



## Navigating differential structures in complex networks

Leonardo L. Portes <sup>1,\*</sup> and Michael Small <sup>1,2</sup>

<sup>1</sup>*Complex Systems Group, Department of Mathematics and Statistics, University of Western Australia, Nedlands, Perth, WA 6009, Australia*

<sup>2</sup>*Mineral Resources, CSIRO, Kensington, Perth, WA 6151, Australia*



(Received 23 August 2020; revised 9 November 2020; accepted 20 November 2020; published 7 December 2020)

Structural changes in a network representation of a system, due to different experimental conditions, different connectivity across layers, or to its time evolution, can provide insight on its organization, function, and on how it responds to external perturbations. The deeper understanding of how gene networks cope with diseases and treatments is maybe the most incisive demonstration of the gains obtained through this *differential* network analysis point of view, which led to an explosion of new numeric techniques in the last decade. However, *where* to focus one's attention, or how to *navigate* through the differential structures in the context of large networks, can be overwhelming even for a few experimental conditions. In this paper, we propose a theory and a methodological implementation for the characterization of shared “structural roles” of nodes *simultaneously* within and between networks. Inspired by recent methodological advances in chaotic phase synchronization analysis, we show how the information about the shared structures of a set of networks can be *split and organized* in an automatic fashion, in scenarios with very different (i) community sizes, (ii) total number of communities, and (iii) even for a large number of 100 networks compared using numerical benchmarks generated by a stochastic block model. Then, we investigate how the network size, number of networks, and mean size of communities influence the method performance in a series of Monte Carlo experiments. To illustrate its potential use in a more challenging scenario with real-world data, we show evidence that the method can still split and organize the structural information of a set of four gene coexpression networks obtained from two cell types  $\times$  two treatments (interferon- $\beta$  stimulated or control). Aside from its potential use as for automatic feature extraction and preprocessing tool, we discuss that another strength of the method is its “story-telling”-like characterization of the information encoded in a set of networks, which can be used to pinpoint unexpected shared structure, leading to further investigations and providing new insights. Finally, the method is flexible to address different research-field-specific questions, by *not* restricting what scientific-meaningful characteristic (or relevant feature) of a node shall be used.

DOI: [10.1103/PhysRevE.102.062301](https://doi.org/10.1103/PhysRevE.102.062301)

### I. INTRODUCTION

Modeling a complex system through a network representation has been consolidated as a powerful abstraction and data-driven paradigm for the science of complexity [1]. A recent step further is the comparative analysis of structures within a *set* of network representations, for example, examining the community structures that change (or on the other hand, remain coherent) along with different experimental conditions [2], layers [3], or time points [4], which can bring relevant information about the structure and function of the system's interacting parts.

Maybe the most known and prolific aspect of that line of inquiry is *differential* network analysis, which has been mainly developed by the bioinformatics community in the context of gene expression analysis (see [5] for a review). That approach has contributed to several discoveries related to how genes communicate to support the emergence of physiological responses of an organism in different disease states [6–9]. However, aside from the existence of several numeric techniques to accomplish the task, *where* to focus ones attention

on, or how to *navigate* through, the differential structures in the context of large networks, can be overwhelming even along with few experimental conditions (i.e., a small set with few networks).

In this paper, we provide a method to automatically unfold that information, not only by identifying similar structures, but by *splitting that outcome in a highly organized way* even for large sets of networks. Hence, allowing one to pinpoint *how* and *what* communities' structures change (or remain the same) along with different networks.

Our approach is inspired by recent advances in the field of chaotic phase synchronization (PS) and through regarding its characterization and detection via multivariate singular spectrum analysis (M-SSA). It has been shown that orthogonal rotations of the eigenvectors, obtained from concatenated trajectory matrices, provide clear and almost automatic identification of oscillatory modes that are being shared by the coupled chaotic oscillators [10–13]. Specifically, the final outcome in that version of M-SSA is a set of rotated eigenvectors that clearly encode the *shared oscillatory components* of those oscillators in a highly organized and informative way, which can be used for further detection of PS and characterization of phase synchronized clusters.

\*ll.portes@gmail.com

Here, we make a parallel between shared oscillatory modes in PS analysis and the shared “structural role” of nodes to analyze the “synchronized” structures in a set of networks. In short, our method takes advantage of (i) a varimax rotation to simplify the structure of the eigenvectors obtained from (ii) the eigendecomposition of (iii) a concatenated adjacency matrix that represents the network in different conditions. For the sake of discussion, we will call it the concatenate-decompose-rotate (CDR) approach. In the new context of network analysis, we show that the outcome is a highly *interpretable map* of the nuances of the network in those different conditions.

It is worth noting that some combination of those three steps (concatenate, decompose, rotate) has already been applied in other works and by different scientific communities [14,15]. However, both their main goals and outcomes are different from the CDR as articulated here. Indeed, we expect that the fundamental underlying ideas of the CDR could be applied over and above the other methods that have been developed within different scientific communities, or even be used as a bridge between them to nurture a deeper theoretical understanding, for example, as the recently demonstrated equivalence of modularity maximization and the method of maximum likelihood [16]).

Nevertheless, there has been some criticism regarding the methodological aspects of community detection in network science [17]. In particular, we share the pertinent view that a “method based on a mere hunch that something might work is inherently less trustworthy than one based on a provable result or fundamental mathematical insight” [18]. The CDR approach can be described in a very simple and direct way, just by showing that a varimax rotation of eigenvectors obtained from a “generic” spectral algorithm works in some specific scenarios. However, we start this paper, and devote a large part of it, to providing a simple theory for *how* and *why* the CDR works. Aiming at a broader audience from different fields, and trying to use the most simple and transparent concepts as possible, this is done from a factor analysis point of view illustrated by a minimalistic toy scenario. Both the theoretical aspect and the scenarios explored here do not represent a complete work, but the exploration for the feasibility of a CDR method to “navigate” through differential structures in complex networks.

This paper is organized as follows. The mathematical theory, and illustrative motivational toy scenario, for differential network analysis with the CDR are presented in Sec. II. Then, we explore the method with larger networks and in a larger number of conditions in Sec. III. First, CDR is applied on synthetic networks with  $N = 2000$  nodes in  $H = 100$  different conditions, with a random number of communities (between 10 and 20) of random sizes (10 to 100 nodes) distributed in those conditions. The scope here is on networks of nonoverlapping clusters (or communities), generated from the stochastic block model (SBM). The clusters can be present or absent in different conditions. The method’s accuracy is investigated through Monte Carlo simulations, where we manipulated the inner probabilities (how strongly the nodes within a community are connected) and the outer probability (how strongly nodes from different communities, and the “noisy background,” are connected). That is followed by the

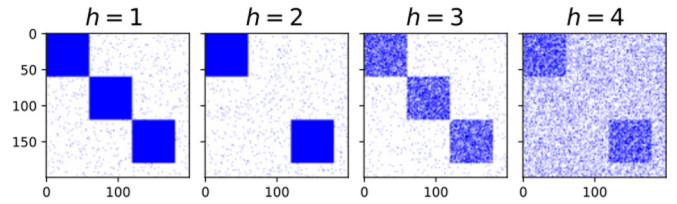


FIG. 1. Toy scenario of a small network in  $H = 4$  different conditions. It has  $N = 200$  nodes, which could be members of  $R = 3$  communities. This scenario can be thought of as a system in  $H$  different experimental conditions, or an evolving network in  $H$  different time points.

investigation of how the network size, number of networks, and the mean size of communities influence the method performance. Finally, we show evidence that the method can still split and organize the structural information in a much richer and complex scenario, with a real-world data set composed of four gene coexpression networks obtained from two cell types  $\times$  two treatments (interferon- $\beta$  stimulated or control). Concluding remarks are made in Sec. IV.

## II. METHODS

In this section, we introduce the simple theory and basic methodology from a factor analysis (FA) point of view. The material is introduced by blending together a brief review of some FA results [19,20] with the pursuit of a theory for differential network analysis and community structure characterization. This means that we will start by being as abstract as necessary and that references to a “toy scenario” (Sec. II A) will be used as a motivation and to illustrate the main aspects of the proposed framework. Some small conceptual differences from actual factor analysis will be discussed when necessary.

### A. Motivation: Toy scenario

Consider the scenario of an undirected and unweighted network with  $N = 200$  nodes, in  $H = 4$  different conditions in respect to its connections, as shown in Fig. 1. Assume that nodes neither disappear nor are created, only connections may change. For each condition  $h = 1, \dots, H$ , the network is represented by its respective adjacency matrix  $\mathbf{A}_h$  of size  $N \times N$ , with elements  $a_{ij} = 1$  if nodes  $i$  and  $j$  are connected, but  $a_{ij} = 0$  otherwise. Hence, the set  $\{\mathbf{A}_h\}_{h=1}^H = \{\mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3, \mathbf{A}_4\}$  of adjacency matrices represents the structure of the network across the different  $H$  conditions. There are  $R = 3$  highly connected and nonoverlapping communities of sizes  $N_r = 60$  nodes each, as seen at Fig. 1, labeled as  $r = 1, 2, 3$ .

The set  $\{\mathbf{A}_h\}_{h=1}^H$  was generated by a stochastic block model [21] with the package NETWORKX [22]. The connection probabilities between nodes within the same community (inner probability) is  $p$ , being  $q$  otherwise (i.e., the outer probability). Setting these probabilities as  $0 \leq q < p \leq 1$  allows one to generate the  $R$  highly connected communities, which can be visualized as blocks in Fig. 1. For the rest of the network structure, we considered a “ghost” community of size  $N - \sum_r N_r = 20$  nodes, with both the within and between probabilities equal to  $q$ . Let the adjacency matrices  $\mathcal{A}_1, \mathcal{A}_2,$

and  $\mathcal{A}_3$  represent each of these communities. Accordingly, the scenario depicted here can be seen (conceptually) as  $R$  Erdős and Rényi [23] random networks  $G_{N,r,p}$  “planted” on a  $G_{N,q}$ . Because of this, we will often refer to  $G_{N,r,p}$  as the “random background.”

By design, the toy scenario illustrates two aspects regarding the structural changes in the network. The first one refers to the presence or absence of a community in a given condition. Specifically, communities  $r = 1$  and  $r = 3$  are present at all  $H$  conditions, while  $r = 2$  is absent at conditions  $h = 2$  and 4. In this aspect, the structure of the network is the *same* in (i) conditions 1 and 3, as well as in (ii) conditions 2 and 4, but (iii) different otherwise. That is the main structural change that we want to identify.

