Statistica Sinica **34** (2024), 933-954
doi:https://doi.org/10.5705/ss.202021.0427

A UNIFIED FRAMEWORK FOR TUNING HYPERPARAMETERS IN CLUSTERING PROBLEMS

Xinjie Fan[#1], Y. X. Rachel Wang[#*2], Purnamrita Sarkar[1] and Yuguang Yue[1]

[1] *University of Texas at Austin and* [2] *University of Sydney*

Abstract: In general, selecting hyperparameters for unsupervised learning problems is challenging, owing to the lack of ground truth for validation. Despite the prevalence of this problem in statistics and machine learning, especially in clustering problems, there are not many methods for tuning these hyperparameters with theoretical guarantees. In this paper, we provide a framework that relies on maximizing a trace criterion connecting a similarity matrix with clustering solutions. This framework has provable guarantees for selecting hyperparameters in a number of distinct models. We consider both the sub-Gaussian mixture model and network models as examples of independently and identically distributed (i.i.d.) data and non-i.i.d. data, respectively. We demonstrate that the same framework can be used to choose the Lagrange multipliers of the penalty terms in semidefinite programming relaxations for community detection and the bandwidth parameter for constructing kernel similarity matrices for spectral clustering. By incorporating a cross-validation procedure, we show that the framework also provides consistent model selection for network models. Using a variety of simulated and real data examples, we show that our framework outperforms other widely used tuning procedures in a broad range of parameter settings.

Key words and phrases: Clustering, hyperparameter tuning, model selection, network models, sub-Gaussian mixtures.

1. Introduction

A standard statistical model has parameters, which characterize the underlying data distribution; an inference algorithm to learn these parameters typically involve hyperparameters (or tuning parameters). Popular examples include the penalty parameter in regularized regression models, the number of clusters in clustering analysis, the bandwidth parameter in kernel-based clustering, nonparameteric density estimation or regression methods (Wasserman (2006); Tibshirani, Wainwright and Hastie (2015)). It is well known that selecting these hyperparameters may require repeated training in order to search through combinations of plausible hyperparameter values and often has to rely on good heuristics and the user's domain knowledge.

Cross-validation (CV) is a nonparametric procedure often used to perform

*Corresponding author.
#These two authors contributed equally to this work.

automated hyperparameter tuning (Stone (1974); Zhang (1993)), and has been used extensively in machine learning and statistics (Hastie, Tibshirani and Friedman (2001); Feng and Simon (2020)). CV has been studied extensively in supervised learning settings, particularly for low-dimensional linear models (Shao (1993); Yang (2007)) and for penalized regression in high dimension (Wasserman and Roeder (2009)). Other notable stability-based methods for model selection in similar supervised settings include those of Breiman (1996), Bach (2008), Meinshausen and Bühlmann (2010) and Lim and Yu (2016). Finally, numerous empirical methods exist in the machine learning literature for tuning hyperparameters in various training algorithms (Bergstra and Bengio (2012); Bengio (2000); Snoek, Larochelle and Adams (2012); Bergstra et al. (2011)). However, few of these methods provide theoretical guarantees.

In contrast to the supervised setting with i.i.d. data used in many of the aforementioned methods, we consider *unsupervised* clustering problems with a possible dependence structure in the data points. We propose an overarching framework for hyperparameter tuning and model selection for a variety of probabilistic clustering models. Here, the challenge is two-fold. First, because labels are not available, choosing a criterion for evaluation and a method for selecting hyperparameters is not easy. One may consider splitting the data into folds and selecting the model or hyperparameter with the most stable solution. However, for multiple splits of the data, the inference algorithm may get stuck at the same local optima; thus, stability alone can lead to a suboptimal solution (von Luxburg (2010)). Wang (2010) and Fang and Wang (2012) overcome this problem by redefining the number of clusters as that which gives the most stable clustering for a given algorithm. In Meila (2018), a semi-definite program (SDP) maximizing an inner product criterion is performed for each clustering solution, and the value of the objective function is used to evaluate the stability of the clustering. Their analysis is done without model assumptions, and when many clustering solutions need to be evaluated, perfoming SDP for each solution can become computationally expensive. The second difficulty in the unsupervised clustering setting arises if there is a dependence structure in the data points, which necessitates careful splitting procedures in a CV-based procedure.

