form of (3), with

$$\boldsymbol{Y}_i = \left(C_i(1), C_i(2), \ldots, C_i(M)\right)^T \tag{13}$$

$$\boldsymbol{X}_i = \left(\frac{1}{R_{i1}}, \ldots, \frac{1}{R_{i,i-1}}, \frac{1}{R_{i,i+1}}, \ldots, \frac{1}{R_{iN}}\right)^T \tag{14}$$

$$\boldsymbol{A}_i = \begin{pmatrix} V_{i1}(1) & \cdots & V_{i,i-1}(1) & V_{i,i+1}(1) & \cdots & V_{iN}(1) \\ V_{i1}(2) & \cdots & V_{i,i-1}(2) & V_{i,i+1}(2) & \cdots & V_{iN}(2) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ V_{i1}(M) & \cdots & V_{i,i-1}(M) & V_{i,i+1}(M) & \cdots & V_{iN}(M) \end{pmatrix} \tag{15}$$

where $V_{ij} = U_i - U_j$, and if i and j are disconnected, $1/R_{ij} = 0$; otherwise, $1/R_{ij} = 1$. It can be assumed that only the voltages and currents can be obtained and the resistor network is to be reconstructed.

The numerical simulation process of RN can be described as follows. Each node has an initial random number $\Delta\omega_i \in$

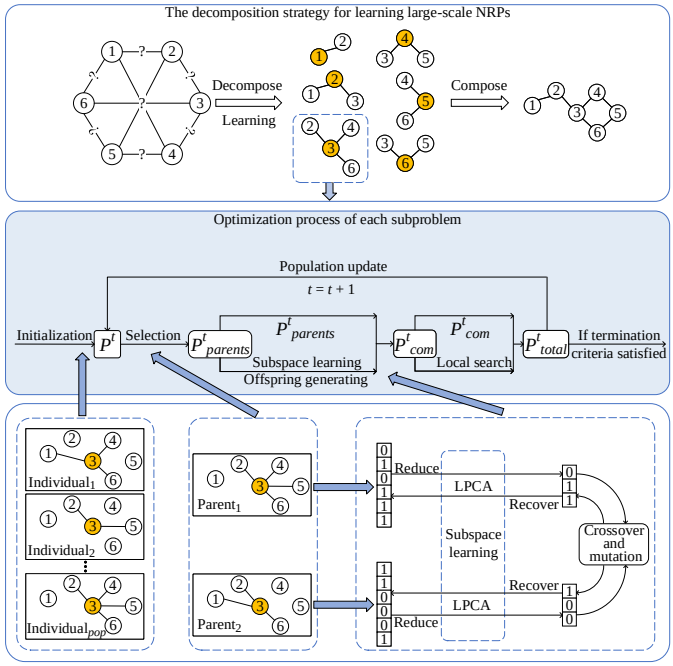


Fig. 1. The framework of SLEMO-NR.

$[0, 20]$ in the beginning. At time t, the voltages can be calculated using (12) and the currents are obtained via (11). They are both recorded as time series in a period.

III. SLEMO-NR

A. Framework of SLEMO-NR

SLEMO-NR is an EA with network structure subspace learning and PLSO to solve large-scale NRPs, which is based on the framework of NSGA-II [39]. The whole NRP is first decomposed into N subproblems, and then they are optimized separately using SLEMO-NR. For each subproblem, an initial population P^0 with size *pop* is first generated. In the following main loop, the binary tournament selection is utilized to select *pop* individuals according to the non-dominated front number and crowding distance of each individual. Unlike NSGA-II, crossover and mutation operators are performed to generate offspring in a subspace using the selected individuals. Combining the offspring with P^t, population P^t_{com} is obtained to undergo preference-based local search focusing on k-sparse solutions, and the generated solutions are also added to P^t_{com}. Finally, after deleting duplicated solutions in P^t_{com}, the elitist strategy makes *pop* individuals survive to the next generation. In short, SLEMO-NR is capable of searching for solutions in the subspace by LPCA and focusing on the k-sparse solution with the PLSO compared to NSGA-II. The general framework of SLEMO-NR is presented in Fig. 1 and Algorithm 1. In the following subsections, the main steps in SLEMO-NR are presented in detail.

B. Alternative Representations of Binary-Coded Individuals

LPCA is an extension of principal component analysis (PCA) [40] that captures the features underlying the Bernoulli distributed data. A projection of the data with minimum reconstruction error, which is the purpose of the ordinary PCA, can be viewed alternatively as a projection of the natural parameters of a saturated model with minimum deviance. In our algorithm, the decision variable $\boldsymbol{X}_i$ is binary-coded representing neighboring connections of a node. Assume that x_{ij} obeys the Bernoulli distribution [33] with the probability of p_{ij} and its corresponding natural parameter θ_{ij} is

$$\theta_{ij} = \text{logit}\,(p_{ij}) = \ln \frac{p_{ij}}{1 - p_{ij}} \tag{16}$$

We have the reverse formulation:

$$p_{ij} = \text{sigmoid}\,(\theta_{ij}) = \frac{1}{1 + \exp\,(-\theta_{ij})} \tag{17}$$

The natural parameters of the saturated model are denoted as $\tilde{\theta}_{ij}$. The saturated model, which is the best possible fit to the data, occurs on the condition that $p_{ij} = x_{ij}$ [33], which also means:

$$\tilde{\theta}_{ij} = \begin{cases} -\infty & x_{ij} = 0 \\ \infty & x_{ij} = 1 \end{cases} \tag{18}$$

For convenience, the binary variable is converted from $\{0, 1\}$ to $\{-1, 1\}$ through a linear projection $q_{ij} = 2x_{ij} - 1$. Numerically, $\tilde{\theta}_{ij}$ can be approximated by $m \cdot q_{ij}$ when m is large enough. In this paper, m is set to 10. Under this circumstance, the probability p_{ij} is 1.0 for $x_{ij} = 1$ and 4.5×10^{-5} for $x_{ij} = 0$ according to (17).

With the above definitions, $\tilde{\theta}_{ij}$ and p_{ij} can thereby be regarded as alternative representations of binary variable x_{ij}

Algorithm 1 Framework of SLEMO-NR

Input: *pop* (population size), $\boldsymbol{Y}_{net}$ and $\boldsymbol{A}_{net}$ (time series data), N (number of nodes);
Output: P_{final} (final population);

for $i = 1$ to N **do**
  Obtain the ith slice of $\boldsymbol{Y}_{net}$ and $\boldsymbol{A}_{net}$ as $\boldsymbol{Y}_i$ and $\boldsymbol{A}_i$;
  Set $t \leftarrow 0$ and generate the initial population P^0;
  $[F_1, F_2, \ldots] \leftarrow$ NondominatedSorting(P^0);
  Dis $\leftarrow$ CrowdingDistance$(F_1, F_2, \ldots)$;
  while *termination criteria are not satisfied* **do**
    $P^t_{parents} \leftarrow$ *Selection*$(P^t$, *pop*$)$: Perform the binary tournament selection operator to choose *pop* parents;
    $P^t_{com} \leftarrow P^t \cup$ *OffspringGenerating*$(P^t_{parents})$;// Algorithm 2
    $P^t_{total} \leftarrow P^t_{com} \cup$ *PLSO*(P^t_{com});//Algorithm 3
    // Update population
    Delete duplicated solutions in P^t_{total};
    $[F_1, F_2, \ldots] \leftarrow$ NondominatedSorting(P^t_{total});
    Dis $\leftarrow$ CrowdingDistance$(F_1, F_2, \ldots)$;
    $l \leftarrow$ The minimum value s.t. $|F_1 \cup F_2 \cup \ldots| \geqslant$ *pop*;
    Delete $|F_1 \cup F_2 \cup \ldots| -$ *pop* solutions from F_l with the smallest *Dis*;
    $P^{t+1} \leftarrow F_1 \cup F_2 \cup \ldots F_l$;
    $t \leftarrow t + 1$;
  end while
  $P_{final} \leftarrow$ The final population;
end for
return P_{final}.

in the evolutionary process. Meanwhile, we can also decode x_{ij} from p_{ij} according to the Bernoulli distribution when the alternative representations are used [41].

C. Network Structure Subspace Learning and Offspring Generating

In recent years, LPCA has been proposed in the literature to extract features from binary data [33]. Several successful applications for solving challenging problems appeared in many fields, such as classification of RNA data [42], clustering and representation of multivariate data [43], and recommendation systems [44].