However, the *internal* structure of conditions (and background noise) may be different, as follows. The second aspect refers to how strong a community is, as compared to the connections of its members to the other nodes (i.e., nodes from the other communities, as well as from nodes from the random background). This is done by using different combinations of the probabilities  $p$  and  $q$ . The values for conditions 1 and 2 are  $(p, q) = (1, 0.02)$ , therefore, the communities are fully connected (known as 1-cliques). The communities became much more weakly connected in conditions 3 and 4 by setting  $p = 0.6$ , while the background noise becomes stronger by a factor of 10 at condition 4 with  $q = 0.2$ . So, the *mixing* between communities and background noise is larger at condition 3, and much larger at condition 4. Later, we will investigate how the mixing level interferes on the CDR method.

*Remark 1.* We call attention to one consequence of the SBM, that could otherwise pass unnoticed. For example, the *internal structure* of the community  $r = 1$  can be completely different between conditions 3 and 4. They just share the same inner probability  $p = 0.6$ , but the actual connections of their nodes are set at random by the model. The same occurs for the random background in conditions 1 and 3: the inner and outer probabilities  $p = q = 0.02$  are the same, but the actual connections are not.

**B. Theoretical framework**

The initial assumptions of the method are as follows. There exists an abstract *property*  $y_i, i = 1, \dots, HN$ , for the  $N$  nodes in the  $H$  conditions. For example, this means that for a given node  $j \in [1, N]$ , the properties  $y_{j+(h_1-1)N}$  and  $y_{j+(h_2-1)N}$  will refer to the same node  $j$  in two different conditions  $h_1 \neq h_2 \in [1, H]$ . Conceptually, we will assume that a causal relationship exists between the set  $\{y_i\}_{i=1}^{HN}$  and the sets of unknown and abstract processes  $\{\xi_i\}_{i=1}^v$  and  $\{e_i\}_{i=1}^{HN}$  (i.e., the factors or latent variables). For now, we just assume that  $y, \xi$ , and  $e$  can be represented as vectors in an abstract vector space, mainly because we will need the concept of inner products  $\langle \bullet | \bullet \rangle$  to represent the extent to which they are (or they are not) close. In the set  $\{e_i\}$  are the *factors* specific for each  $y_i$  (i.e., the *unique factors*). The set  $\{\xi_i\}$  are factors that can influence any and several  $y_i$  (i.e., the *common factors*). We assume that the common and unique factors are independent,  $\langle \xi_i | e_j \rangle = 0$ , and that the unique factors are orthogonal,  $\langle e_i | e_j \rangle = 0$  if  $i \neq j$  (orthogonality of the common factors will not be assumed

yet). By defining the column matrices  $\mathcal{Y} = [y_1 \dots y_{HN}]^T$ ,  $\mathcal{X} = [\xi_1 \dots \xi_v]^T$ , and  $\mathcal{E} = [e_1 \dots e_{HN}]^T$ , one can write the linear model

$$\mathcal{Y} = \mathbf{\Lambda} \mathcal{X} + \mathbf{\Psi} \mathcal{E}, \tag{1}$$

known as the *fundamental equation* of factor analysis [20]. The matrices  $\mathbf{\Lambda}$  (size  $N \times v$ ) and  $\mathbf{\Psi}$  (size  $HN \times HN$ ) provide the common and unique factor loadings (or weights), respectively. Matrix  $\mathbf{\Psi}$  is diagonal (i.e., off-diagonal elements are equal to zero) because the factors  $e_i$  are unique. Without loss of generality, we assume the factors’ norm is equal to one because they can be absorbed by the factor weight matrices  $\mathbf{\Lambda}$  and  $\mathbf{\Psi}$ . So,  $\langle e_i | e_j \rangle = \delta_{i,j}$  and  $\langle \xi_i | \xi_i \rangle = 1$  ( $\delta$  is the Kronecker’s delta function).

Finally, we assume that the goal of modeling a given phenomenon (with data from a set of observations) through network theory is to investigate the *differential* clustering of nodes: what structures remain the same, and what changes, along with the different conditions  $H$ . Now, we explore two consequences of model (1) under those assumptions.

**1. Feasibility for differential network analysis and clustering**

The characterization of the clustering of nodes along conditions due to the sharing of latent variables  $\xi_i$  could be achieved by inspecting the structure of the matrix  $\mathbf{\Lambda}$ . To see this, consider the product between  $y_i$  and  $\xi_j$ :  $\mathbf{R}_{\mathcal{Y}\mathcal{X}} \doteq \mathcal{Y} \mathcal{X}^T$ , with elements  $[\mathbf{R}_{\mathcal{Y}\mathcal{X}}]_{i,j} = \langle \xi_i | y_j \rangle$ . Because  $\langle \xi_i | e_j \rangle = 0$ , we have

$$\mathbf{R}_{\mathcal{Y}\mathcal{X}} \doteq \begin{bmatrix} \langle \xi_1 | y_1 \rangle & \dots & \langle \xi_v | y_1 \rangle \\ \vdots & \ddots & \vdots \\ \langle \xi_1 | y_{HN} \rangle & \dots & \langle \xi_v | y_{HN} \rangle \end{bmatrix} \equiv \mathbf{\Lambda} \mathbf{R}_{\mathcal{X}\mathcal{X}}, \tag{2}$$

where  $\mathbf{R}_{\mathcal{X}\mathcal{X}} \doteq \mathcal{X} \mathcal{X}^T$ .

Expression (2) can be simplified even further if we are allowed to assume orthogonality between the common factors  $\langle \xi_i | \xi_j \rangle = \delta_{i,j}$ . Under that new assumption, for which henceforth we restrict the scope of this paper, (2) becomes

$$\mathbf{R}_{\mathcal{Y}\mathcal{X}} = \mathbf{\Lambda}. \tag{3}$$

To gather insights of the implications of (2) and (3), we use (1) to frame the problem of differential network analysis and community characterization of the toy scenario shown in Fig. 1. In that context, and because we now know the real community structures across conditions (i.e., the ground truth), a reasonable hypothesis is that the “true” underlying community structure to be captured by (1) is given by the  $R$  known planted communities only, and not by the background noise from  $G_{N,q}$ . Then, by design the underlying theoretical assumptions for the FA model (1) would be (i)  $\xi_1$  is the “cause” of the community structure of nodes 1 to 60 (i.e.,  $r = 1$ ) at all  $H = 4$  conditions; (ii)  $\xi_2$  for nodes 61 to 120, but *only* at conditions  $h = 1, 2$ . (iii)  $\xi_3$  for nodes 121 to 180 at all  $H = 4$  conditions. Note that the labels 1, 2, and 3 were used here for the sake of illustration (e.g., community  $r = 1$  could be related to  $\xi_3$  and so on). Because of that, expressions (2) and (3) tell us that these causal relationships should be reflected on the structure of the common factor loading matrix  $\mathbf{\Lambda}$  as high loadings related to those three factors  $\xi$  for the properties  $y_i$

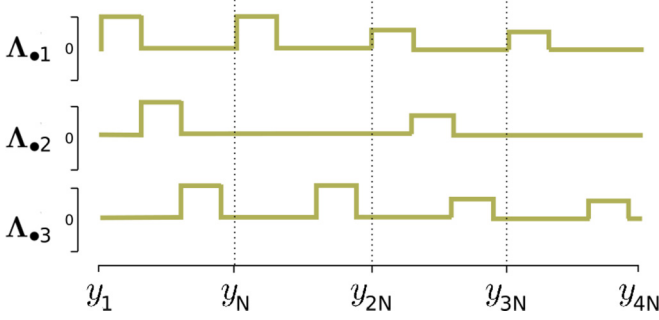


FIG. 2. Schematic picture of the intuitive expected structure of matrix  $\Lambda$  in the context of the toy scenario of Fig. 1. The leading three columns  $\Lambda_{\bullet k} = [\langle \xi_k | y_1 \rangle \dots \langle \xi_k | y_{HN} \rangle]^\top$ , with  $k = 1, 2, 3$ , are shown.

within these ranges. That is schematically shown in Fig. 2, where  $\Lambda_{\bullet k} = [\langle \xi_k | y_1 \rangle \dots \langle \xi_k | y_{HN} \rangle]^\top$ .

The main result of this paper is based upon finding (or extracting) *that* special structure from measured data because it clearly reports on the structural changes of the network. In a sense, this provide us with a map, or a book composed by  $H$  “chapters” within the  $H$  segments of length  $N$ , that allows one to navigate through the network differential structure history. Therefore, our aim now is to obtain that idealized structure after fitting *observed data* on a model based on (1), and then both (i) community structure characterization and (ii) differential network analysis would be straightforward.

There is a subtle conceptual difference between model (1) and an actual factor analytical model. In the latter,  $y$  refers to *measured data*. However, in this paper we are considering  $y$  just as an *abstract concept* within a vector space, which will provide us with flexibility latter. Accordingly, no assumptions will be made regarding particular statistics of  $y$  (e.g., a random variable with zero mean and unity variance). As well, we will make use of the concept of a Gramian matrix, instead of the covariance or the correlation matrices, for the matrices related to the inner products. Actual measured data will be inserted into the framework in the next section, when we show one way to extract the structure  $\Lambda$ .

## 2. Extracting the structure of $\Lambda$

Consider the projection of  $y_i$  onto itself,  $\mathbf{R}_{yy} \doteq \mathcal{Y}\mathcal{Y}^\top$  (with matrix elements  $[\mathbf{R}_{yy}]_{i,j} = \langle y_i | y_j \rangle$ ). Because the common and unique factors are orthogonal, one can write  $\mathbf{R}_{yy} = \Lambda^2 \mathbf{R}_{\chi\chi} + \Psi^2$ , an expression that is often known as the *fundamental theorem* of factor analysis [20]. Rearranging for  $\Lambda^2$ , and because one previously assumed the orthonormality between the common factors, we have

$$\Lambda^2 = \mathbf{R}_{yy} - \Psi^2. \quad (4)$$

We will simplify (5) even further by assuming that the term  $\Psi^2$  is negligible, so

$$\Lambda^2 = \mathbf{R}_{yy}. \quad (5)$$

That is an assumption very often used in FA. If there exists a way to estimate the contributions from the unique factors, one can go back and simply update the diagonal elements of  $\Lambda^2$  (because  $\Psi^2$  is a diagonal matrix). Finally, writing the

eigendecomposition  $\mathbf{R}_{yy} = \mathbf{V}\Sigma\mathbf{V}^\top$ , we have

$$\Lambda = \Sigma^{1/2}\mathbf{V}. \quad (6)$$

The eigenvectors  $\mathbf{v}_k$  correspond to the columns of matrix  $\mathbf{V}$ , with respective eigenvalues  $\sigma_k$ ,  $k = 1, 2, \dots, n$ , in the main diagonal of matrix  $\Sigma$ . We assume they are in the decreasing order  $\sigma_1 > \sigma_2 > \dots > \sigma_n$ .