To illustrate the generality of our framework, we focus on sub-Gaussian mixtures and statistical network models as two examples for i.i.d. data and non i.i.d. data, respectively, where clustering is a natural problem. We diversify the models considered in Fan et al. (2020) in two ways. First, we use a different formulation for sub-Gaussian mixtures with a more realistic noise structure. Second, in addition to the stochastic block model (SBM), we consider the more general mixed membership stochastic block model (MMSB). By observing the fact that clustering algorithms typically operate on a similarity matrix arising from these models, which can be decomposed as signal plus noise, we propose a unified framework that measures the quality of a clustering solution using a

trace criterion involving the similarity matrix. The framework provides provable guarantees for hyperparameter tuning and model selection in these models. Our study contributes to the literature as follows:

1. Our framework can provably tune the following **hyperparameters** in a computationally efficient way, *without* needing CV:
   (a) the Lagrange multiplier of the penalty term in a type of semidefinite relaxation for community detection problems in SBM; and
   (b) the bandwidth parameter used in kernel spectral clustering for sub-Gaussian mixture models.
2. We show that the same framework incorporating a CV procedure performs consistent **model selection** (i.e., determining number of clusters):
   (a) when the model selection problem is embedded in the choice of the Lagrange multiplier in another type of SDP relaxation for community detection in SBM; and
   (b) for general model selection for the MMSB, which includes the SBM as a sub-model.

We choose to focus on model selection for network-structured data, because there already is an extensive repertoire of empirical and provable methods for i.i.d. mixture models, including the gap statistic (Tibshirani, Walther and Hastie (2001)), silhouette index (Rousseeuw (1987)), slope criterion (Birgé and Massart (2001)), eigen-gap (von Luxburg (2007)), penalized maximum likelihood (Leroux (1992)), information theoretic approaches (AIC (Bozdogan (1987)), BIC (Keribin (2000); Drton and Plummer (2017)), minimum message length (Figueiredo and Jain (2002))), spectral clustering, and diffusion-based methods (Maggioni and Murphy (2018); Little, Maggioni and Murphy (2017)). Next, we discuss related work on models considered in this paper.

1.1. Related work

Hyperparameters and model selection in network models: In network analysis, many methods exist for selecting the true number of communities (denoted by r) with consistency guarantees, including those of Lei (2016), Wang and Bickel (2017), Le and Levina (2015), and Bickel and Sarkar (2016) for SBM, and that of Fan et al. (2022) for more general models such as the degree-corrected mixed membership block model. However, these methods have not been generalized to other hyperparameter selection problems. For CV-based methods, existing strategies involve node splitting (Chen and Lei (2018)) or edge splitting (Li, Levina and Zhu (2020)). In the former, it is established that CV prevents underfitting for model selection in SBM. In the latter, a similar one-sided consistency result for random dot product models (which includes SBM as

a special case, see Young and Scheinerman (2007) and a comprehensive survey in Athreya et al. (2017)) is shown. This method has also been applied empirically to tune other hyperparameters, although no provable guarantee was provided.

In terms of algorithms for community detection or clustering, SDP methods have gained a lot of attention (Abbe, Bandeira and Hall (2015); Amini and Levina (2018); Guédon and Vershynin (2016); Cai and Li (2015); Hajek, Wu and Xu (2016)) due to their strong theoretical guarantees. Typically, SDP-based methods can be divided into two broad categories. The first class maximizes a penalized trace of the product of the adjacency matrix and an unnormalized clustering matrix (see definition in Section 2.2). Here, the hyperparameter is the Lagrange multiplier of the penalty term (Amini and Levina (2018); Cai and Li (2015); Chen and Lei (2018); Guédon and Vershynin (2016)). In this formulation, the optimization problem does not need to know the number of clusters. However, it is implicitly required in the final step, which obtains the memberships from the clustering matrix.

The second class of SDP methods uses a trace criterion with a normalized clustering matrix (definition in Section 2.2) (Peng and Wei (2007); Yan and Sarkar (2021); Mixon, Villar and Ward (2017)). Here, the constraints directly use the number of clusters. Yan, Sarkar and Cheng (2017) use a penalized alternative of this SDP to perform provable model selection for SBMs. However, most of these methods require appropriate tuning of the Lagrange multipliers, which are themselves hyperparameters. Usually, the theoretical upper and lower bounds on these hyperparameters involve unknown model parameters, which are nontrivial to estimate. The proposed method in Abbe and Sandon (2015) is agnostic of model parameters, but involves a highly tuned and hard to implement spectral clustering step (also noted by Perry and Wein (2017)).