For large-scale NRPs, we aim at finding the structure of the network. However, directly searching for the network structure is extremely hard and time-consuming. In this paper, a subspace containing the features of network structure is learned via LPCA, where the search for network structure is much more efficient. Different from the Pearson's initial formulation of PCA, LPCA seeks to find a rank-d ($d < (N-1)$) projection of the natural parameters from the Bernoulli saturated model and minimize the Bernoulli deviance so that the natural parameters can be projected into a subspace and recovered with little deviance. Suppose there be n data used to learn the subspace, each of which is a $(N-1)$ dimensional vector. We denote the alternative representation of the ith vector as $\tilde{\boldsymbol{\theta}}_i$, then its projection in the subspace is

$$\tilde{\boldsymbol{\theta}}_{is} = \boldsymbol{W}^T \left(\tilde{\boldsymbol{\theta}}_i - \boldsymbol{\mu} \right) \tag{19}$$

and the subspace solution $\tilde{\boldsymbol{\theta}}_{is}$ can be recovered into the original space:

$$\tilde{\boldsymbol{\theta}}_{io} = \boldsymbol{\mu} + \boldsymbol{W} \tilde{\boldsymbol{\theta}}_{is} \tag{20}$$

where $\boldsymbol{\mu} \in \mathbb{R}^N$ is the sample mean and $\boldsymbol{W} \in \mathbb{R}^{N \times d}$ is the principal component loadings matrix. Therefore, the natural parameters are estimated by

$$\hat{\boldsymbol{\theta}}_i = \boldsymbol{\mu} + \boldsymbol{W}\boldsymbol{W}^T \left(\tilde{\boldsymbol{\theta}}_i - \boldsymbol{\mu} \right) \tag{21}$$

It also has the matrix notation:

$$\hat{\boldsymbol{\Theta}} = \mathbf{1}_n \boldsymbol{\mu}^T + \left(\tilde{\boldsymbol{\Theta}} - \mathbf{1}_n \boldsymbol{\mu}^T \right) \boldsymbol{W}\boldsymbol{W}^T \tag{22}$$

where $\tilde{\boldsymbol{\Theta}} \in \mathbb{R}^{n \times N}$, and its ith row is $\tilde{\boldsymbol{\theta}}_i^T$. To solve $\boldsymbol{\mu}$ and $\boldsymbol{W}$, the Bernoulli deviance is minimized:

$$\begin{aligned} D(\boldsymbol{X} \mid \hat{\boldsymbol{\Theta}}) = -2 \left\langle \boldsymbol{X}, \mathbf{1}_n \boldsymbol{\mu}^T + \left(\tilde{\boldsymbol{\Theta}} - \mathbf{1}_n \boldsymbol{\mu}^T \right) \boldsymbol{W}\boldsymbol{W}^T \right\rangle \\ +2 \sum_{i=1}^{n} \sum_{j=1}^{N} \ln \left(1 + \exp \left(\mu_j + \left[\boldsymbol{W}\boldsymbol{W}^T \left(\tilde{\boldsymbol{\theta}}_i - \boldsymbol{\mu} \right) \right]_j \right) \right) \end{aligned} \tag{23}$$

subject to $\boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}_d$, where $\langle \mathbf{A}, \mathbf{B} \rangle = tr(\mathbf{A}^T\mathbf{B})$ is the trace inner product and $\boldsymbol{I}_d$ is the identity matrix.

To effectively represent individuals in the subspace, selecting the appropriate subspace dimension d is of vital importance. The parameter d is selected such that a chosen proportion η, which is set to 95% in this paper, is met or exceeded. Let $\hat{\boldsymbol{W}}_d$ denote the rank-d estimation of the principal component loadings. With the input data $\boldsymbol{X}$, the corresponding rank-d estimation $\hat{\boldsymbol{\Theta}}_d$, and sample mean $\hat{\boldsymbol{\mu}}$, the parameter d is selected as the smallest integer satisfying

$$1 - \frac{D\left(\boldsymbol{X}; \hat{\boldsymbol{\Theta}}_d \right)}{D\left(\boldsymbol{X}; \mathbf{1}_n \hat{\boldsymbol{\mu}}^T \right)} > \eta \tag{24}$$

This criterion can be explained similarly as in standard PCA that at least $100\eta\%$ of the deviance is explained. In this way, little information is lost in the projection between the original space and the subspace. Note that with $(N-1)$ principal component loadings, we have $\hat{\boldsymbol{W}}_{N-1}^T \hat{\boldsymbol{W}}_{N-1} = \boldsymbol{I}_{N-1}$ and $D\left(\boldsymbol{X}; \tilde{\boldsymbol{\Theta}}_{N-1} \right) = 0$. Therefore, 100% of the deviance is explained by $(N-1)$ components as expected.

Majorization-minimization (MM) [45] is an approach to minimizing (23) iteratively. In the beginning, set t = 0. Let $\tilde{\boldsymbol{\Theta}}_c^{(t)} := \tilde{\boldsymbol{\Theta}} - \mathbf{1}_n \left(\boldsymbol{\mu}^{(t)} \right)^T$ be the column-centered saturated parameters. As recommended in [46], $\boldsymbol{\mu}^{(0)}$ is initialized to the column means of $\tilde{\boldsymbol{\Theta}}$, $\boldsymbol{W}^{(0)}$ is initialized to the first d right singular vectors of $\tilde{\boldsymbol{\Theta}}_c^{(0)}$, and the following steps are repeated for 20 times. At time t:

1) Calculate the working variables

$$v_{ij}^{(t+1)} = \hat{\theta}_{ij}^{(t)} + 4 \left[x_{ij} - \text{sigmoid} \left(\hat{\theta}_{ij}^{(t)} \right) \right] \tag{25}$$

where

$$\hat{\theta}_{ij}^{(t)} = \mu_j^{(t)} + \left[\boldsymbol{W}^{(t)} \left(\boldsymbol{W}^{(t)} \right)^T \left(\tilde{\boldsymbol{\theta}}_i - \boldsymbol{\mu}^{(t)} \right) \right]_j \tag{26}$$

2) Update $\boldsymbol{\mu}^{(t+1)}$

$$\boldsymbol{\mu}^{(t+1)} = \left(\boldsymbol{V}^{(t+1)} - \tilde{\boldsymbol{\Theta}} \boldsymbol{W}^{(t)} \left(\boldsymbol{W}^{(t)} \right)^T \right)^T \mathbf{1}_n / n \tag{27}$$

where $\boldsymbol{V}^{(t+1)}$ is a matrix whose element is $v_{ij}^{(t+1)}$.

3) Let $\boldsymbol{V}_c^{(t+1)} := \boldsymbol{V}^{(t+1)} - \mathbf{1}_n \left(\boldsymbol{\mu}^{(t+1)} \right)^T$ be the adjusted responses. Perform eigen decomposition of

$$\begin{aligned} \left(\tilde{\boldsymbol{\Theta}}_c^{(t+1)} \right)^T \boldsymbol{V}_c^{(t+1)} + \left(\boldsymbol{V}_c^{(t+1)} \right)^T \tilde{\boldsymbol{\Theta}}_c^{(t+1)} \\ - \left(\tilde{\boldsymbol{\Theta}}_c^{(t+1)} \right)^T \tilde{\boldsymbol{\Theta}}_c^{(t+1)} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T \end{aligned} \tag{28}$$

and update $\boldsymbol{W}^{(t+1)}$ with the first k eigenvectors in $\boldsymbol{Q}$.

Afterward, the subspace containing network structure features is learned. By means of the principal component loadings matrix $\boldsymbol{W}$ and sample mean $\boldsymbol{\mu}$, the natural parameters of each solution can be mapped between the original space and the subspace. In the evolutionary process, the offspring are generated in the learned network structure subspace first, and mapped back into the original space to be evaluated.

The non-dominated solutions, which are superior to other solutions, are considered to contain more features of the network structure. They are thereby adopted as an approximation of network structure for learning the subspace. Thereafter, each time two randomly selected individuals from the parent population are projected into the subspace by (17), where two offspring solutions are generated via the single-point crossover and bitwise mutation genetic operators. The features of network structure learned in the subspace are transferred into the original space through recovering offspring solutions

with (20). Note that the natural parameters, as the alternative representation of binary vectors, are used in the genetic operator. The overall procedure of subspace learning and offspring generating is presented in Algorithm 2.

Algorithm 2 OffspringGenerating

Input: $P_{parents}$ (selected parent population);
Output: O (generated offspring population);

$P_{alter} \leftarrow$ Obtain the natural parameters of $P_{parents}$;
Train LPCA with the non-dominated solutions in P_{alter};
$O \leftarrow \varnothing$;
while $P_{alter} \neq \varnothing$ **do**
  $[\boldsymbol{\theta}_a, \boldsymbol{\theta}_b] \leftarrow$ Randomly select two vectors from P_{alter};
  $P_{alter} \leftarrow P_{alter} \backslash \{\boldsymbol{\theta}_a, \boldsymbol{\theta}_b\}$
  $[\boldsymbol{x}_{as}, \boldsymbol{x}_{bs}] \leftarrow$ Project $\boldsymbol{\theta}_a$ and $\boldsymbol{\theta}_a$ by (19), calculate their Bernoulli probability by (17) and decode to binary individuals;
  $[\boldsymbol{x}_p, \boldsymbol{x}_q] \leftarrow$ Apply the single-point crossover and bitwise mutation on $\boldsymbol{x}_{as}$ and $\boldsymbol{x}_{bs}$;
  $[\boldsymbol{x}_{o1}, \boldsymbol{x}_{o2}] \leftarrow$ Obtain the natural parameters of $\boldsymbol{x}_p$ and $\boldsymbol{x}_q$, recover them by (20), calculate their Bernoulli probability, and decode to binary individuals;
  $O \leftarrow O \cup \{\boldsymbol{x}_{o1}, \boldsymbol{x}_{o2}\}$;
end while
return O.

D. Preference-Based Local Search Operator

In order to enhance the search ability, some approaches employ a local search operator, beyond crossover and mutation, to get the individuals further perturbed [24], [47]. The iterative soft-thresholding (IST), as a representative local search operator, can effectively search for new solutions along the gradient descent direction. Nevertheless, the IST is performed on real-coded variables, and therefore cannot be used directly in our approach. Benefitting from the alternative representations, we propose a PLSO combining the IST and the preference-based strategy to make our approach concentrate more on finding the k-sparse solutions. In the PLSO, the IST is first performed on a selected solution to search for a new solution. Then the preference-based strategy is performed to generate a group of solutions near the new solution. The details of the PLSO is described as follows.