*Remark 2.* While we will be using sequential indexing to facilitate the interpretability of the figures, no aspect of the CDR is influenced by it. Specifically, by shuffling the nodes’ indices of the  $H$  adjacency matrices through a random permutation  $P_\pi$ , the new factor loading matrix is simply  $\hat{\Lambda} = P_\pi \Sigma^{1/2} \mathbf{V} \equiv P_\pi \Lambda$ . To make this point clear for an audience with diverse backgrounds, that equivalence (i) is illustrated in the simulated examples, and (ii) the mathematical proof is given in the Appendix A.

That is one of the several procedures for extracting the structure of the factor loading matrix. It is sometimes called *principal component factor analysis* [19], or referred to as extraction through *principal components analysis* (PCA) [20]. However, that is the case when  $\mathbf{R}_{yy}$  comes from *measured data* (equivalently,  $y_i$  is a random vector). So, now we need to address the aforementioned conceptual different between model (1) and an actual factor analytical model:  $y_i$  are concepts, not random variables.

*What does it mean that two nodes belong to the same community?* The answer can (and should) depend on the actual research question and the field-dependent characteristics that one aims at by grouping the nodes and asking for their differential network structure. Here, letting  $y_i$  be concepts and not actual data, we aim at that flexibility for the CDR framework to address different field-specific points of view. This can be put more clearly through the following example, which will be used as well to establish remaining procedures.

Consider again the toy scenario of Fig. 1, and assume that the measured data is the set  $\{\mathbf{A}_h\}_{h=1}^H$ . For the task at hand (differential network analysis) and the characteristics of the community structures (high within connected blocks, low between connectivity), one reasonable choice is to use the similarity of the list of neighbors between nodes and conditions to capture the relevant question one wants to address through a differential network analysis. Given two nodes  $i$  and  $j$ , their list of neighbors in conditions  $h_1$  and  $h_2$  are the columns  $[\mathbf{A}_{h_1}]_{\bullet i}$  and  $[\mathbf{A}_{h_2}]_{\bullet j}$  of the respective adjacency matrices. Let  $\mathbf{X} = [\mathbf{A}_1 \mathbf{A}_2, \dots, \mathbf{A}_H]$  be the  $N \times HN$  matrix formed by horizontally concatenating the  $H$  adjacency matrices (see Fig. 3, top panel). Accordingly, one defines the estimate  $\hat{\mathbf{R}}_{yy}$  for  $\mathbf{R}_{yy}$  as

$$\mathbf{R}_{yy} \doteq \mathcal{Y}\mathcal{Y}^\top \triangleq \hat{\mathbf{R}}_{yy} \triangleq \mathbf{X}^\top \mathbf{X}, \quad (7)$$

where the symbol  $\triangleq$  stands for “estimated from.” Here, we are using the symbol  $\triangleq$  to emphasize that this definition depends on the field-specific characteristics that are pertinent for the question one wants to answer. Henceforth in this paper, we will make use of (7) to estimate the common factor loadings  $\hat{\Lambda}$  through the eigendecomposition of  $\mathbf{X}^\top \mathbf{X}$ .

The leading five columns of the estimated loadings,  $\hat{\Lambda}_{\bullet k}$  for  $k = 1, \dots, 5$ , are shown in Fig 3 (middle row). Henceforth,



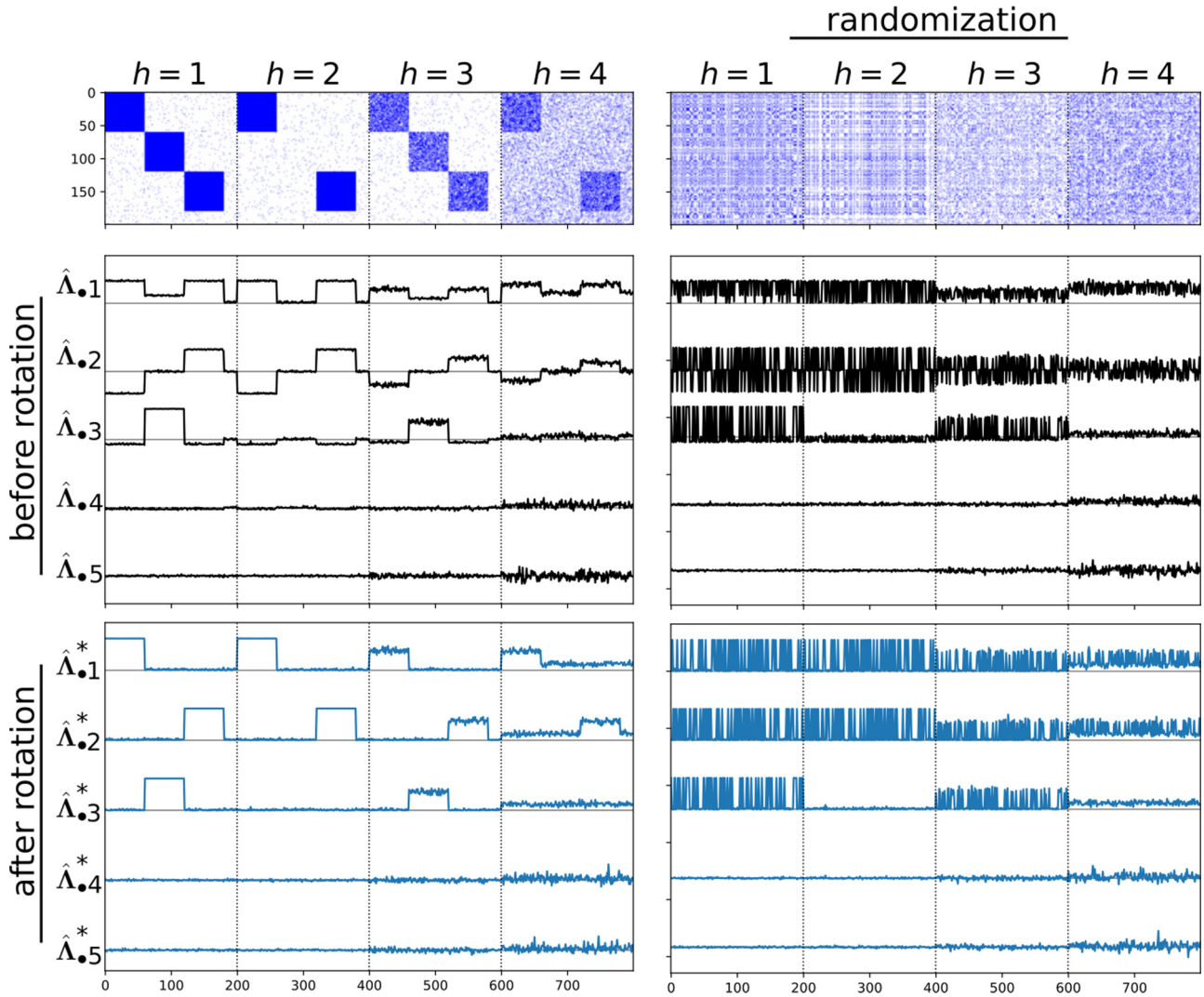


FIG. 3. Results of the CDR on the toy scenario of Fig. 1. The concatenated matrix  $\mathbf{X}$  is shown on top. The structure of matrix  $\mathbf{\Lambda}$  is shown before (middle) and after (bottom) the varimax rotation. Before rotation, the leading loadings  $\hat{\Lambda}_{\bullet k}$  contain mixed information regarding the  $R = 3$  communities. After the varimax rotation (bottom row), the rotated loadings  $\hat{\Lambda}_{\bullet k}^*$  are unmixed and bring more detailed information about the communities and on how they change between the  $H = 3$  conditions, a structure that clearly mirrors the idealized one of Fig. 2.

we will call them simply “loadings.” They contrast deeply with the desired “simple” structure previously shown in Fig. 2. Specifically, one sees a mixed signature of communities, which is more entangled in the leading two loadings. Actually, that is indeed the expected *intermediate* result from FA. The reason is that we have applied PCA to extract the loadings and PCA, by itself, is the solution for the maximization problem  $\max \text{tr} \text{cov}(X)$  given the restriction of orthonormality of the principal directions  $\langle \mathbf{v}_i | \mathbf{v}_j \rangle = \delta_{i,j}$ . So, PCA maximizes the variance explained by the leading  $k$  components, and a large amount of the information becomes entangled in the leading eigenvectors  $\mathbf{v}_i$  and, consequently, in the leading  $\hat{\Lambda}_{\bullet k}$ .

The usual followup procedure in FA is based on Thurstone’s concept of simple structure [24]: the rotation of the factor loadings by a given criterion that maximizes the *simplicity* of  $\hat{\Lambda}$ , and so (hopefully) enhancing the interpretability of that matrix.

### 3. Varimax rotation and the simple structure of $\mathbf{\Lambda}$

Kaiser’s varimax [25] is considered the most-efficient (orthogonal) rotation in FA [26], and the most often applied. Let the elements of the factor loading matrix be  $\hat{\Lambda} = [\lambda_{k,d}]$ . The varimax rotation aims at finding the orthogonal rotation  $\hat{\Lambda}^* = \hat{\Lambda} \mathbf{T}$  that satisfies the varimax criterion (VC)

$$\text{VC}(\mathbf{\Lambda}) = \sum_{k=1}^S \left[ \frac{1}{D} \sum_{d=1}^D \lambda_{dk}^4 - \left( \frac{1}{D} \sum_{d=1}^D \lambda_{dk}^2 \right)^2 \right]. \quad (8)$$

Specifically, (8) is the *raw* varimax criterion. It represents the maximization of the variance across the columns of the squared factor loadings matrix. The summation is over the first  $S$  factors  $\hat{\Lambda}_{\bullet k}$ ,  $k = 1, \dots, S$ .