In this paper, we use an SDP from the first class (SDP-1) to demonstrate our provable tuning procedure, and another SDP from the second class (SDP-2) to establish the consistency guarantee for our model selection method.

Spectral clustering with mixture models: In the statistical machine learning literature, spectral clustering analyses typically use a Laplacian matrix built from an appropriately constructed similarity matrix of the data points. There has been much work (Hein, Audibert and von Luxburg (2005); Hein (2006); Belkin and Niyogi (2003); Giné and Koltchinskii (2006)) on establishing different forms of asymptotic convergence of the Laplacian. Recently, Löffler, Zhang and Zhou (2019) establish error bounds for spectral clustering that uses the gram matrix as the similarity matrix. Srivastava, Sarkar and Hanasusanto (2023) obtain error bounds for a variant of spectral clustering for the Gaussian kernel in the presence of outliers. However, most existing tuning procedures for the bandwidth parameter of the Gaussian kernel are heuristic and do not have provable guarantees. Notable methods include that of von Luxburg (2007), which

chooses an analogous parameter, namely, the radius ϵ in an ϵ-neighborhood graph, "as the length of the longest edge in a minimal spanning tree of the fully connected graph on the data points". Other discussions on selecting the bandwidth can be found in Hein, Audibert and von Luxburg (2005), Coifman et al. (2008) and Schiebinger, Wainwright and Yu (2015). Shi, Belkin and Yu (2008) propose a data-dependent way of setting the bandwidth parameter by suitably normalizing the 95% quantile of a vector containing 5% quantiles of the distances from each point.

We present our problem setup, which applies to both mixture and network models, in Section 2. Section 3 proposes and analyzes our hyperparameter tuning method, named MATR, for networks and sub-Gaussian mixtures. Next, in Section 4, we present MATR with a CV-based extension (MATR-CV) and the related consistency guarantees for model selection for SBM and MMSB models. Section 5 presents detailed simulated and real data experiments. Section 6 concludes the paper with a discussion.

2. Preliminaries and Notation

2.1. Notation

Let $(C_1, \ldots, C_r)$ denote a partition of n data points (or nodes in a network) into r clusters; $m_i = |C_i|$ denotes the size of C_i. Denote $\pi_{\min} = \min_i m_i/n$. The cluster membership of each data point is represented by an $n \times r$ matrix Z, where $Z_{ij} = 1$ if data point i belongs to cluster j, and 0 otherwise. Since r is the true number of clusters, $Z^T Z$ has full rank. Given Z, the corresponding unnormalized clustering matrix is ZZ^T, and the normalized clustering matrix is $Z(Z^T Z)^{-1} Z^T$. For ease of notation, X can be either a normalized or an unnormalized clustering matrix, and will be made clear in the context. We use $\tilde{X}$ to denote the matrix returned by SDP algorithms, which may not be a clustering matrix. Denote $\mathcal{X}_r$ as the set of all possible normalized clustering matrices with cluster number r. Let Z_0 and X_0 be the membership and the corresponding normalized clustering matrix, respectively, from the ground truth. λ is a general hyperparameter; although with a slight abuse of notation, we also use λ to denote the Lagrange multiplier in SDP methods. For any matrix $X \in \mathbb{R}^{n \times n}$, let X_{C_k, C_ℓ} be a matrix such that $X_{C_k, C_\ell}(i, j) = X(i, j)$ if $i \in C_k, j \in C_\ell$, and 0 otherwise. E_n is the $n \times n$ matrix of all ones. We write $\langle A, B \rangle = \text{trace}(A^T B)$. Standard notations of o, O, o_P, O_P, Θ and Ω will be used. Note that "with high probability" (w.h.p.) means with probability tending to one.