1) Iterative soft-thresholding search: The iterative soft-thresholding search [48] is designed to solve the relaxed L_1 regularization problem, which has the following formulation:

$$\min_{\boldsymbol{X}_i} \left(\frac{1}{2} \left\| \boldsymbol{A}_i \boldsymbol{X}_i - \boldsymbol{Y}_i \right\|_2^2 + \lambda \left\| \boldsymbol{X}_i \right\|_1 \right) \tag{29}$$

There are three major steps in IST: gradient descent, parameter setting, and soft-thresholding truncation. However, it is impossible to directly perform gradient descent since the individuals are binary-coded. Thus, the alternative representation takes the place of original individuals to perform the preference-based local search. According to the definitions in Section III.B, we have

$$x_{ij} = \frac{1}{2} + \frac{1}{2m} \ln \frac{p_{ij}}{1 - p_{ij}} \tag{30}$$

where x_{ij} and p_{ij} $(j = 1, 2, ..., N-1)$ are the jth element of $\boldsymbol{X}_i$ and $\boldsymbol{P}_i$, respectively. In IST, a new solution $\boldsymbol{P}_i^{r+1}$ is obtained from the previous solution $\boldsymbol{P}_i^r$. Let $f(\boldsymbol{X}_i)$ be the first term of (29), which becomes $f(\boldsymbol{P}_i^r)$ after the replacement of variables. In the following, the major steps of IST are introduced in brief.

a) *Gradient Descent:* For the current solution $\boldsymbol{P}_i^r$ (corresponding to $\boldsymbol{X}_i^r$), its trial solution $\tilde{\boldsymbol{P}}_i$ is generated by

$$\begin{aligned} \tilde{\boldsymbol{P}}_i &= \boldsymbol{P}_i^r - \frac{1}{\gamma^r} \nabla f\left(\boldsymbol{P}_i^r\right) \\ &= \boldsymbol{P}_i^r - \frac{1}{\gamma^r} \left[\boldsymbol{A}_i^T \left(\boldsymbol{A}_i \boldsymbol{X}_i^r - \boldsymbol{Y}_i \right) \right] \odot \boldsymbol{\Omega}_i \end{aligned} \tag{31}$$

$$\begin{aligned} \boldsymbol{\Omega}_i &= \left(\frac{\partial x_{i1}^r}{\partial p_{i1}^r}, \ldots, \frac{\partial x_{iN}^r}{\partial p_{iN}^r} \right)^T \\ &= \frac{1}{2m} \left(\frac{1}{p_{i1}^r \left(1 - p_{i1}^r\right)}, \ldots, \frac{1}{p_{iN}^r \left(1 - p_{iN}^r\right)} \right)^T \end{aligned} \tag{32}$$

where $\nabla f(\boldsymbol{P}_i^r)$ is the gradient of $f(\boldsymbol{P}_i^r)$, "$\odot$" is the Hadamard product, and $\boldsymbol{\Omega}_i$ is a vector containing the partial derivative of x_{ij}^r to p_{ij}^r.

b) *Parameter Setting:* The parameter γ^r is calculated by optimizing the Barizilai-Borwein equation [49]:

$$\gamma^r = \arg\min_{\gamma} \left\| \gamma \boldsymbol{s}^r - \boldsymbol{\tau}^r \right\|_2^2 = \frac{\left(\boldsymbol{s}^r\right)^T \boldsymbol{\tau}^r}{\left(\boldsymbol{s}^r\right)^T \boldsymbol{s}^r} \tag{33}$$

where $\boldsymbol{s}^r = \frac{1}{2m} \left(\ln \frac{p_{i1}^r \left(1 - p_{i1}^{r-1}\right)}{\left(1 - p_{i1}^r\right) p_{i1}^{r-1}}, \ldots, \ln \frac{p_{iN}^r \left(1 - p_{iN}^{r-1}\right)}{\left(1 - p_{iN}^r\right) p_{iN}^{r-1}} \right)^T$ and $\boldsymbol{\tau}^r = \nabla f\left(\boldsymbol{P}_i^r\right) - \nabla f\left(\boldsymbol{P}_i^{r-1}\right)$.

c) *Soft-thresholding Truncation:*

$$\boldsymbol{P}_i^{r+1} = \text{soft}\left(\tilde{\boldsymbol{P}}_i, \frac{\lambda_{st}}{\gamma^r} \right) \tag{34}$$

where

$$\begin{aligned} \text{soft}(a, b) &= \text{sgn}(a) \max\{|a| - b, 0\} \\ &= \begin{cases} \text{sgn}(a)(|a| - b) & \text{if } |a| > b \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{35}$$

As suggested in [50], λ_{st} is generated by a decreasing sequence of $\{\lambda_r, r = 0, 1, \ldots, n\}$, where $\lambda_0 > \lambda_1 > \ldots > \lambda_n$, $\lambda_0 = 0.1 \left\| \boldsymbol{A}_i^T \boldsymbol{Y}_i \right\|_\infty$, and $\lambda_n \to 0$.

2) Preference-Based Local Search Strategy: MOEAs aim to find an evenly distributed Pareto front considering the balance of exploration and exploitation in the evolutionary process. Whereas in practice, approximating the region far from the k-sparse solution is not helpful to infer the network structure. To amend this drawback, the preference-based strategy is employed to focus on a local part of the weakly Pareto front containing the optimal k-sparse solution. The searching efficiency is thus improved a great deal.

In PLSO, there are two parts of information to be maintained.

a) The archived population E, which contains part of the solutions in the population. It is divided into two subsets with a cut-off:

$$E_1 = \left\{ \boldsymbol{X} \in E \mid f_1(\boldsymbol{X}) < f_1^{\min} + \varepsilon \right\} \tag{36}$$

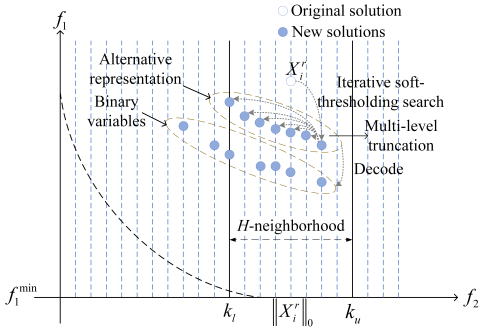


Fig. 2. IST with multi-level truncation in PLSO.

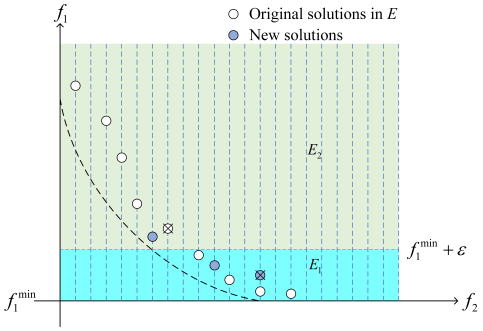


Fig. 3. Graphical illustration of the archived population E and its update.

$$E_2 = \left\{ \boldsymbol{X} \in E \mid f_1(\boldsymbol{X}) \geq f_1^{\min} + \varepsilon \right\} \tag{37}$$

where f_1^{min} is the minimal value of the first objective function found by now in *E*, and ε is a parameter with a positive value.

b) The H-neighborhood, which describes a small region near a solution $\boldsymbol{X}_i$ with sparsity level $[\|\boldsymbol{X}_i\|_0 - H, \|\boldsymbol{X}_i\|_0 + H]$, is considered to ensure the effectiveness of the generated solutions. Here *H* is the neighborhood size.

At first, the archived population E needs to be determined. Since the non-dominated solutions in P_{com}^t are more approximate to optimal solutions, they are adopted to form E and generate new solutions through preference-based local search. Then, the subset E_1 is determined according to (36). The parameter ε, usually with a rather small positive value, is adjusted to control the quality of solutions selected for the following soft-thresholding procedure. The value of objective functions varies from problem to problem, so ε is set to the percentage of $(f_1^{max} - f_1^{min})$, where f_1^{max} and f_1^{min} are the maximum and minimum values of the first objective function found by now in *E*, respectively. For example, $\varepsilon = 10$ indicates that $E_1 = \left\{ X \in E \mid f_1(X) < \left[f_1^{\min} + 10\% \times \left(f_1^{\max} - f_1^{\min} \right) \right] \right\}$ and E_2 is set in the similar way. The first objective function f_1 reveals the deviance of solutions from true network structure to some extent, which means it should be optimized with priority. With respect to f_1, solutions in E_1 are superior to those in E_2. In this way, a starting solution $\boldsymbol{X}_i^r$ is randomly selected from E_1. An H-neighborhood of $\boldsymbol{X}_i^r$ is thereafter determined with $k_l = \|\boldsymbol{X}_i^r\|_0 - H$ and $k_u = \|\boldsymbol{X}_i^r\|_0 + H$ as the lower and upper bounds, respectively.

After the above-mentioned preparations, IST with multi-level truncation is implemented to generate new solutions, which is illustrated in Fig. 2. To choose the adaptive step size for the gradient descent operation, a supportive solution is selected from P_{com}^t. As proved by Li *et al.* [47], if $\boldsymbol{X}_i^r$ is chosen from non-dominated solutions and $\boldsymbol{X}_i^{r-1}$ is chosen from dominated solutions, there is a higher probability to generate improved non-dominated solutions via IST. Therefore, $\boldsymbol{X}_i^{r-1}$ is deliberately selected from solutions dominated by $\boldsymbol{X}_i^r$. Note that $\boldsymbol{X}_i^r$ and $\boldsymbol{X}_i^{r-1}$ are binary-coded. They need to be represented by their probability distribution before performing gradient descent. After that, a new solution $\boldsymbol{P}_i^{r+1}$ is generated by IST.