The result of that rotation is shown in Fig. 3(bottom), with  $S = 20$ . Each leading  $\hat{\Lambda}_{\bullet k}^*$  carries now a unique fingerprint of (i) each community for (ii) each condition, similar to the

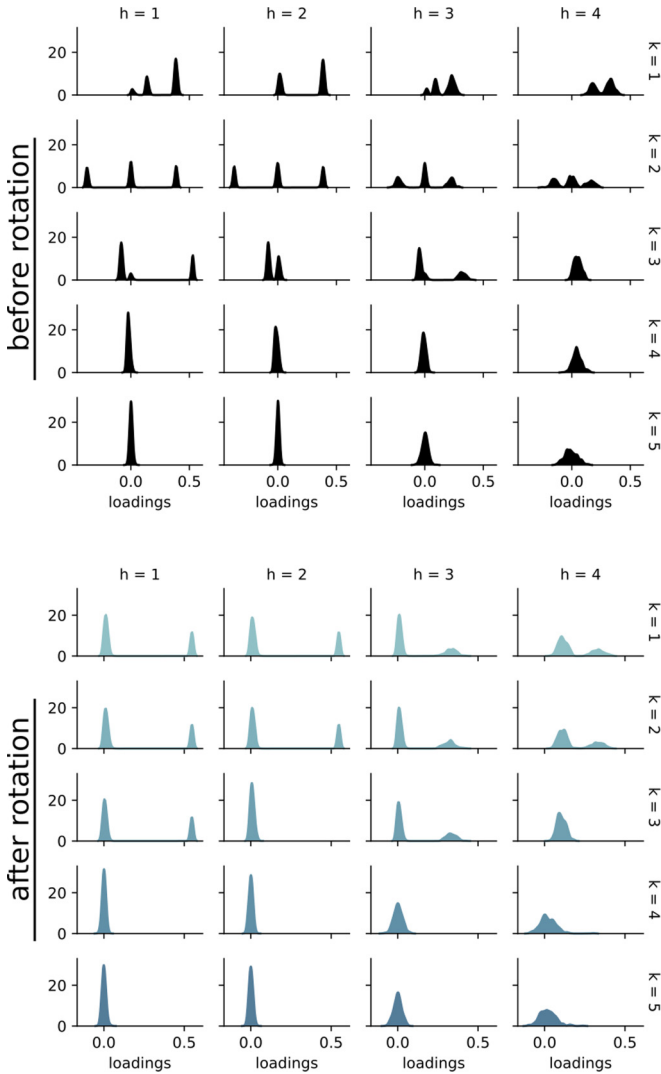


FIG. 4. Estimated Gaussian kernel densities for each segment  $h$  for the leading five rotated factor loadings  $\hat{\mathbf{A}}_{\bullet k}^*$ , corresponding to Fig. 3 (bottom). Darker shades refer to larger values of  $k$ .

expected idealized structure in Fig. 2. The main result of this paper is that that approach recovers the full “story” of the network, where each “chapter” is encoded on each  $H = 3$  segment of length  $N$ . Then, it becomes straightforward to read: communities 1 and 2 were present along with all  $H = 3$  conditions, while community 2 disappeared in condition  $h = 2$  but reappeared in  $h = 3$ . Another information is provided by the different magnitude of the pulselike pumps at different segments of the same  $\hat{\mathbf{A}}_{\bullet k}^*$ . For example, consider the  $\hat{\mathbf{A}}_{\bullet 1}^*$ . The first two pumps have the same magnitude, which is larger than the magnitude of the last two segments. This means that, aside from the community  $r = 1$  being present in all  $H = 4$  conditions, something in its structure is more similar within conditions  $h \in \{1, 2\}$  and  $h \in \{3, 4\}$  than between them. By design, we know that this should be a consequence of the different inner probabilities:  $p = 1$  for  $h \in \{1, 2\}$  and  $p = 0.6$  for  $h \in \{3, 4\}$ .

Another point of view is provided by Fig. 4, which shows the distribution of weights for all  $\hat{\mathbf{A}}_{\bullet k}^*$  segments (before and

after rotation). They correspond to Fig. 3, and are equivalent for both the sequential indexing and randomized one.

Before rotation, there are two problematic aspects that could compromise the use of post-processing tools to identify the communities  $r$  that share the same condition  $h$ . First, several segments contain mixed information from different communities. Second, some distributions are extremely close [e.g.,  $(k, h) = (1, 3)$  and  $(3, 1)$ ]. That can occur even for a segment containing information of only two distributions [e.g.,  $(k, h) = (3, 2)$ ]. That distance could be so close (as compared to the others) that it would be necessary to isolate those cases and look for a specific fine tuning to identify the indices corresponding to each cluster.

On the contrary, both problematic aspects were rectified by the rotation. Each segment contains information about different communities, and the distance between distributions increased. In Sec. II C, we will show how that greatly simplifies the identification of the communities  $r$  that share the same condition  $h$ , by allowing one to use each segment within  $\hat{\mathbf{A}}_{\bullet k}^*$  as *binary* classifiers.

As a remark, note that because of the stochastic block model applied, the structure of a given community is not identical in different conditions (when  $p < 1$ , i.e., communities are *not* 1-cliques), nor are the connections of its nodes with the outer nodes (for any value of  $p$ ). Even so, the proposed approach shows a clear representation of the shared community structure, by splitting and organizing that information in different factor loading vectors  $\hat{\mathbf{A}}_{\bullet k}^*$ .

That outcome shows that the number of rotated vectors  $S$  should be larger than the number of features to be “split and organized.” This allows for sufficient degrees of freedom for the rotation to accommodate those features and the “noise.” This does not imply the necessity of *a priori* knowledge of the number of communities, nor on how they are shared in different networks. On the contrary, one can overshoot as we did by using  $S = 20$  (using all vectors would imply a higher computational demand, which could be prohibitive in some scenarios). A complementary strategy is comparing the results for two different values of  $S$ : if they are different,  $S$  should be increased.

There is another feature in Fig. 3 worthy of comment. A constant trend (vertical shift) in the leading loadings (both before and after rotation) is clearly seen when the larger outer probability  $q = 0.2$  is used to build the network. That trend is a known consequence of not using a column-centralized matrix in PCA. Actually, the trend is present in the other loadings too. It is more visible for larger values of  $q$  because the larger the  $q$ , the larger the mean value of each column of  $\mathbf{X}$  (i.e., more connections imply at more “ones” instead of “zeros”). For the rotated loadings, it is clearly visible in the fourth (last) segment of length  $N$  of the leading three rotated loadings  $\hat{\mathbf{A}}_{\bullet k}^*$ . For our goal in this paper, that trend is irrelevant.

*Remark 3.* The analysis here could be conducted by the point of view of the eigenvectors  $\mathbf{v}_k$  (the columns of  $\mathbf{V}$ ). In that case, the correct way to obtain the varimax rotated  $\mathbf{v}_k^*$  is using the scaled  $\sigma^{\frac{1}{2}} \mathbf{v}_k$  to find the rotation  $\mathbf{T}$ , and then applying the rotation to the original (not scaled) eigenvectors  $\mathbf{v}_k^* = \mathbf{v}_k \mathbf{T}$ . The pitfall here is that, because the scaled eigenvectors correspond to the common factor loadings, one could assume

incorrectly that the rotated eigenvectors could be obtained simply by rescaling the already rotated factor loadings. However, this process of scaling, rotating, and rescaling yields an *oblique* rotation.

On the one hand, the interpretability provided by the visual inspection of the rotated factor loadings can provide insights into the data set and be used as an exploratory tool. On the other hand, because of the way information is organized, it is straightforward to use each  $\hat{\Lambda}_{\bullet,k}^*$  segment as *binary* classifiers to identify the communities  $r$  that share the same condition  $h$ . This is discussed in the next section.

**C. Segments of  $\hat{\Lambda}_{\bullet,k}^*$  as binary identifiers**

Let  $\hat{\Lambda}_{\bullet,k,h}^*$  denote each segment of length  $N$  of the rotated factor loadings matrix  $k$ th column, with elements  $\hat{\Lambda}_{\bullet,k,h}^*(i)$ ,  $i = 1, 2, \dots, N$ . Those segments are binary classifiers, which can be used to cluster the nodes into one or two groups given their respective weights  $\hat{\Lambda}_{\bullet,k,h}^*(i)$  (by referring to that clusters as “groups” we aim to avoid any confusion with the planted communities  $r$ ). For instance, consider the segment  $\hat{\Lambda}_{\bullet,1,1}^*$  in Fig. 3 (bottom). The weights  $\hat{\Lambda}_{\bullet,1,1}^*(i)$  clearly form two clusters: one with values near the “noise floor” and the other with much larger values than the noise floor variance. That can be seen, as well, in the distribution of weights in Fig. 4. Let us denote those two groups seen in  $\hat{\Lambda}_{\bullet,1,1}^*(i)$  by labels 0 and 1. We define the index vector  $\mathbf{s}^{(k,h)}$  of length  $N$  with elements

$$s_i^{(k,h)} = \begin{cases} 0, & \text{if node } i \text{ belongs to group 0} \\ 1, & \text{otherwise.} \end{cases} \quad (9)$$

On the other hand, note that the “flat” segment  $\hat{\Lambda}_{\bullet,3,2}^*$ , which is the fingerprint of the absence of  $r = 2$  in condition  $h = 2$ , will generate a  $\mathbf{s}^{(3,2)}$  with all its elements equal to 0 (i.e., all nodes are associated to the noise floor, and so to group 0). The differential structure in Fig. 3 is so clearly discernible that the clustering could be done by visual inspection (even for the example with randomization of indices). However, that could potentially lead to rather subjective conclusions, and for large numbers of conditions and communities a nonautomated procedure would be prohibitive.

An alternative solution would be applying a given clustering algorithm of choice. In this paper, we opt to use the density-based spatial clustering of applications with noise [27,28] (DBSCAN), with the parameters’ *number of neighbors* and *distance* equal to 5 and  $\frac{1}{100}$  of the amplitude of the  $\hat{\Lambda}_{\bullet,k,h}^*$ , respectively. By design in the next section’s numerical experiments, the ground truth is known, and hence it is used to construct the index vectors  $\mathbf{s}_{\text{true}}^r$  for each  $r = 1, 2, \dots, R$  planted community.

**D. Statistical analysis**

In order to quantify the accuracy of the rotated  $\hat{\Lambda}_{\bullet,k}^*$  on correctly identifying the communities  $r$  that share the same condition  $h$ , we use the score provided by the area under the receiver operating characteristics curve (AUC-ROC). The ROC curve is a common tool in machine learning, used to compare the performance of binary classifiers. It is a graphical representation where the true positive rate (also called

sensitivity or recall) is plotted against the false positive rate for different thresholds used in the decision function. That score can have values between 1 and 0.5: a random classifier will have AUC-ROC equal to 0.5, whereas a perfect classifier will have ROC-AUC equal to 1.