2.2. Problem setup and the trace criterion

We consider a general clustering setting, where the data $\mathcal{D}$ gives rise to an $n \times n$ observed similarity matrix $\hat{S}$, where $\hat{S}$ is symmetric. Denote $\mathscr{A}$ as a clustering algorithm that operates on the data $\mathcal{D}$ with a hyperparameter λ and

outputs a clustering result in the form of $\hat{Z}$ or $\hat{X}$. Here, note that $\mathscr{A}$, $\hat{Z}$ and $\hat{X}$ could all depend on λ. We assume that $\hat{S}$ has the form $\hat{S} = S + R$, where R is a matrix of arbitrary noise, and S is the "population similarity matrix". As we consider different clustering models for network-structured data and i.i.d. mixture data, it will be made clear what $\hat{S}$ and S are in each context.

Assortativity (weak and strong): In some cases, we require weak assortativity on the similarity matrix S, defined as follows. Suppose for data points $i, j \in C_k$, $S_{ij} = a_{kk}$. Define the minimal difference between the diagonal term and the off-diagonal terms in the same row cluster as

$$p_{\text{gap}} = \min_k \left(a_{kk} - \max_{\substack{i \in C_k, j \in C_\ell, \\ \ell \neq k}} S_{ij} \right). \tag{2.1}$$

Weak assortativity requires $p_{\text{gap}} > 0$. This condition is similar to the weak assortativity defined for block models (e.g. Amini and Levina (2018)). It is mild compared with strong assortativity, which requires $\min_k a_{kk} - \max_{i \in C_k, j \in C_\ell, \ell \neq k} S_{ij} > 0$.

SBM: The SBM is a generative model of networks with a community structure on n nodes. By first partitioning the nodes into r classes which leads to a membership matrix Z, the $n \times n$ binary adjacency matrix A is sampled from the probability matrix $P_{ij} = Z_i B Z_j^T 1(i \neq j)$, where Z_i and Z_j are the i-th and j-th row, respectively, of the matrix Z, and B is the $r \times r$ block probability matrix. The aim is to estimate the node memberships given A. We assume the elements of B have order $\Theta(\rho)$ with $\rho \to 0$ at some rate.

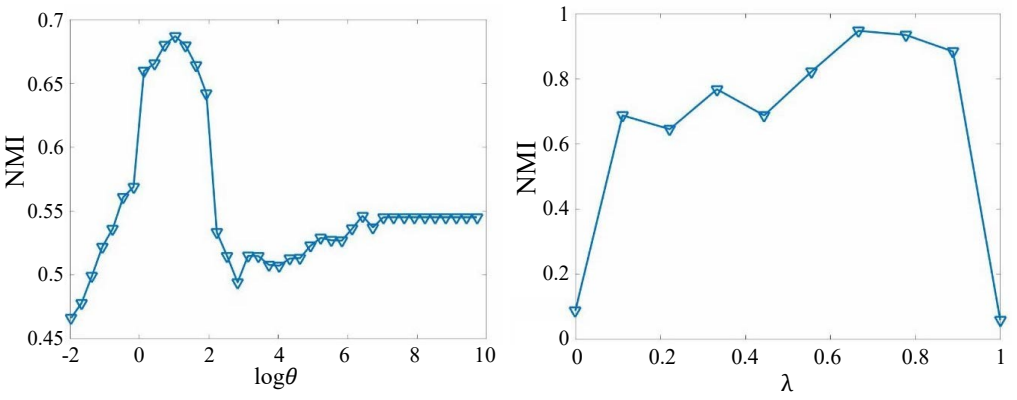
MMSB: The SBM can be restrictive when modeling real world networks. As a result, various extensions have been proposed. The MMSB (Airoldi et al. (2008)) relaxes the requirement on the membership vector Z_i being binary and allows the entries to be in $[0, 1]^r$, such that they sum to one for each i. We denote this soft membership matrix by Θ.

Under the MMSB, the $n \times n$ adjacency matrix A is sampled from the probability matrix P with $P_{ij} = \Theta_i B \Theta_j^T 1(i \neq j)$. We use an analogous definition for the normalized clustering matrix: $X = \Theta(\Theta^T\Theta)^{-1}\Theta$. Note that this reduces to the usual normalized clustering matrix when Θ is a binary cluster membership matrix.

Mixture of sub-Gaussian random variables: Let $Y = [Y_1, \ldots, Y_n]^T$ be an $n \times d$ data matrix. We consider a setting in which Y_i are generated from a mixture model with r clusters:

$$Y_i = \mu_a + W_i, \quad \mathbb{E}(W_i) = 0, \quad Cov(W_i) = \sigma_a^2 I, \qquad a = 1, \ldots, r, \tag{2.2}$$

where W_i are independent sub-Gaussian vectors.