Following the IST search procedure, which only generates one solution at a time, the main purpose of multi-level truncation is to obtain a group of solutions in the H-neighborhood of $\boldsymbol{X}_i^r$. By truncating the new solution $\boldsymbol{P}_i^{r+1}$ at different sparsity levels, no more than $2 \times H$ solutions with sparsity in the range of $[k_l, k_u]$ are obtained and added to $\tilde{E}$ to avoid too much extra computational cost.

At last, all the generated solutions in $\tilde{E}$ are used to update the archived population E, which is illustrated in Fig. 3. It should be noted that the subsets E_1 and E_2 are updated separately. The newly generated solutions are first added to E, and the inferior solutions are removed according to some principles. Then separate E into E_1 and E_2. Moreover, as the k-sparse solution is more likely to be located in the center of the H-neighborhood, the solutions in E with the smallest and largest sparsity levels are not preferred. The details of the PLSO is described in Algorithm 3.

IV. Experiments

In this section, a series of experiments are conducted to empirically validate the performance of our algorithm. Section IV.A gives the experimental settings. We have designed 16 problems based on four kinds of benchmark networks, and the configurations are shown in Section IV.B. In Section IV.C, the performance of the network structure subspace learning approach is verified. Then, Section IV.D demonstrates the effectiveness of the PLSO. The analysis about the effect of two major parameters in our algorithm is further conducted in Section IV.E. After that, we compare SLEMO-NR with six state-of-the-art methods on six real-world networks in Section IV.F. Finally, the computational efficiency analysis is given in Section IV.G.

A. Experimental Setup

To quantify the performance of network reconstruction methods, two common indices are adopted, namely the area under the receiver operating characteristic curve (AUC) [51] and the Matthews correlation coefficient (MCC) [52]. Both of them are capable of producing a truthful and informative score in evaluating the performance.

In the experiments, an initialization operator based on LASSO is employed to generate the initial population. In [23], the initialization operator was proved to be capable of accelerating the convergence speed of EA. It should be noted that

Algorithm 3 Preference-BasedLocalSearchOperator (PLSO)

Input: P_{com}^t (combined population);
Output: P_{local}^t (generated population);
$E \leftarrow$ Non-dominated solutions in P_{com}^t;
Determine E_1 according to (36);
$\boldsymbol{X}_i^r \leftarrow$ Randomly select a starting solution from E_1;
Determine sparsity levels in an H-neighborhood: $k_l = \|\boldsymbol{X}_i^r\|_0 - H$, $k_u = \|\boldsymbol{X}_i^r\|_0 + H$;
// Iterative soft-thresholding search
$\boldsymbol{X}_i^{r-1} \leftarrow$ Randomly select a supporting solution dominated by $\boldsymbol{X}_i^r$ from P_{com}^t;
$\tilde{\boldsymbol{P}}_i \leftarrow$ Obtain the alternative representations of $\boldsymbol{X}_i^r$ and $\boldsymbol{X}_i^{r-1}$, and generate a trial solution using (31);
$\tilde{\boldsymbol{P}}_i^{r+1} \leftarrow$ Generate a new solution by (34);
// Multilevel truncation
Set $\tilde{E} \leftarrow \varnothing$ and $\boldsymbol{z} \leftarrow \boldsymbol{P}_i^{r+1}$;
Set the smallest $(\|\boldsymbol{z}\|_0 - k_u)$ nonzero elements of $\boldsymbol{z}$ as zeros if $\|\boldsymbol{z}\|_0 > k_u$;
while $\|\boldsymbol{z}\|_0 > k_l$ **do**
    Set the smallest nonzero element of $\boldsymbol{z}$ as zero, and $\tilde{E} \leftarrow \tilde{E} \cup \boldsymbol{z}$;
end while
Decode the solutions in $\tilde{E}$ into binary-coded individuals;
// Update archived population E
for each $\boldsymbol{z} \in \tilde{E}$ **do**
    if $f_1(\boldsymbol{z}) < f_1^{\min} + \varepsilon$ **then**
        If $\exists \boldsymbol{z}' \in E$ with the same sparsity as $\boldsymbol{z}$, then set $\boldsymbol{z}'$ as the one with smaller f_1; otherwise, $E \leftarrow E \cup \boldsymbol{z}$;
    else
        Add $\boldsymbol{z}$ into E and remove dominated solutions in E;
    end if
end for
Separate E into E_1 and E_2 according to (36) and (37);
If $|E_1| > H$, then $|E_1| - H$ solutions with the largest sparsity levels are removed from E_1;
If $|E_2| > H$, then $|E_2| - H$ solutions with the smallest sparsity levels are removed from E_2;
$P_{local}^t \leftarrow E_1 \cup E_2$;
return P_{local}^t.

any state-of-the-art L_1-minimization methods can be adopted, such as the orthogonal matching pursuit (OMP) [53] and fast iterative shrinkage thresholding algorithm (FISTA) [54]. For fair comparisons, this initialization operation is implemented in all the EA-based algorithms to guarantee an equal and better initial state. For the L_1-minimization problems, the choice of regularization parameter λ is a critical problem. With an ill-chosen parameter, the results can be far away from the true network structure. Whereas in practice, we can still generate a group of solutions via LASSO with different λ values. For each individual in the initial population, we uniformly sample a value between 0 and 1 as λ. To maintain the diversity of the initial population and accelerate the speed, only a few iterations, e.g. 20 iterations used in this paper, is needed for LASSO initialization. After that, a cut-off value of 0.5 is used to transform the real-coded results of LASSO into binary-coded individuals.

Besides, the proposed SLEMO-NR is a MOEA, and it provides a group of candidate solutions. The first objective is the reconstruction error and the second objective is the sparsity level. In the optimization process, the real network structure is unknown, and we cannot compare two solutions directly. The reconstruction error can show how close a solution is from the optimal solution to some extent, so it can reflect the superiority of a solution over other solutions. Considering the first objective function is a simple and effective way when selecting solutions. Therefore, in this paper, we select the individual in P_{final} with the smallest value of the first objective function in each subproblem to compose the final reconstructed network.

The parameters in SLEMO-NR are set as follows. For all the experiments, the population size is set to 100 and the number of function evaluations is set to 20000. With respect to the single-point crossover and bitwise mutation, the crossover and mutation probabilities are set to 1 and $1/(N-1)$, respectively, where $(N-1)$ is the dimension of decision variables. The major parameters in the PLSO are H and ε, both of which are set to 1.

B. Test Problems

Based on four kinds of benchmark networks, namely Erdős-Rényi random networks (ER) [55], Barabási-Albert scale-free networks (BA) [56], Newman-Watts small-world networks (NW) [57], and Watts-Strogatz small-world networks (WS) [58], we have designed 16 problems varying in network models, network types, the number of nodes N and average degree of nodes $\langle D \rangle$. The detailed configurations of the 16 designed are presented in Table I. Problems 1-8 are the combination of two network models and four network types with N = 300 and $\langle D \rangle$ = 20. Problems 9-16 have the same configuration except that N = 500 and $\langle D \rangle$ = 50. Besides, for all problems the experiments are conducted under different length of time points with N_M = 0.3 and 0.5, where N_M is the total length of time points M divided by network size N.

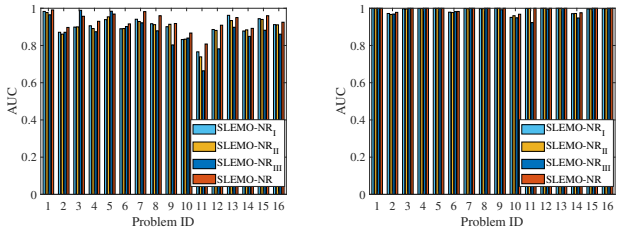
For each test problem, 30 independent runs are conducted for each MOEA to obtain statistical results. Moreover, to maintain the reliability of observed results, the Wilcoxon rank-sum test with a significance level of 0.05 is utilized, where '+', '−', and '≈' indicate that the result of another approach is significantly better, significantly worse, and statistically similar to that of SLEMO-NR, respectively.

C. Effectiveness of LPCA in Offspring Generating

To validate the effectiveness of LPCA, SLEMO-NR is compared with its three variants. We denote SLEMO-NR without LPCA, SLEMO-NR with RBM [59], and SLEMO-NR with ordinary PCA as SLEMO-NR$_{\mathrm{I}}$, SLEMO-NR$_{\mathrm{II}}$, SLEMO-NR$_{\mathrm{III}}$, respectively. Particularly, SLEMO-NR$_{\mathrm{III}}$ utilizes real-coded variables during the whole evolutionary process to maintain the effectiveness of PCA. Without LPCA, the searching process is conducted in the original space, which indicates that the offspring are generated in the same way as that in NSGA-II. Moreover, although many ML-based methods are

TABLE I
SIXTEEN PROBLEMS WITH DIFFERENT CONFIGURATIONS (NETWORK MODEL, NETWORK TYPE, NUMBER OF NODES, AVERAGE DEGREE OF NODES).