Then, we compute the AUC-ROC score between a single  $\mathbf{s}^{(k,h)}$  and all the  $R$  other  $\mathbf{s}_{\text{true}}^r$  vectors (see Sec. II C). We pick the largest value and the value of  $r$  for which it occurred, and use them to represent both the method score in identifying the community with label  $r$  through the the  $\hat{\Lambda}_{\bullet,k,h}^*$  segment. Note that the knowledge about the condition  $h$ , where that community  $r$  was found, is already provided by the  $h$  index of the current  $\hat{\Lambda}_{\bullet,k,h}^*$ .

Finally, the overall accuracy will be quantified by the fraction of *reliably* detected communities. Specifically, we count the number of detected communities with a ROC-AUC score above a given threshold and divide that value by the real (known) number of communities. The value of this threshold is 0.8.

**III. RESULTS**

The motivation in this section is twofold. First, numerical experiments are conducted to test the accuracy of the CDR method in unveiling the history of structural changes in more challenging scenarios, by identifying the communities  $r$  that share the same condition  $h$ . We apply the CDR on synthetic networks with  $N = 2000$  nodes in  $H = 100$  different conditions, with a random number of communities (between 10 and 20) of random sizes (10 to 100 nodes) distributed in those conditions. The scope here is on networks of nonoverlapping clusters (or communities), generated from the stochastic block model. The method’s accuracy is investigated through Monte Carlo simulations in scenarios with  $H = 100$  and 10 conditions, where we manipulated the inner probabilities (how strongly the nodes within a community are connected) and the outer probability (how strongly nodes from different communities, and the “noisy background” are connected). Then, we explore how the network size, number of networks, mean size of communities, and number of possible shared communities influence the method performance.

Second, we show evidence that the method can still split and organize the structural information, in a much more complex and real-world scenario, in the context of gene co-expression network analysis of single-cell data. The data set used consists of expression data of 9768 genes, from 2 different cell types and in 2 different experimental conditions. In our specific setting, this means a network with 9768 nodes in  $H = 4$  conditions. While the adjacency matrices in that context are weighted rather than binary, and as well there could be both positive and negative connections, we show that the CDR can find unique and subtle nuances of the structural differences. That is done here for illustrative purposes, and not to unveil any meaningful biological result for the specific data set we used. Biological inference through CDR is currently being investigated, with results to appear soon [29].

While the numerical benchmarks represented here are a much more complex scenario than the previous toy one, it is only a small representation of the myriad ways in which the network structure could change in real-world systems. For



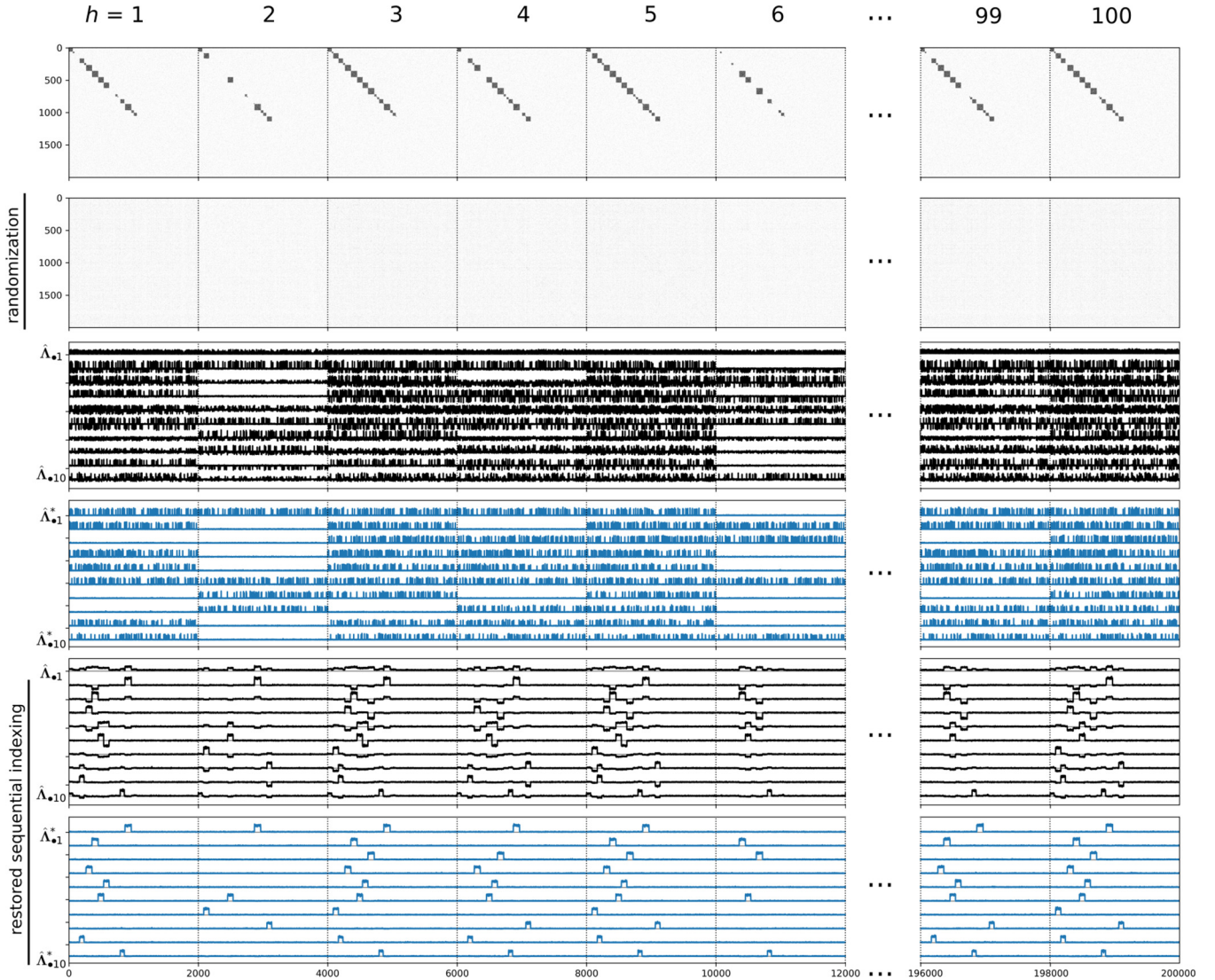


FIG. 5. Concatenated matrix  $\mathbf{X}$  (top) for the large network ( $N = 2000$ ) with  $R = 20$  communities randomly distributed along  $H = 100$  conditions (before and after randomization of indices). Community sizes  $10 \leq N_r \leq 100$  were sampled from a uniform distribution. Inner and outer probabilities are  $(p, q) = (0.6, 0.02)$ . Before rotation (black) the factor loadings  $\hat{\mathbf{A}}_{*k}$  show mixed signatures of the communities (clearly visible with the restored sequential indexing). After rotation (blue), each community in each “shared” condition is clearly represented within the same  $\hat{\mathbf{A}}_{*k}^*$ . See Fig. 6 for the respective scatter plots.

instance, we are focusing on communities that could only be present or absent. In reality, there can be superpositions, growing and shrinking, combinations of those processes, etc. Notwithstanding, as the motivation of this paper is to provide a first step for the CDR method, we make our best to explore this constrained scenario deeply.

### A. Synthetic networks

We start by mimicking the scenario of a large network with  $N = 2000$  nodes in  $H = 100$  conditions. In a similar fashion as in the toy scenario of Fig. 1, a set of 100 adjacency matrices  $\{\mathbf{A}_h\}_{h=1}^{100}$  was generated using the SBM. However, the difference is that the number of planted communities in each condition, the community sizes and their specific labels were chosen from random uniform distributions. Figure 5 (top) shows the 10 blocks of the concatenated matrix

$\mathbf{X} = [\mathbf{A}_1 \mathbf{A}_2, \dots, \mathbf{A}_{100}]$  associated with the first seven and last two adjacency matrices. The inner and outer probabilities are  $p = 0.6$  and  $q = 0.02$ .

The specific steps to generate this scenario are as follows:

- (1) An array containing 20 tuples  $(r, N_r)$  was generated with  $N_r \sim \mathcal{U}(10, 100)$ , associating a given community label  $r = 1, \dots, 20$  with its respective community size.
- (2) An array containing 100 tuples  $(h, R_h)$ , with  $R_h \sim \mathcal{U}(10, 20)$ , specifies the number of communities  $R_h$  to be planted in a condition  $h$ .
- (3) Fixing  $h$ , the adjacency matrix  $\mathbf{A}_h$  was generated with  $R_h$  communities as specified by the step (2), but with community labels randomly sampled (with equal probability) from the array generated in step (1).
- (4) Step (3) was repeated for  $h = 1, \dots, 100$ .

Note that, as in the toy scenario, the nodes belonging to the same community with label  $r$ , but in different conditions



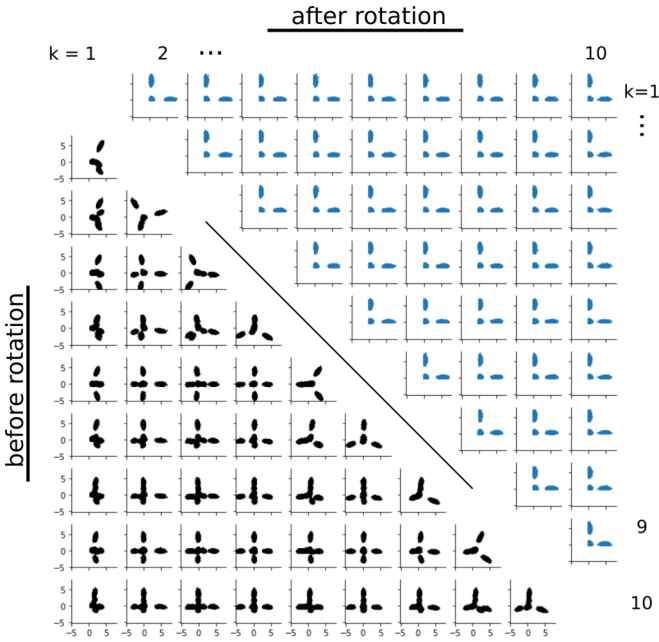


FIG. 6. Scatter plots of the leading 10 factor loading vectors of Fig. 5, obtained through the CDR from the networks *with* randomization of indices. The CDR can split and organize the information of the shared network structures (see text).

$h$ , will have a different specific internal (and external) connections. Only their inner and outer probabilities are the same across conditions.

In summary, each condition  $h = 1, \dots, H$  can have 10 to 20 planted communities, and each of them can have (or not) different sizes. A given community can appear in several conditions, but its internal and external connectivity will be (very likely) different.