(a) (b)

Figure 1. Tuning hyperparameters in spectral clustering and SDP; accuracy measured by normalized mutual information (NMI). (a) NMI vs. θ, where θ is the bandwidth parameter in kernel spectral clustering; (b) NMI vs. λ, where λ is the Lagrange multiplier in 3.1.

Trace criterion: Our framework is centered around the trace $\langle \hat{S}, X_\lambda \rangle$, where X_λ is the normalized clustering matrix associated with the hyperparameter λ. This criterion is used in relaxations of the k-means objective (Mixon, Villar and Ward (2017); Peng and Wei (2007); Yan, Sarkar and Cheng (2017)) for SDP methods, and in evaluating stability of a clustering solution (Meila (2018)).

The idea is that the trace criterion is large when data points within the same cluster are more similar. As a result, this makes the implicit assumption that the similarity matrix $\hat{S}$ (and S) is assortative, that is, data points within the same cluster have higher similarity based on $\hat{S}$. While this is reasonable for i.i.d. mixture models, the SBM or MMSB may have a mixture of assortative and disassortative structures. In what follows, we assume weak assortativity for the SBM since our algorithms of interest, SDP methods, operate on weakly assortative networks. For the MMSB, which includes the SBM as a sub-model, we show that the same criterion still works without assortativity, if we choose $\hat{S}$ to be A^2 with the diagonal removed.

3. Hyperparameter Tuning with Known r

In this section, we consider tuning hyperparameters when the true number of clusters r is known. First, we provide two simulation studies to motivate this section. The detailed parameter settings for generating the data can be found in the Supplementary Material, Section S3.1.

First, we consider a four-component Gaussian mixture model. We perform spectral clustering (k-means on the top r eigenvectors) on the widely used Gaussian kernel matrix (denoted as K, where $K(i,j) = \exp\left(-\|Y_i - Y_j\|_2^2/(2\theta^2)\right)$

for data points Y_i and Y_j), with the bandwidth parameter θ. Figure 1(a) shows the clustering performance against the ground truth as θ varies using normalized mutual information (NMI), a common metric used to compare two partitions of points. The flat region of suboptimal θ shows when the two adjacent clusters cannot be separated well.

As mentioned in Section 1.1, SDP is an important class of methods for community detection in the SBM, but its performance can depend on the choice of the Lagrange multiplier parameter. We consider the following SDP formulation (Li, Chen and Xu (2018)), which has been widely used, with slight variations, in the literature (Amini and Levina (2018); Perry and Wein (2017); Guédon and Vershynin (2016); Cai and Li (2015); Chen and Lei (2018)),

$$(\mathrm{SDP}-1) \qquad \begin{aligned} &\max\ \mathrm{trace}(AX) - \lambda \mathrm{trace}(XE_n) \\ &\text{s.t. } X \succeq 0, \quad X \geq 0, \quad X_{ii} = 1 \text{ for } 1 \leq i \leq n, \end{aligned} \tag{3.1}$$

where λ is a hyperparameter. Typically, one then performs spectral clustering (i.e., k-means on the top r eigenvectors) on the output of the SDP to get the clustering result. In Figure 1(b), we generate an adjacency matrix from the probability matrix described in the Supplementary Material, Section S3.1 and use (3.1) with tuning parameter λ from 0 to 1. The accuracy of the clustering result is measured by the NMI and shown in Figure 1(b). We can see that different values of λ lead to widely varying clustering performance.

In the general case, we show that when the true cluster number r is known, an ideal hyperparameter λ can be chosen by simply maximizing the trace criterion introduced in Section 2.2. The tuning algorithm (MATR) is presented in Algorithm 1. It takes a general clustering algorithm $\mathscr{A}$, data $\mathcal{D}$, and a similarity matrix $\hat{S}$ as input, and outputs a clustering result $\hat{Z}$ depending on λ^*, chosen by maximizing the trace criterion.

Algorithm 1: MAx-TRace (MATR) based tuning algorithm for known number of clusters.