Problem ID	Configuration	Problem ID	Configuration
1	(EG, ER, 300, 20)	9	(EG, ER, 500, 50)
2	(EG, BA, 300, 20)	10	(EG, BA, 500, 50)
3	(EG, NW, 300, 20)	11	(EG, NW, 500, 50)
4	(EG, WS, 300, 20)	12	(EG, WS, 500, 50)
5	(RN, ER, 300, 20)	13	(RN, ER, 500, 50)
6	(RN, BA, 300, 20)	14	(RN, BA, 500, 50)
7	(RN, NW, 300, 20)	15	(RN, NW, 500, 50)
8	(RN, WS, 300, 20)	16	(RN, WS, 500, 50)

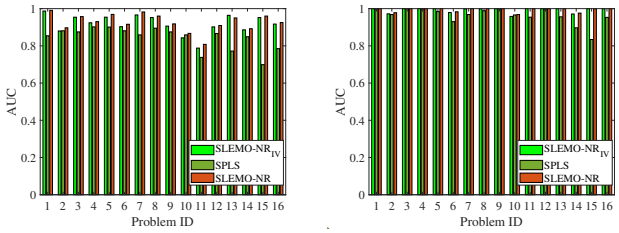


(a) (b)

Fig. 4. The comparison of SLEMO-NR with SLEMO-NR$_{\rm I}$, SLEMO-NR$_{\rm II}$, and SLEMO-NR$_{\rm III}$ in terms of AUC on 16 test problems when (a) N_M = 0.3 and (b) N_M = 0.5.

capable of learning subspaces, most of them can not deal with binary variables and are not recoverable apart from RBM. They are thereby unable to be used in the genetic operator.

Figs. 4 and S1 (in Supplementary Materials) illustrate the comparisons of SLEMO-NR with its three variants in terms of the average values of AUC and MCC on Problems 1-16 when N_M = 0.3 and 0.5. The detailed numerical results are shown in Supplementary Materials S.I. As can be seen from the figures, the original SLEMO-NR is significantly better or statistically similar in terms of AUC and MCC in 29 out of 32 cases and loses once to SLEMO-NR$_{\rm I}$. This is possibly attributed to the network structure subspace learning and the consideration of information loss in the projection process. With LPCA in our SLEMO-NR, searching for optimal solutions is conducted in a subspace containing the features of network structure, thus SLEMO-NR is more effective than SLEMO-NR$_{\rm I}$. Furthermore, the selection of the subspace dimension for LPCA guarantees that little information is lost in the projection between the original space and the subspace. Whereas in SLEMO-NR$_{\rm II}$, the subspace dimension is chosen according to an adaptation strategy in [60], which has been proved to be more effective than setting to a constant value. As shown in the figures, SLEMO-NR$_{\rm I}$ outperforms or matches SLEMO-NR$_{\rm II}$ in 26 out of 32 cases in terms of AUC and in 24 out of 32 cases in terms of MCC, indicating that RBM with the adaptation strategy of subspace dimension may not be able to effectively capture the feature of network structure and may even harm the performance in some cases. Besides, SLEMO-NR matches or exceeds SLEMO-NR$_{\rm III}$ in 30 out 32 cases, which not only shows the effectiveness of LPCA, but also demonstrates the advantage of using binary-coded variables.



(a) (b)

Fig. 5. The comparison of SLEMO-NR with SLEMO-NR$_{\rm IV}$ and SPLS in terms of AUC on 16 test problems when (a) N_M = 0.3 and (b) N_M = 0.5.

D. Effectiveness of Preference-Based Local Search Operator

In this section, we demonstrate the effectiveness of the PLSO on SLEMO-NR. Experiments are conducted on Problems 1-16 when N_M = 0.3 and 0.5 as before. SLEMO-NR$_{\rm IV}$ denotes SLEMO-NR without the PLSO, which is adopted as a comparison. Besides, SPLS [48], as a representative IST-based MOEA, is also compared in this section.

Figs. 5 and S2 (in Supplementary Materials) illustrate the comparison of SLEMO-NR with SLEMO-NR$_{\rm IV}$ and SPLS in terms of the average values of AUC and MCC on Problems 1-16 when N_M = 0.3 and 0.5. The detailed numerical results can be found in Supplementary Materials S.II. The original SLEMO-NR matches or exceeds SLEMO-NR$_{\rm IV}$ and SPLS in 31 out of 32 cases in terms of AUC. With respect to MCC, SLEMO-NR losses once to SLEMO-NR$_{\rm IV}$ and twice to SPLS. These comparisons indicate the effectiveness of the PLSO. There are a few cases that this operator has a side effect on SLEMO-NR. This is possibly due to the fact that the PLSO leads the population to a local optimal area since it is based on a gradient descent approach and the NRP is non-convex.

Compared with SLEMO-NR, SPLS fails to get satisfying performance. Especially in some cases when SLEMO-NR has already perfectly reconstructed the network, the result of SPLS is still far from the true network. Although both of them utilize IST and the preference-based strategy, there are two major differences. First, SLEMO-NR selects a supportive solution to get adaptive step size for IST, while in SPLS the step size is constant. Second, the search area of SPLS is restricted to a small region near the solutions generated by IST. While Algorithm 3 acts as a local search operator and is performed on the current population generated by genetic operators in each generation, which combines the search ability of both genetic operators and IST, and improves the capability of searching for optimal solutions compared with SPLS.

E. Analysis of the Parameter Sensitivity

In this subsection, the effect of two parameters on the performance of SLEMO-NR is investigated. The search ability of PLSO is affected by the two parameters. The first parameter ε, which determines the cut-off between two subsets E_1 and E_2, affects the selection of solutions to perform IST. It should be noted that when ε = 0, the selection of solution to perform IST is restricted to the one with the smallest value of the first objective function. The second parameter is the neighborhood

size H in the PLSO, which controls the breadth of the search. With a larger value of H, the generated solutions through the multi-level truncation may be farther away from the originally selected solution. If H is set to 0, the PLSO becomes invalid.

To analyze the sensitivity of parameters, Problems 3, 8, 9, and 14 are selected as representative problems. For any two of them, there are at least two differences in the configurations. The experimental results are summarized as follows.

1) Fig. 6 shows the performance of our algorithm on the four problems in terms of MCC under different N_M, where ε takes the value from $\{0, 1, 5, 10, 20\}$, and H is set to 1. From the figures, we can observe that SLEMO-NR has the optimal or at least the suboptimal performance when ε = 1. When ε = 5 or 10, we may obtain a good performance occasionally, but the overall performance is not good enough.
2) Fig. 7 shows the performance of our algorithm on the four problems in terms of MCC under different N_M, where H takes the value from $\{0, 1, 5, 10, 20\}$ and ε is set to 1. As can be seen from the figures, the setting of H has an impact on the performance, especially when the length of data is not large enough. With H setting to 1, our algorithm achieves the best performance in all but one case. Moreover, when H is set to a relatively small value, we can obtain better performance compared to H = 0 (without the PLSO) in most cases.

According to the experiments conducted in this subsection, these parameters affect the performance of SLEMO-NR to some extent. When the amount of data is sufficient, such as $N_M \geqslant 0.5$ in Problems 3, 8, and 9 and $N_M \geqslant 0.6$ in Problem 14, SLEMO-NR can always fully reconstruct the networks whatever the choice of these parameters. Whereas in the other cases when the amount of data is limited, properly selected parameters enhance the overall performance significantly. With both the parameters setting to small values, the PLSO is mainly conducted near the solutions with smaller values of the first objective function. This helps perform deeper search and find non-dominated solutions. Note that the characteristics of different NRPs may vary greatly, so the proper combinations of parameters in one NRP maybe not suitable for the others and need to be adjusted accordingly.

F. Application to Real-world Networks

In this subsection, we validate the performance of SLEMO-NR on 6 real-world networks, including polbooks, USAir97, 130bit, bibd-12-5, TSOPF_RS_b162_c1 (take TSOPF for short) [61], and email-Eu-core [62]. The number of nodes N and the number of links L of these networks adopted in this paper are presented in Table II. The dataset used to reconstruct the networks is generated using EG and RN models described in Section II, and N_M is set to $\{0.2, 0.3, 0.4, 0.5\}$ for each network. The performance of SLEMO-NR is compared with 6 state-of-the-art methods. As for the ML-based approaches, we select CS, LASSO, and OMP as a contrast. Furthermore, three EA-based algorithms are employed in the experiments as comparisons. MAST-Net [24] is a memetic algorithm specially designed for NRPs, which has shown superiority on small-scale networks. SparseEA [63] and MOEA/PSL [60] have been validated to be effective on large-scale MOPs, and are implemented on the evolutionary multiobjective optimization platform PlatEMO [64]. For the comparative methods, we use identical parameters suggested in the original papers and codes. For all EA-based methods, the number of function evaluations is set to $100 \times N$ for fair comparisons. All experimental results in terms of AUC and MCC are obtained by averaging over 30 runs on each dataset. The Wilcoxon rank-sum test with a significance level of 0.05 is utilized, where '$+$', '$-$', and '$\approx$' indicate that the result of another approach is significantly better, significantly worse, and statistically similar to that of SLEMO-NR, respectively.