The leading 10 loadings before and after rotation are shown in Fig. 5. As in Sec. II A, we present the results for both the sequential indexing of nodes and the randomization of indices. The varimax rotation was performed with  $S = 2R = 40$  vectors, assuming that the double of the maximum allowed number of communities (in any condition) will provide a sufficient degree of freedom for the algorithm to achieve the desired simple structure. The expected mixing before rotation, and clear representation of the differential network structure after rotation, can be seen. Comparing the rotated factor loadings with and without randomization of indices, it is clear that the CDR can unfold and organize the shared structures along with the different networks. Its performance and limitations will be investigated now by using each segment as binary identifiers (as proposed in Sec. II C), it is worth mentioning the fingerprints of the smallest communities at the last loadings  $\hat{\Lambda}_{\bullet k}^*$  (e.g., for  $k = 20$ , not shown) would *not* be visually discernible from the random “noise floor” in a real application (i.e., with nonsequential indexing). However, next we show that they can still be used as binary classifiers for the DBSCAN and correctly identify these communities and where (which  $h$ ) they correspond to.

A clear representation of what the CDR is achieving is provided by the scatter plots in Fig. 6. It shows  $\hat{\Lambda}_{\bullet k_1}^*$  versus  $\hat{\Lambda}_{\bullet k_2}^*$  (before rotation) and  $\hat{\Lambda}_{\bullet k_1}^*$  versus  $\hat{\Lambda}_{\bullet k_2}^*$  (after rotation),

with  $k = 1, \dots, 10$  and  $k_1 \neq k_2$ . Before rotation, two features can be seen. The first one is that different factor loading vectors can be correlated. This can be seen in the clusters that are not located mainly along the horizontal or vertical axis. The second is that each vector contains information of several clusters, which can be seen for all cases in the figure. After rotation, the information is completely reorganized: (i) no vectors remain correlated, (ii) and each vector contains information only of one community (cluster) and the noise background. Note that those scatter plots are equivalent for both successive and randomized indexings.

Now, we assess the fidelity of this representation provided by the rotated  $\hat{\Lambda}_{\bullet k}^*$ . The AUC-ROC scores are shown in Fig 7(a) (for the sake of clarity, only values above 0.6 are shown). The distribution of values will be discussed later. Still, one can see that the majority of them are near 1: the best achievable balance between almost perfect (i) sensitivity (100% true positive rate) and specificity (0% false positive rate). Those values can be contrasted with the ground truth shown in Fig. 7(b). The filled squares indicate the planted communities: black if the community was identified by the method (i.e., a ROC-AUC score above 0.8), and red otherwise. The fraction between detected and planted communities (the number of black squares divided by the total number of squares) is 0.84. That means 921 successful identifications, and 130 misses. By comparison with the actual community sizes  $N_r$ , given by plotting the array with 20 tuples  $(r, N_r)$  in Fig 7(c), we see that the small communities, with size near 10, are more likely to be missed.

Figure 8 shows the distribution of ROC-AUC scores relative to the community sizes. Two features can be seen, which confirms the previous discussion. First, communities with size above 60 nodes were detected with a score above 0.98, and they form the vast majority of identifications [see Fig. 8(b)]. Second, the smaller communities can be harder to detect, and this is more prominently seen for communities with size below 20 nodes.

The previous results depend on the specific realization that generates the 20 tuples  $(r, N_r)$  [Fig. 7(c)], as well as the other random features. Furthermore, we would like to explore how the mixing (relative magnitudes between  $p$  and  $q$ ) influences those results. So, we now employ a Monte Carlo strategy, for which the previous scenario can be considered one of its specific realizations. The steps are as follows:

- a. We fix in inner and outer probabilities  $(p, q)$ .
- b. Matrix  $\mathbf{X} = [\mathbf{A}_1 \mathbf{A}_2 \dots \mathbf{A}_H]$  is generated by using steps 1–4.
- c. The fraction of communities detected and planted is calculated as before.
- d. Steps a–c are repeated  $N_{MC} = 20$  times.

That was done for 20 increasing values of the outer probability  $q \in [0.01, 0.9]$ . We considered scenarios with inner probability  $p = 0.6, 0.8$ , and 1 (1-cliques). Figure 9(a) shows the result (mean  $\pm$  standard deviation) for  $H = 100$  conditions. It is seen that the lower the mixing (i.e.,  $p$  been more prominent than  $q$ ) the detection curve (i) decays slower and (ii) starts decaying at a larger- $q$  value. As mentioned before, if a given small community appears in several conditions, its “signal” would be stronger. That would counterbalance its small size, allowing it to be detected. Because of that, we

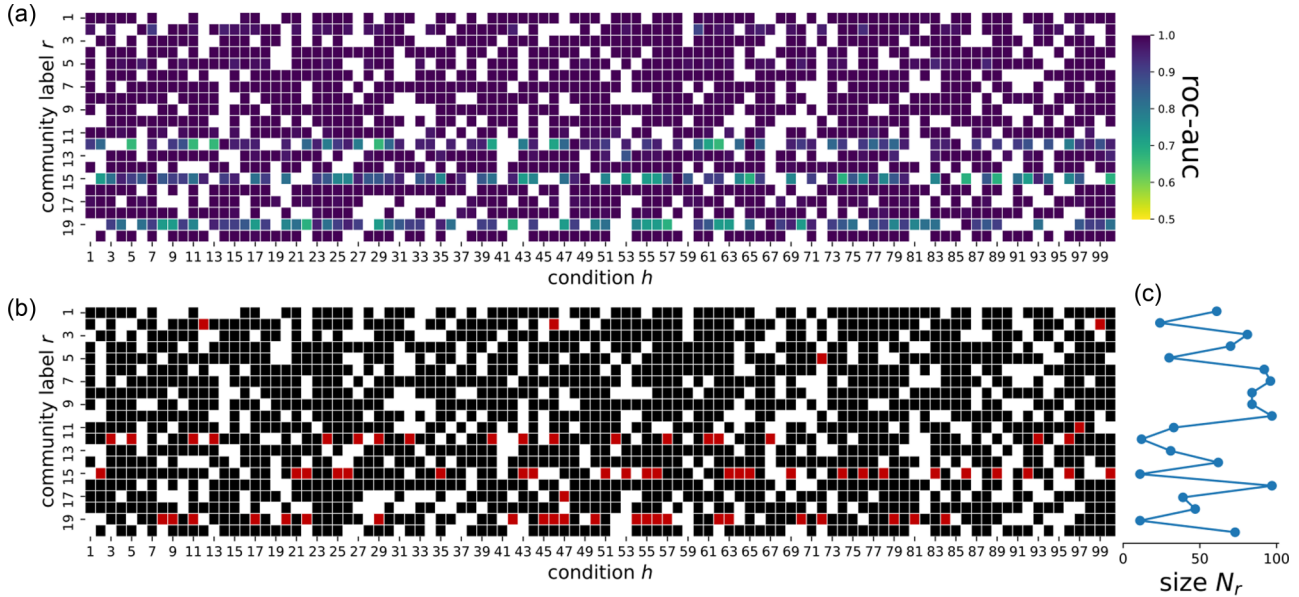


FIG. 7. Communities planted and detected for the large network in  $H = 100$  conditions (see Fig. 5). (a) ROC-AUC scores of the communities detected by the rotated loadings  $\hat{\Lambda}_k^*$ . For clarity, only scores above 0.6 are shown. We considered a *successful* detection if the ROC-AUC score is above 0.8. (b) The ground truth: communities planted in each condition  $h$ . Colors indicate if they were successfully detected (black) or not (red). (c) The community sizes  $N_r \sim \mathcal{U}(10, 100)$  used in this simulation. Smaller communities are more likely to be missed by the method.

repeat the experiment for a much lower number of conditions  $H = 10$ . In this scenario, it is much less likely that any of the 20 possible communities will appear several times. As a consequence [Fig. 9(b)], the detection curves start decaying at a lower value of  $q$  as compared to the scenario with  $H = 100$ . Regarding the speed of decay (slope), it is very similar to the previous case but for a small segment (between  $q \approx 0.1$  and 0.2) for the  $p = 1$  curve.

Now, we investigate how the number of planted networks  $R$  and the total number of nodes  $N$  influences the CDR accuracy. For this, we fixed the number of networks (conditions)  $H = 10$  and the connection probabilities  $(p, q) = (0.8, 0.2)$ , which correspond to the vertical dotted line in Fig. 9(b). For

each  $h$ , the number of planted communities  $R_h$  is drawn from a random uniform distribution centered at  $R_{\text{mean}}$  with a dispersion of 20%. Specifically,  $R_h \sim \mathcal{U}([0.8R_{\text{mean}}], [1.2R_{\text{mean}}])$ .

The parameter plane  $(R_{\text{mean}}, N)$  is explored as follows. Ten linearly spaced values  $R_{\text{mean}} \in [5, 50]$  are used. For the network size, we use nine linearly spaced values  $N \in [1000, 5000]$ . For each combination  $(R_{\text{mean}}, N)$ , we calculate the mean value of the fraction of correctly identified communities over 20 different realizations.

Figure 10 show the results. For the range of parameters explored, there is a noticeable dependence on the network size. The larger  $N$ , the smaller the ratio of identified communities. On the contrary, there is very little dependence (if any) upon the (mean) number of planted communities  $R_{\text{mean}}$ . Note that a high value of  $q = 0.2$  has been used here. For values  $q < 0.1$  (not shown), no noticeable dependence was found. As a remark, because of the random sampling of  $R_h$ , the sum of community sizes could be larger than  $N$  at some realizations. If that occurs for at least one of the 20 realizations, we considered the mean fraction as a missing value, which is responsible for the white squares in the figure.

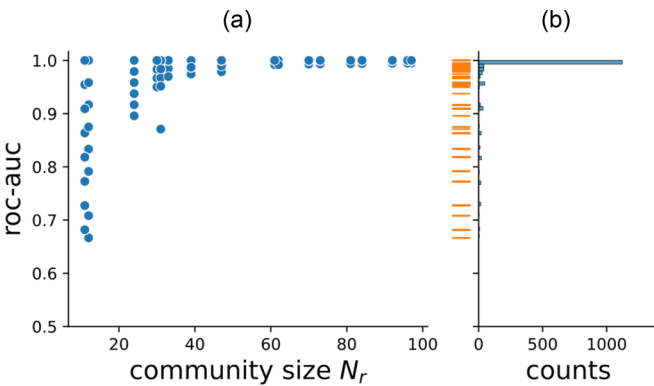


FIG. 8. Distribution of ROC-AUC scores shown in Fig. 7(b). The fidelity of community detection (a) is higher for larger communities. Because several scores are equal to 1, the respective markers are overlapped, and in (b) one can see better their distribution. Remark: for the sake of clarity, only values with ROC-AUC larger than 0.6 are plotted.