TABLE II
DESCRIPTION OF REAL-WORLD NETWORKS USED IN THIS PAPER

Name	N	L
polbooks	105	441
USAir97	332	2126
130bit	584	6120
bibd-12-5	792	7920
email-Eu-core	1005	25571
TSOPF	5376	205399

The results of application to the EG network reconstruction are shown in Tables III and SV (in Supplementary Materials S.III). In terms of AUC, SLEMO-NR matches or exceeds the other methods in 21 out of 24 cases and loses three times to MAST-Net on polbooks and TSOPF networks when N_M = 0.2 and 0.3. In terms of MCC, SLEMO-NR matches or exceeds the other methods in 23 out of 24 cases and loses once to MAST-Net on USAir97 network when N_M = 0.3. The results of application to RN network reconstruction are shown in Tables IV and SVI (in Supplementary Materials S.III). In terms of both AUC and MCC, SLEMO-NR matches or exceeds all the other methods in all cases. To sum up, for large-scale NRPs, SLEMO-NR is better than the other state-of-the-art methods due to the effectiveness of the subspace learning approach and the PLSO. Especially when the amount of data is limited, the superiority of SLEMO-NR over other methods is prominent. Moreover, SLEMO-NR can not only handle large-scale NRPs, but also outperform other methods in most cases when dealing with small-scale NRPs. These results demonstrate the significant performance of SLEMO-NR.

Besides, we use four exemplary cases, which are subproblems selected from polbooks and 130bit datasets based on EG model with N_M = 0.2 and N_M = 0.4 respectively, to demonstrate our algorithm's capacity to reconstruct networks. For a certain subproblem, we plot all the non-dominated solutions obtained by MOEAs and the best solutions obtained by other methods. The results are shown in Fig. 8. As illustrated in these figures, it can be clearly observed that SLEMO-NR concentrates more on the regions near the optimal solutions and can obtain solutions with the smallest reconstruction error in most cases. For the compared methods, they are likely to get stuck in small regions far from the optimal solutions and thus getting unsatisfying performances.

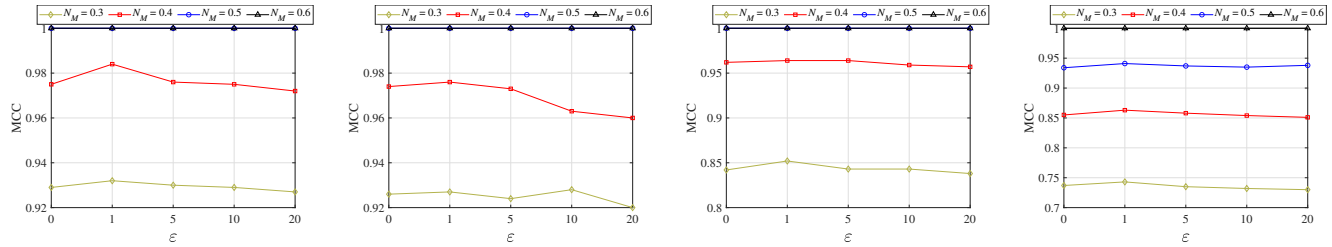


Fig. 6. Performance of our algorithm with different values of ε in terms of MCC on (a) Problem 3, (b) Problem 8, (c) Problem 9, and (d) Problem 14.

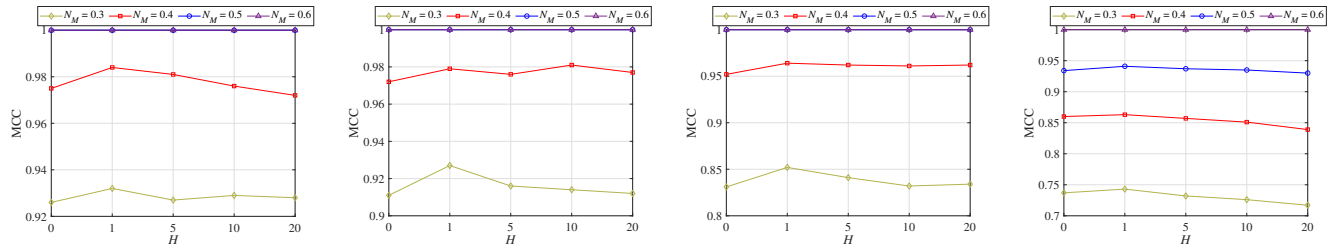


Fig. 7. Performance of our algorithm with different values of H in terms of MCC on (a) Problem 3, (b) Problem 8, (c) Problem 9, and (d) Problem 14.

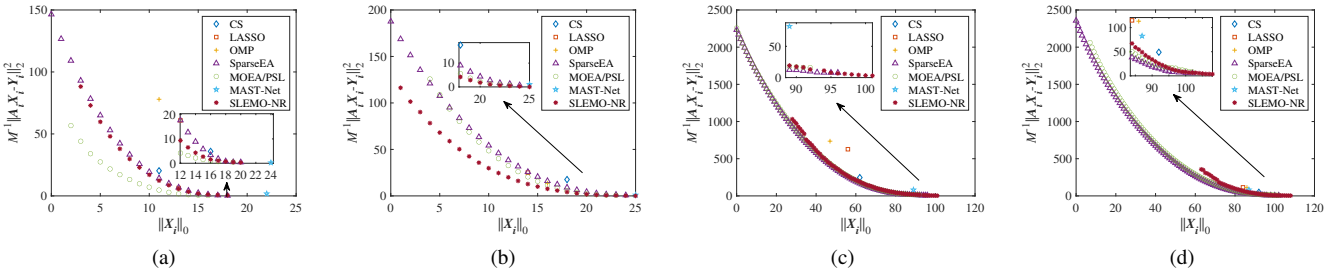


Fig. 8. The reconstruction results on (a) polbooks with $N_M = 0.2$, (b) polbooks with $N_M = 0.4$, (c) 130bit with $N_M = 0.2$, and (d) 130bit with $N_M = 0.4$.

G. Computational Efficiency Analysis

Lastly, the computational complexity of the proposed algorithm and its runtime comparisons with other algorithms are analyzed. In the initialization process, we use LASSO in this paper and its time complexity is $O(pop(N^3 + MN^2))$, where pop, M, and N denote the population size, the length of time points, and the dimension of decision variables, respectively. The non-dominate sort requires $O(pop^2)$ basic operations. The binary tournament selection operator requires $O(pop)$ basic operations. In offspring generating, the time complexity of training LPCA is $O(N^3)$. After that, the time complexity of generating offspring solutions through the crossover and mutation is $O(popNd)$, where d denotes the dimension of decision variables in the subspace. Finally, the preference-based local search process can be divided into three main parts. The time complexity of IST is $O((M+1)N^2)$. Then, the time complexity of multilevel truncation is $O(NlogN)$ for sorting nonzero elements. While updating archived population E, non-dominated sorting is used $2 \times H$ times in the worst case, so the time complexity is $O(Hpop^2)$, where H is half the size of E. The population update process is the same as that of NSGA-II, which has a complexity of $O(pop^2)$ [39]. Furthermore, all the above steps are the optimization process of a subproblem, and the whole NRP is consisted of N subproblems. In summary, the overall computation complexity of SLEMO-NR in one generation is $O(N^4 + popN^2d + HNpop^2)$.

Besides, the runtimes (in seconds) of all the compared algorithms are shown in Table V, where all the experiments are conducted on the six real-world datasets based on RN model with N_M = 0.4. Compared with ML-based algorihms, EA-based algorithms take longer runtime because of maintaining a group of solutions, but they can obtain better results in most cases. In order to fairly compare the ML-based and EA-based algorithms, we increase the maximum number of iterations of the ML-based algorithms to make them consume the same computation cost as EAs, but their performance fail to get improved. MAST-Net is a single objective EA, so it takes a little shorter time to run compared with MOEAs. Since SLEMO-NR needs to learn network structure subspaces and perform preference-based local search, its runtime is slightly longer than that of the two other MOEAs in three out of six cases. In short, the proposed SLEMO-NR has competitive efficiency compared to the other algorithms.

TABLE III
THE COMPARISON OF SLEMO-NR AGAINST OTHER METHODS ON THE SIX REAL-WORLD DATASETS BASED ON THE EG MODEL IN TERMS OF AUC. THE BEST RESULT IN EACH ROW IS SHOWN IN BOLDFACE.

N_M	Dataset	CS	LASSO	OMP	SparseEA	MOEA/PSL	MAST-Net	SLEMO-NR
0.2	polbooks	0.711	0.664	0.613	0.725	0.747	**0.811**	0.760
	USAir97	0.770	0.762	0.678	0.787	0.793	0.783	**0.814**
	130bit	0.900	0.891	0.817	0.895	0.900	0.919	**0.923**
	bibd_12_5	0.828	0.819	0.758	0.837	0.833	0.854	**0.865**
	email-Eu-core	0.886	0.826	0.763	0.851	0.875	0.877	**0.915**
	TSOPF	0.887	0.884	0.875	0.886	0.902	**0.925**	0.919
0.3	polbooks	0.841	0.760	0.700	0.856	0.883	**0.951**	0.935
	USAir97	0.851	0.831	0.757	0.848	0.874	**0.880**	**0.883**
	130bit	0.947	0.944	0.902	0.955	0.963	0.958	**0.966**
	bibd_12_5	0.912	0.885	0.808	0.888	0.881	0.910	**0.936**
	email-Eu-core	0.976	0.928	0.881	0.950	0.953	0.958	**0.984**
	TSOPF	0.908	0.906	0.895	0.883	0.926	0.918	**0.938**
0.4	polbooks	0.966	0.915	0.820	0.935	0.974	0.970	**0.989**
	USAir97	0.933	0.913	0.818	0.935	0.926	0.932	**0.953**
	130bit	0.966	0.962	0.938	0.971	0.971	0.976	**0.980**
	bibd_12_5	**0.997**	0.994	0.845	0.987	0.985	0.994	**0.998**
	email-Eu-core	**0.998**	0.970	0.951	0.983	0.994	0.985	**0.999**
	TSOPF	0.933	0.928	0.921	0.925	0.906	0.924	**0.963**
0.5	polbooks	0.998	**1**	0.894	0.988	0.996	**1**	**1**
	USAir97	0.982	0.971	0.914	0.982	0.980	0.981	**0.992**
	130bit	0.974	0.972	0.953	0.977	0.980	0.984	**0.989**
	bibd_12_5	**1**	0.998	0.894	0.999	0.998	0.996	**1**
	email-Eu-core	**1**	0.990	0.984	0.999	0.999	0.998	**1**
	TSOPF	0.966	0.955	0.954	0.935	0.951	0.939	**0.984**
	$+/-/\approx$	0/20/4	0/23/1	0/24/0	0/24/0	0/24/0	3/19/2	–

TABLE IV
THE COMPARISON OF SLEMO-NR AGAINST OTHER METHODS ON THE SIX REAL-WORLD DATASETS BASED ON THE RN MODEL IN TERMS OF AUC. THE BEST RESULT IN EACH ROW IS SHOWN IN BOLDFACE.

N_M	Dataset	CS	LASSO	OMP	SparseEA	MOEA/PSL	MAST-Net	SLEMO-NR
0.2	polbooks	0.766	0.744	0.629	0.744	0.737	0.763	**0.827**
	USAir97	0.775	0.777	0.684	0.789	0.784	0.767	**0.811**
	130bit	0.904	0.904	0.828	0.931	**0.932**	0.926	**0.935**
	bibd_12_5	0.800	0.804	0.767	0.813	0.816	0.816	**0.845**
	email-Eu-core	0.897	0.792	0.757	0.894	0.892	0.898	**0.924**
	TSOPF	0.874	0.861	0.860	0.876	0.902	0.873	**0.926**
0.3	polbooks	0.851	0.871	0.732	0.848	0.867	0.843	**0.899**
	USAir97	0.849	0.843	0.752	0.828	0.863	0.834	**0.874**
	130bit	0.947	0.945	0.901	**0.961**	0.963	0.954	**0.964**
	bibd_12_5	0.859	0.857	0.795	0.849	0.856	0.855	**0.909**
	email-Eu-core	0.978	0.915	0.888	0.973	0.974	0.958	**0.986**
	TSOPF	0.890	0.872	0.870	0.882	0.915	0.927	**0.941**
0.4	polbooks	0.949	0.945	0.839	0.948	0.927	0.946	**0.975**
	USAir97	0.925	0.924	0.803	0.908	0.933	0.906	**0.952**
	130bit	0.968	0.963	0.935	0.975	0.976	0.967	**0.982**
	bibd_12_5	0.999	0.998	0.832	0.94	0.933	0.954	**1**
	email-Eu-core	**0.998**	0.968	0.954	**0.998**	**0.998**	0.984	**0.999**
	TSOPF	0.906	0.888	0.883	0.907	0.925	0.934	**0.963**
0.5	polbooks	0.990	0.995	0.897	0.991	0.982	0.990	**1**
	USAir97	0.970	0.967	0.878	0.964	0.977	0.957	**0.987**
	130bit	0.975	0.972	0.953	0.979	0.980	0.981	**0.991**
	bibd_12_5	**1**	**1**	0.882	0.992	0.987	0.995	**1**
	email-Eu-core	**1**	0.988	0.986	**1**	0.999	0.993	**1**
	TSOPF	0.924	0.905	0.896	0.924	0.934	0.946	**0.972**
	$+/-/\approx$	0/21/3	0/23/1	0/24/0	0/21/3	0/22/2	0/24/0	–

TABLE V
RUNTIMES (IN SECOND) OF ALL COMPARED ALGORITHMS ON THE SIX REAL-WORLD DATASETS BASED ON THE RN MODEL WITH $N_M = 0.4$.

Dataset	CS	LASSO	OMP	SparseEA	MOEA/PSL	MAST-Net	SLEMO-NR
polbooks	1.90e-01	1.30e+01	**1.68e-01**	1.76e+01	2.23e+01	7.64e+00	1.67e+01
USAir97	**2.39e+00**	4.07e+02	9.81e+00	3.46e+02	3.87e+02	1.91e+02	3.05e+02
130bit	1.49e+01	1.43e+03	**6.01e+00**	1.73e+03	1.83e+03	6.53e+02	1.82e+03
bibd_12_5	4.41e+02	4.53e+02	**1.67e+01**	3.75e+03	3.89e+03	1.47e+03	5.74e+03
email-Eu-core	7.52e+01	6.64e+02	**4.06e+01**	8.99e+03	9.05e+03	2.31e+03	1.28e+04
TSOPF	2.76e+04	6.02e+04	**4.71e+03**	1.79e+05	1.54e+05	8.37e+04	2.09e+05

V. Conclusion

Due to the difficulty in setting an appropriate cut-off value and the curse of dimensionality in solving large-scale NRPs, the performances of existing methods are far from satisfactory. To overcome these shortcomings, we propose a subspace learning based evolutionary multiobjective network reconstruction algorithm, termed as SLEMO-NR. The core of SLEMO-NR is to use alternative representations in place of binary-coded individuals in the evolutionary process. The alternative representations not only carry the information of individuals but also can perform local search to find better solutions. Furthermore, we use LPCA to learn a subspace containing all the features of the network structure. Searching for optimal solutions in the subspace is much more effective than in the original space. The offspring solutions are generated in the subspace and mapped back to the original space. Afterward, a preference-based local search operator is proposed to focus on finding k-sparse solutions. The experimental results on synthetic networks show that the network structure subspace learning and preference-based strategy can significantly improve the performance of SLEMO-NR. Extensive experiments on six real-world networks validate the superiority of SLEMO-NR over six methods in solving large-scale network reconstruction problems.

This paper shows the necessity of network structure subspace learning in solving large-scale network reconstruction problems. Although the proposed method has a significant performance by using LPCA, it may become impractical when dealing with networks with millions of nodes. Therefore, further investigation of this problem is still desirable. Moreover, we are trying to find broad applications of SLEMO-NR in more challenging and practical problems.

References

[1] S. H. Strogatz, "Exploring complex networks," *Nature*, vol. 410, no. 6825, pp. 268–276, 2001.

[2] S. Hempel, A. Koseska, J. Kurths, and Z. Nikoloski, "Inner composition alignment for inferring directed networks from short time series," *Physical Review Letters*, vol. 107, no. 5, p. 054101, 2011.

[3] T. S. Gardner, D. di Bernardo, D. Lorenz, and J. J. Collins, "Inferring genetic networks and identifying compound mode of action via expression profiling," *Science*, vol. 301, no. 5629, pp. 102–105, 2003.

[4] K. Wu and J. Liu, "Learning large-scale fuzzy cognitive maps based on compressed sensing and application in reconstructing gene regulatory networks," *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 6, pp. 1546–1560, 2017.

[5] J. Liu, Y. Chi, and C. Zhu, "A dynamic multiagent genetic algorithm for gene regulatory network reconstruction based on fuzzy cognitive maps," *IEEE Transactions on Fuzzy Systems*, vol. 24, no. 2, pp. 419–431, 2016.

[6] W. Wang, R. Yang, Y. Lai, V. Kovanis, and C. Grebogi, "Predicting catastrophes in nonlinear dynamical systems by compressive sensing," *Physical Review Letters*, vol. 106, no. 15, p. 154101, 2011.

[7] A. Roebroeck, E. Formisano, and R. Goebel, "Mapping directed influence over the brain using granger causality and fMRI," *Neuroimage*, vol. 25, no. 1, pp. 230–242, 2005.

[8] G. Cimini, R. Mastrandrea, and T. Squartini, "Reconstructing networks," *arXiv preprint arXiv:2012.02677*, 2021.

[9] T. Squartini, G. Caldarelli, G. Cimini, A. Gabrielli, and D. Garlaschelli, "Reconstruction methods for networks: the case of economic and financial systems," *Physics Reports*, vol. 757, pp. 1–47, 2018.

[10] K. Wu, C. Wang, and J. Liu, "Evolutionary multitasking multilayer network reconstruction," *IEEE Transactions on Cybernetics*, 2021, doi: 10.1109/TCYB.2021.3090769.

[11] ——, "Multilayer nonlinear dynamical network reconstruction from streaming data," *SCIENTIA SINICA Technologica*, 2021, doi: 10.1360/SST-2020-0491.

[12] W. Wang, Y. Lai, and C. Grebogi, "Data based identification and prediction of nonlinear and complex dynamical systems," *Physics Reports*, vol. 644, pp. 1–76, 2016.

[13] G. Davis and S. Mallat, "Adaptive nonlinear approximations," Ph.D. dissertation, Graduate School of Arts and Science, New York University, 1994.

[14] K. Wu, J. Liu, X. Hao, P. Liu, and F. Shen, "Online reconstruction of complex networks from streaming data," *IEEE Transactions on Cybernetics*, 2020, doi: 10.1109/TCYB.2020.3027642.

[15] Y. Zhou, S. Kwong, H. Guo, X. Zhang, and Q. Zhang, "A two-phase evolutionary approach for compressive sensing reconstruction," *IEEE Transactions on Cybernetics*, vol. 47, no. 9, pp. 2651–2663, 2017.

[16] D. Napoletani and T. D. Sauer, "Reconstructing the topology of sparsely connected dynamical networks," *Physical Review E*, vol. 77, no. 2, p. 026103, 2008.