**B. Genetic data**

Here we give an illustrative example of how the CDR can split and organize the information about the shared community structures of real-world data, which could be used in the context of differential gene coexpression network analysis. The scope in this section is purely mathematical. The use of CDR for *biological* inference is currently being investigated in collaboration with a team of bioinformaticians, with results to appear soon [29].

We will use a gold-standard data set [30] for single-cell analysis with 12 different cell types and two experimental

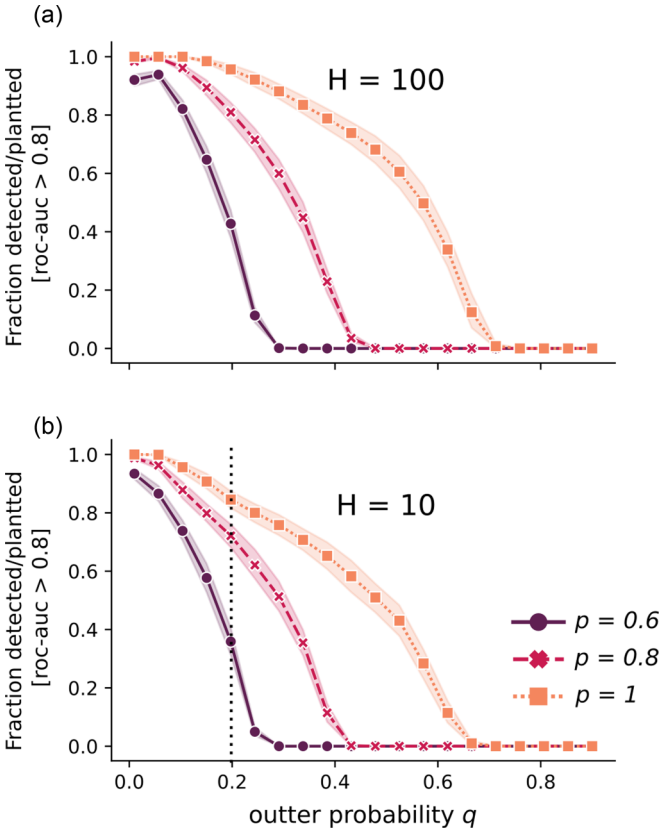


FIG. 9. Fraction of detected communities with ROC-AUC score above 0.8. The curves show the mean ( $\pm$  standard deviation) over 20 Monte Carlo runs for an increasing outer probability  $q \in [0.01, 0.9]$  and three fixed values of the inner probability  $p$ . The number of conditions is (a)  $H = 100$  (see Figs. 5–7) and (b)  $H = 10$ , both for a network of size  $N = 2000$ . A larger  $H$  increases the chance of a given community appearing on multiple conditions, which increases its chance of being detected. This causes the slower decay of the curves in (a) as compared to (b).

conditions (treatment group). To illustrate the CDR, we arbitrarily selected two cell types: CD4 naive  $T$  cell (henceforth referred as  $T$  cells) and CD14-mono cells, both labeled as STIM (IFNB stimulation) or CTRL (control) regarding their

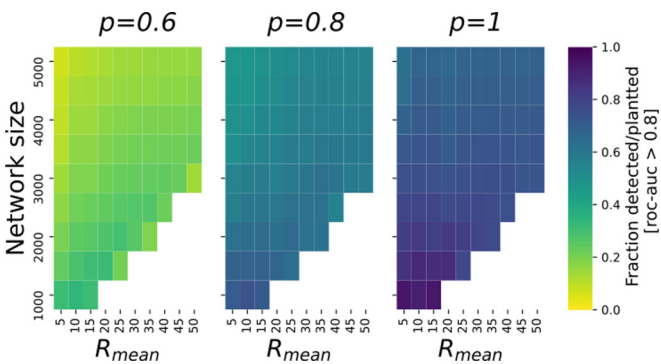


FIG. 10. Fraction of detected communities in the parameter plane ( $R_{\text{mean}}, N$ ), corresponding to the vertical dotted line in Fig. 9(b) [i.e.,  $(H, q) = (10, 0.2)$ ]. Values represent the mean over 20 Monte Carlo runs.

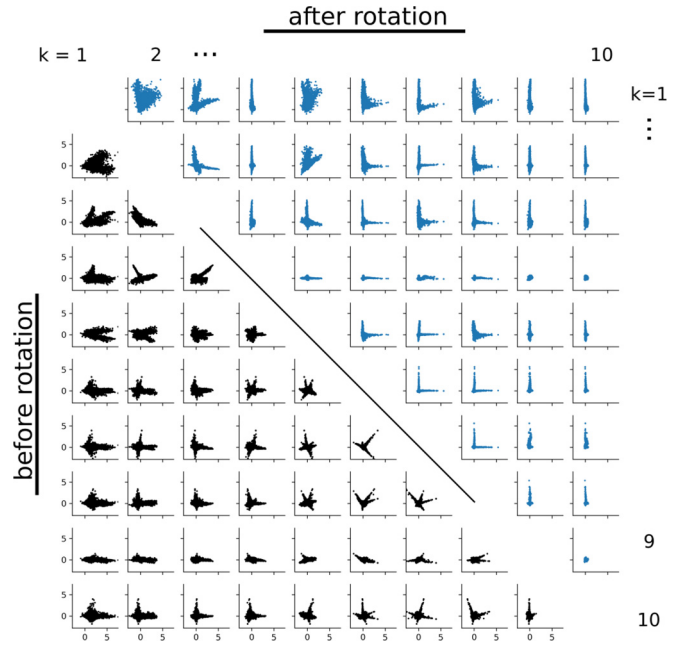


FIG. 11. Genetic data scatter plots (before and after rotation) show the enhanced interpretability (i.e., simple structure) provided by the varimax rotation as compared to the unrotated  $\hat{\Lambda}_{\bullet k}$ .

respective treatment group. Accordingly,  $H = 4$  weighted and undirected (coexpression) networks  $\mathbf{W}_h \in \mathbb{R}^{9768, 9768}$ , which conceptually represent what we called conditions in the CDR method, were built with the 9768 genes with larger expression variance across samples (see Appendix B for details).

Finally, the input for the CDR was the matrix  $\mathbf{X} = [\mathbf{W}_1 \mathbf{W}_2 \mathbf{W}_3 \mathbf{W}_4]$  of size  $9768 \times 39072$ . The varimax rotation was performed with  $S = 60$ . We experimented with different values (e.g., 40 and 100), with no impact on the following results.

Figure 11 shows the scatter plots of the leading 10 vectors, both before and after rotation. One can see that the rotation, in general, was able to split the “arms” of the “starlike” shapes into *different* and *uncorrelated* vectors. The few vectors that remain correlated became more aligned to the horizontal and vertical axes. However, note that in contrast with the well behaved scenario from the SBM, there is no clear-cut distinction between different distributions (clusters) in those scatter plots. Nevertheless, the information of the shared structures is much more organized.

Now, we turn to a specific question: *Do the split and organized nuanced features correspond to the mathematical ground truth within the respective correlation matrices?* Figure 12 shows the leading seven rotated factor loading vectors  $\hat{\Lambda}_{\bullet k}^*$ . From them, we arbitrarily selected two segments and took the gene labels corresponding to the six largest absolute loadings: the sets  $L_1$  and  $L_2$ .

The set  $L_1$  corresponds to six genes with the exact same (high) loadings at the  $T_{\text{ctrl}}$  condition, which are highlighted with a rectangular (orange) box in Fig. 12(a). Their loadings are near to zero at the other conditions. Those features indicate that the  $L_1$  genes form a highly connected community only at the first condition. Figure 12(b) shows that the prediction



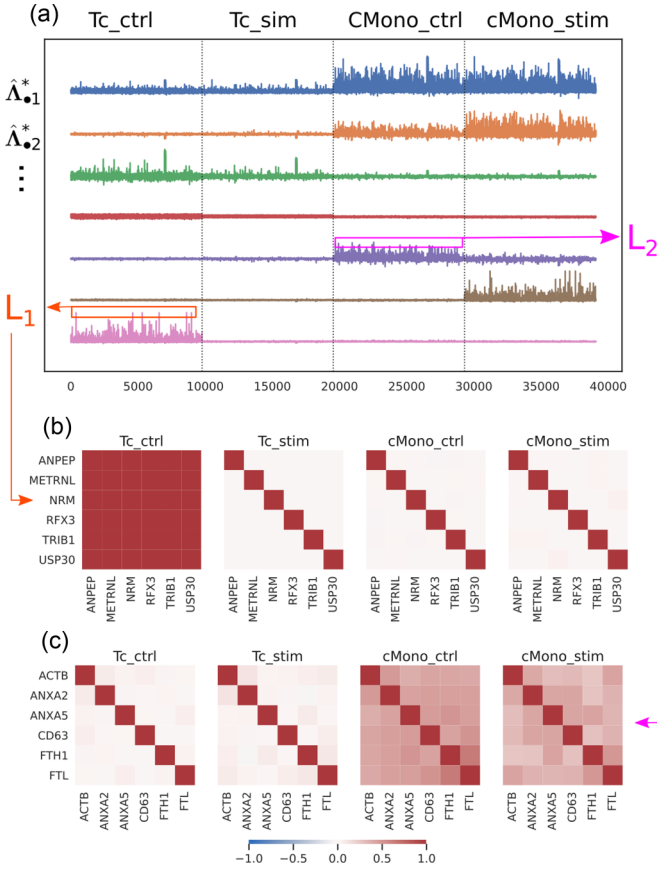


FIG. 12. Applying the CDR framework to gene coexpression networks. (a) Cell and/or group specific nuances are clearly suggested by the rotated loading vectors  $\hat{\Lambda}_{*k}$ . (b) Correlations in the original expression data for the selected genes in the set  $L_1$  show that they form a highly connected community *specific* for Tc cells in the control group. (c) As before, but for the genes from the set  $L_2$ . These genes are part of a more complex structure of highly connected nodes specific for cMono cells, and more “stronger” in the control group than in the stimulated one.

fully agrees with the correlations of the  $L_1$  genes within each *original* expression data set.

The set  $L_2$  (pink box) corresponds to six genes with high but different loadings at the cMono<sub>ctrl</sub> condition. Some of them still have more than average loadings at the last condition cMono<sub>stim</sub>, but for all other conditions their loadings are almost zero. The prediction here is that the  $L_2$  genes form a highly connected community at the third condition, and some aspects of that community are present and attenuated at the last condition. Again, this is in agreement with the correlations seen through the *original* expression data set, Fig. 12(c).