[17] X. Han, Z. Shen, W. Wang, and Z. Di, "Robust reconstruction of complex networks from sparse data," *Physical Review Letters*, vol. 114, no. 2, p. 028701, 2015.

[18] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B*, vol. 58, no. 1, pp. 267–288, 1996.

[19] W. Wang, Y. Lai, C. Grebogi, and J. Ye, "Network reconstruction based on evolutionary-game data via compressive sensing," *Physical Review X*, vol. 1, no. 2, p. 021021, 2011.

[20] D. L. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[21] Z. Shen, W. Wang, Y. Fan, Z. Di, and Y. Lai, "Reconstructing propagation networks with natural diversity and identifying hidden sources," *Nature Communications*, vol. 5, no. 1, pp. 1–10, 2014.

[22] K. Deb, *Multi-objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, 2001.

[23] K. Wu, J. Liu, and S. Wang, "Reconstructing networks from profit sequences in evolutionary games via a multiobjective optimization approach with lasso initialization," *Scientific Reports*, vol. 6, p. 37771, 2016.

[24] K. Wu, J. Liu, and D. Chen, "Network reconstruction based on time series via memetic algorithm," *Knowledge-Based Systems*, vol. 164, pp. 404–425, 2019.

[25] P. Moscato, "On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms," *Caltech Concurrent Computation Program, C3P Report*, vol. 826, p. 1989, 1989.

[26] D. Yu, M. Righero, and L. Kocarev, "Estimating topology of networks," *Physical Review Letters*, vol. 97, no. 18, p. 188701, 2006.

[27] X. Zhang, K. Zhou, H. Pan, L. Zhang, X. Zeng, and Y. Jin, "A network reduction-based multiobjective evolutionary algorithm for community detection in large-scale complex networks," *IEEE Transactions on Cybernetics*, 2018.

[28] J. Luo, L. Jiao, F. Liu, S. Yang, and W. Ma, "A Pareto-based sparse subspace learning framework," *IEEE Transactions on Cybernetics*, vol. 49, no. 11, pp. 3859–3872, 2018.

[29] F. Shen, J. Liu, and K. Wu, "A preference-based evolutionary biobjective approach for learning large-scale fuzzy cognitive maps: an application to gene regulatory network reconstruction," *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 6, pp. 1035–1049, 2020.

[30] M. Fazzolari, R. Alcala, Y. Nojima, H. Ishibuchi, and F. Herrera, "A review of the application of multiobjective evolutionary fuzzy systems: Current status and further directions," *IEEE Transactions on Fuzzy Systems*, vol. 21, no. 1, pp. 45–65, 2012.

[31] K. Wu, J. Liu, X. Hao, P. Liu, and F. Shen, "An evolutionary multiobjective framework for complex network reconstruction using community structure," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 2, pp. 247–261, 2021.

[32] L. M. Antonio and C. A. C. Coello, "Use of cooperative coevolution for solving large scale multiobjective optimization problems," in *Proceedings of the 2013 IEEE Congress on Evolutionary Computation*, 2013, pp. 2758–2765.

[33] A. J. Landgraf and Y. Lee, "Dimensionality reduction for binary data through the projection of natural parameters," *Journal of Multivariate Analysis*, vol. 180, p. 104668, 2020.

[34] M. A. Nowak and R. M. May, "Evolutionary games and spatial chaos," *Nature*, vol. 359, no. 6398, pp. 826–829, 1992.

[35] W. Wang and Y. Lai, "Abnormal cascading on complex networks," *Physical Review E*, vol. 80, no. 3, p. 036109, 2009.

[36] J. Hofbauer and K. Sigmund, "Evolutionary game dynamics," *Bulletin of the American Mathematical Society*, vol. 40, no. 4, pp. 479–519, 2003.

[37] G. Szabó and C. Tőke, "Evolutionary prisoner's dilemma game on a square lattice," *Physical Review E*, vol. 58, no. 1, p. 69, 1998.

[38] G. Szabó and G. Fath, "Evolutionary games on graphs," *Physics Reports*, vol. 446, no. 4-6, pp. 97–216, 2007.

[39] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[40] K. Pearson, "LIII. On lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.

[41] A. I. Schein, L. K. Saul, and L. H. Ungar, "A generalized linear model for principal component analysis of binary data," in *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, vol. 3, no. 9, 2003, p. 10.

[42] S. Datta and D. Nettleton, *Statistical Analysis of Next Generation Sequencing Data*. Springer, 2014.

[43] M. Yamamoto and K. Hayashi, "Clustering of multivariate binary data with dimension reduction via L1-regularized likelihood maximization," *Pattern Recognition*, vol. 48, no. 12, pp. 3959–3968, 2015.

[44] S. Geuens, K. Coussement, and K. W. De Bock, "A framework for configuring collaborative filtering-based recommendations derived from purchase data," *European Journal of Operational Research*, vol. 265, no. 1, pp. 208–218, 2018.

[45] D. R. Hunter and K. Lange, "A tutorial on MM algorithms," *The American Statistician*, vol. 58, no. 1, pp. 30–37, 2004.

[46] A. J. Landgraf and Y. Lee, "Generalized principal component analysis: Projection of saturated model parameters," *Technometrics*, pp. 1–14, 2019.

[47] L. Li, X. Yao, R. Stolkin, M. Gong, and S. He, "An evolutionary multiobjective approach to sparse reconstruction," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 6, pp. 827–845, 2013.

[48] H. Li, Q. Zhang, J. Deng, and Z. Xu, "A preference-based multiobjective evolutionary approach for sparse optimization," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 5, pp. 1716–1731, 2017.

[49] S. J. Wright, R. D. Nowak, and M. A. Figueiredo, "Sparse reconstruction by separable approximation," *IEEE Transactions on Signal Processing*, vol. 57, no. 7, pp. 2479–2493, 2009.

[50] M. A. Figueiredo, R. D. Nowak, and S. J. Wright, "Gradient projection for sparse reconstruction: application to compressed sensing and other inverse problems," *IEEE Journal of Selected Topics in Signal Processing*, vol. 1, no. 4, pp. 586–597, 2007.

[51] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006.

[52] B. W. Matthews, "Comparison of the predicted and observed secondary structure of t4 phage lysozyme," *Biochimica et Biophysica Acta (BBA)-Protein Structure*, vol. 405, no. 2, pp. 442–451, 1975.

[53] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Transactions on Information Theory*, vol. 53, no. 12, pp. 4655–4666, 2007.

[54] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[55] P. Erdős and A. Rényi, "On random graphs," *Publicationes Mathematicae Debrecen*, vol. 6, pp. 290–297, 1959.

[56] A. L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999.

[57] M. E. Newman and D. J. Watts, "Renormalization group analysis of the small-world network model," *Physics Letters A*, vol. 263, no. 4-6, pp. 341–346, 1999.

[58] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[59] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[60] Y. Tian, C. Lu, X. Zhang, K. C. Tan, and Y. Jin, "Solving large-scale multiobjective optimization problems with sparse optimal solutions via unsupervised neural networks," *IEEE Transactions on Cybernetics*, vol. 51, no. 6, pp. 3115–3128, 2021.

[61] R. Rossi and N. Ahmed, "The network data repository with interactive graph analytics and visualization," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, vol. 29, no. 1, 2015.

[62] H. Yin, A. R. Benson, J. Leskovec, and D. F. Gleich, "Local higher-order graph clustering," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 555–564.

[63] Y. Tian, X. Zhang, C. Wang, and Y. Jin, "An evolutionary algorithm for large-scale sparse multiobjective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 2, pp. 380–393, 2019.

[64] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "PlatEMO: A MATLAB platform for evolutionary multi-objective optimization [educational forum]," *IEEE Computational Intelligence Magazine*, vol. 12, no. 4, pp. 73–87, 2017.

Chaolong Ying received the B.S. degree in electronic information engineering in 2019 from Xidian University, Xi'an, China, where he is currently working toward the M.Sc. degree with the School of Artificial Intelligence, Xidian University, China. His current research interests include complex networks, evolutionary computation, and graph neural networks.

Jing Liu (SM'15) received the B.S. degree in computer science and technology and the Ph.D. degree in circuits and systems from Xidian University in 2000 and 2004, respectively.

In 2005, she joined Xidian University as a lecture, and was promoted to a full professor in 2009. From Apr. 2007 to Apr. 2008, she worked at The University of Queensland, Australia as a postdoctoral research fellow, and from Jul. 2009 to Jul. 2011, she worked at The University of New South Wales at the Australian Defence Force Academy as a research associate. Now, she is a full professor in the Guangzhou Institute of Technology, Xidian University. Her research interests include evolutionary computation, complex networks, fuzzy cognitive maps, multiagent systems, and data mining.

Dr. Liu is the associate editor of IEEE Trans. Evolutionary Computation. She has been the chair of Emerging Technologies Technical Committee (ETTC) of IEEE Computational Intelligence Society from 2017-2018.

Kai Wu received the B.S. degree in intelligent science and technology and the Ph.D. degree in circuits and systems from Xidian University, Xi'an, China, in 2015 and 2020, respectively. He is currently an Associate Professor with the School of Artificial Intelligence, Xidian University. His current research interests include fuzzy cognitive maps, evolutionary computation, complex networks, and data mining.

Chao Wang received the B.S. degree in intelligent science and technology in 2019 from Xidian University, Xi'an, China, where he is currently working toward the Ph.D. degree with the School of Artificial Intelligence, Xidian University, China. His current research interests include multi-task learning and optimization, evolutionary computation, and complex networks.