IV. CONCLUSION

In this paper, we have proposed a theory and method (CDR) for the characterization of shared “structural roles” of nodes *simultaneously* within and between networks, by splitting and organizing that information in a meaningful way. The outcome is a highly interpretable map that can be used for automatic feature extraction and as an exploratory tool.

Without loss of generality, for the sake of presentation, we have assumed that each network represents different experimental conditions. They could be, equivalently, representations of time-evolving network (i.e., in different time points) layers in a multiplex network, etc.

Supported by a transparent and straightforward theory, rooted in the factor analysis framework, the method provides flexibility to address different research-field-specific questions. This is accomplished by defining *what* is the scientific-meaningful characteristic (or relevant feature) of a node at the problem at hand, and then mapping it to an appropriate mathematical similarity construct to estimate the proximity from measured data. In the context of differential network analysis, with communities of highly connected nodes, in this paper the method was illustrated by assuming as the relevant feature the similarity of the list of neighbors between nodes, captured by the notion of proximity through the inner (vector) product.

The insights provided by the method and its accuracy have been explored in numerical benchmarks generated by a stochastic block model. In the scope of nonoverlapping communities (which could be present or absent on a given condition), the results have shown the method’s high accuracy despite very different (i) community sizes, (ii) total number of communities within a given condition, and (iii) number of networks being compared (e.g., experimental conditions).

The results from the single-cell gene expression data set have provided evidence that the method can still split and organize the structural information in a much richer and complex scenario. How to use that outcome to extract biologically meaningful information is currently being investigated, with promising results to appear soon.

Aside from its potential use as an automatic feature extraction tool and preprocessing tool, we discuss that another potential strength of the method is its “story-telling”-like characterization of the information encoded in a set of networks. We hope that could be used to pinpoint unexpected shared structure, leading to further investigations and providing new insights.

ACKNOWLEDGMENTS

L.L.P. and M.S. acknowledge financial support from Australian Research Council Discovery Grant (DP Grant No. 180100718). This work was partly supported by the Pawsey Supercomputing Centre with funding from the Australian Government and the Government of Western Australia.

APPENDIX A: NONSEQUENTIAL INDEXING

Consider the column vector  $\mathcal{Y} = [y_1 \dots y_{HN}]^T$ . Each consecutive  $h = 1, \dots, H$  block of rows (with size  $N$ ) represents the nodes of each  $h$  network. If one shuffles the nodes’ indices, that means that the same shuffling is applied on each  $H$  segment of  $\mathcal{Y}$ . Let us call the matrix that performs that specific “block” permutation on vector  $\mathcal{Y}$  by  $P_\pi^*$ . Note that  $\{P_\pi^*\} \subset \{P_\pi\}$ , meaning that the subset of all “block” permutations is a subset of all possible permutations. In the following, we work with the generic  $P_\pi$ .

Shuffling the nodes' indices gives the new column vector  $\tilde{\mathcal{Y}} = P_\pi \mathcal{Y}$ . That permutation propagates into the new factor loading matrix as follows. First, we have the new

$\tilde{\mathbf{R}}_{\mathcal{Y}\mathcal{Y}} = \tilde{\mathcal{Y}}\tilde{\mathcal{Y}}^\top = P_\pi \mathcal{Y}(P_\pi \mathcal{Y})^\top = P_\pi \mathcal{Y}\mathcal{Y}^\top P_\pi^\top = P_\pi \mathbf{R}_{\mathcal{Y}\mathcal{Y}} P_\pi^\top$ . Second, given (5) and (6), the new factor loading matrix is  $\tilde{\mathbf{\Lambda}} = P_\pi \mathbf{\Sigma}^{1/2} \mathbf{V} \equiv P_\pi \mathbf{\Lambda}$ . So, the permutation of nodes' indices implies at the same permutation of the rows of  $\mathbf{\Lambda}$ , and not of its columns. If a block permutation  $P_\pi^*$  is used, that guarantees that the same splitting and organizing outcome of CDR is obtained: each shared feature corresponds to the *same segments* of the original columns  $\mathbf{\Lambda}$  (i.e.,  $\tilde{\mathbf{\Lambda}}_{\cdot k}^*$  and after rotation).

## APPENDIX B: GENETIC DATA

The single-cell data set [30] consists of gene expression data of peripheral blood mononuclear cells (PBMCs) divided into two groups based on treatment: one with and the other without interferon- $\beta$  (IFNB) stimulation. Both the data set and one tutorial (with R code) for its analysis can be found in

Ref. [31] and within the SEURAT R package [32]. The data set contains the expression data of 14 053 genes for 12 different cell types. To illustrate the CDR, we arbitrarily chose the expression data of CD4 naive T cell and CD14-mono cells, in the STIM (IFNB stimulation) and CTRL (control) conditions, were used to build  $H = 4$  gene coexpression networks, as follows.

Let the matrices  $\mathbf{E}_h$ , with  $h = 1, \dots, H$ , be the (log-normalized) expression data with genes in rows and samples in columns. From the 14 053 genes, we removed the ones with a variance less than  $10^{-10}$ , and then selected the genes with the highest variance corresponding to the 0.1 quantile (9768 genes). That resulted at four expression data matrices with sizes  $\mathbf{E}_1 \in \mathbb{R}^{9768, 1034}$ ,  $\mathbf{E}_2 \in \mathbb{R}^{9768, 1579}$ ,  $\mathbf{E}_3 \in \mathbb{R}^{9768, 1036}$ , and  $\mathbf{E}_4 \in \mathbb{R}^{9768, 3285}$ . For each one, a coexpression similarity matrix  $\mathbf{W}_h = \text{cor}(\mathbf{E}_h)$  of size  $9768 \times 9768$  was estimated, with  $h = 1, \dots, 4$ . To decrease the computational time of the eigendecomposition, all correlations between  $\pm 0.02$  were set to zero. The concatenation of those matrices,  $\mathbf{X} = [\mathbf{W}_1 \mathbf{W}_2 \mathbf{W}_3 \mathbf{W}_4]$  of size  $9768 \times 39\,072$ , was used as the input for CDR.

- 
- [1] A. Barabási, *Nat. Phys.* **8**, 14 (2012).
- [2] M. Watson, *BMC Bioinf.* **7**, 509 (2006).
- [3] Y. Huang, A. Panahi, H. Krim, and L. Dai, *IEEE Trans. Network Sci. Eng.* **7**, 1697 (2020).
- [4] A. Ghasemian, P. Zhang, A. Clauset, C. Moore, and L. Peel, *Phys. Rev. X* **6**, 031005 (2016).
- [5] H. A. Chowdhury, D. K. Bhattacharyya, and J. K. Kalita, *IEEE/ACM Trans. Computat. Biol. Bioinformatics* **1**, 1 (2019).
- [6] A. R. Sonawane, S. T. Weiss, K. Glass, and A. Sharma, *Front. Genet.* **10**, 1 (2019).
- [7] S. van Dam, U. Vösa, A. van der Graaf, L. Franke, and J. P. de Magalhães, *Briefings Bioinformatics* **19**, bbw139 (2017).
- [8] R. Anglani, T. M. Creanza, V. C. Liuzzi, A. Piepoli, A. Panza, A. Andriulli, and N. Ancona, *PLoS One* **9**, e87075 (2014).
- [9] D. Amar, H. Safer, and R. Shamir, *PLoS Comput. Biol.* **9**, e1002955 (2013).
- [10] L. L. Portes and M. Small, *Phys. Rev. E* **100**, 042218 (2019).
- [11] L. L. Portes and L. A. Aguirre, *Phys. Rev. E* **93**, 052216 (2016).
- [12] L. L. Portes and L. A. Aguirre, *Chaos* **26**, 093112 (2016).
- [13] A. Groth and M. Ghil, *Phys. Rev. E* **84**, 036206 (2011).
- [14] X. Xiao, A. Moreno-Moral, M. Rotival, L. Bottolo, and E. Petretto, *PLoS Genet.* **10**, e1004006 (2014).
- [15] M. Vejmelka and M. Paluš, *Chaos* **20**, 033103 (2010).
- [16] M. E. J. Newman, *Phys. Rev. E* **94**, 052315 (2016).
- [17] S. Fortunato and D. Hric, *Phys. Rep.* **659**, 1 (2016).
- [18] B. Ball, B. Karrer, and M. E. J. Newman, *Phys. Rev. E* **84**, 036103 (2011).
- [19] I. Koch, *Analysis of Multivariate and High-Dimensional Data*, Cambridge Series in Statistical and Probabilistic Mathematics (Cambridge University Press, Cambridge, 2013).
- [20] S. A. Mulaik, *The Foundations of Factor Analysis*, 2nd ed. (Taylor & Francis, New York, 2010), p. 524.
- [21] P. W. Holland, K. B. Laskey, and S. Leinhardt, *Social Networks* **5**, 109 (1983).
- [22] A. A. Hagberg, D. A. Schult, and P. J. Swart, in *Proceedings of the 7th Python in Science Conference*, edited by G. Varoquaux, T. Vaught, and J. Millman (Pasadena, CA USA, 2008), pp. 11–15.
- [23] P. Erdos and A. Renyi, *Publ. Math. Inst. Hungary. Acad. Sci.* **5**, 17 (1960).
- [24] L. L. Thurstone, *Psychol. Rev.* **38**, 406 (1931).
- [25] H. F. Kaiser, *Psychometrika* **39**, 31 (1974).
- [26] M. B. Richman, *J. Climatology* **6**, 293 (1986).
- [27] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu, *ACM Trans. Database Syst.* **42**, 1 (2017).
- [28] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, in *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, edited by E. Simoudis, J. Han, and U. Fayyad (AAAI Press, 1996), pp. 226–231.
- [29] L. L. Portes, M. Chin, M. Small, and T. Lassmann (unpublished).
- [30] H. M. Kang, M. Subramaniam, S. Targ, M. Nguyen, L. Maliskova, E. McCarthy, E. Wan, S. Wong, L. Byrnes, C. M. Lanata *et al.*, *Nat. Biotechnol.* **36**, 89 (2018).
- [31] [https://satijalab.org/seurat/v3.1/immune\\_alignment.html](https://satijalab.org/seurat/v3.1/immune_alignment.html)
- [32] T. Stuart, A. Butler, P. Hoffman, C. Hafemeister, E. Papalexi, W. M. M. III, Y. Hao, M. Stoeckius, P. Smibert, and R. Satija, *Cell* **177**, 1888 (2019).