Input: clustering algorithm $\mathscr{A}$, data $\mathcal{D}$, similarity matrix $\hat{S}$, a set of candidates $\{\lambda_1, \ldots, \lambda_T\}$, number of clusters r;
Procedure:
for $t = 1 : T$ **do**
  run clustering on $\mathcal{D}$: $\hat{Z}_t = \mathscr{A}(\mathcal{D}, \lambda_t, r)$;
  compute normalized clustering matrix: $\hat{X}_t = \hat{Z}_t(\hat{Z}_t^T \hat{Z}_t)^{-1} \hat{Z}_t^T$;
  compute inner product: $l_t = \langle \hat{S}, \hat{X}_t \rangle$;
end for
$t^* = \mathrm{argmax}(l_1, \ldots, l_T)$;
Output: $\hat{Z}_{t^*}$

We have the following theoretical guarantee for Algorithm 1.

Theorem 1. *Consider a clustering algorithm $\mathscr{A}$ with inputs $\mathcal{D}, \lambda, r$, and output $\hat{Z}_\lambda$. The similarity matrix $\hat{S}$ used for Algorithm* 1 *(MATR) can be written as $\hat{S} = S + R$. We further assume S is weakly assortative, with p_{gap} defined in Eq* (2.1)*, and X_0 is the normalized clustering matrix for the true binary membership matrix Z_0. Let $\pi_{\min}$ be the smallest cluster proportion, and $\tau := n\pi_{min}p_{gap}$. As long as there exists $\lambda_0 \in \{\lambda_1, \ldots, \lambda_T\}$, such that $\langle \hat{X}_{\lambda_0}, \hat{S}\rangle \geq \langle X_0, S\rangle - \epsilon$, Algorithm* 1 *will output $\hat{Z}_{\lambda^*}$, such that*

$$\left\|\hat{X}_{\lambda^*} - X_0\right\|_F^2 \leq \frac{2}{\tau}\left(\epsilon + \sup_{X\in\mathcal{X}_r} |\langle X, R\rangle|\right),$$

where $\hat{X}_{\lambda^*}$ is the normalized clustering matrix associated with $\hat{Z}_{\lambda^*}$.

In other words, as long as the range of λ we consider covers some optimal λ value that leads to a sufficiently large trace criterion (compared with the true underlying X_0 and the population similarity matrix S), then the theorem guarantees that Algorithm 1 will lead to a normalized clustering matrix with a small error. The deviation ϵ depends on both the noise matrix R and how close the estimated $\hat{X}_{\lambda_0}$ is to the ground truth X_0, that is, the performance of the algorithm. To better interpret this trace lower bound, if we take $\epsilon = \langle \hat{X}_{\lambda_0} - X_0, S\rangle + \sup_{X\in\mathcal{X}_r} |\langle X, R\rangle|$, then the lower bound on the trace is automatically satisfied. The solution found by Algorithm 1 is then bounded by

$$\left\|\hat{X}_{\lambda^*} - X_0\right\|_F^2 \leq \frac{2}{\tau}\left(\langle \hat{X}_{\lambda_0} - X_0, S\rangle + 2\sup_{X\in\mathcal{X}_r} |\langle X, R\rangle|\right). \tag{3.2}$$

In the bound, the second term is noise, whereas the first term measures the quality of the clustering solution at an ideal λ_0. If both terms are small, then the output from MATR will be close to X_0. Later for specific models, we will give more details on how to interpret the first term. The proof of the theorem is in the Supplementary Material, Section S1.1.

In what follows, we apply MATR to more specific settings, namely, to select the bandwidth parameter in spectral clustering for sub-Gaussian mixtures and the Lagrange multiplier parameter in (3.1) for the SBM.

3.1. Hyperparameter tuning for mixtures of sub-Gaussians

In this case, the data $\mathcal{D}$ is Y defined in Eq (2.2), and the clustering algorithm $\mathscr{A}$ is spectral clustering (see the motivating example in Section 3) on the Gaussian kernel $K(i,j) = \exp\left(-\|Y_i - Y_j\|_2^2/(2\theta^2)\right)$. Note that one could use the similarity matrix as the kernel itself. However, this makes the trace criterion a function of the hyperparameter we are trying to tune, which compounds the difficulty of the problem. For simplicity, we use the negative squared distance matrix as $\hat{S}$, that is, $\hat{S}_{ij} = -\|Y_i - Y_j\|_2^2$. A natural choice for S would be the conditional expectation of $\hat{S}$ given the cluster memberships, which is blockwise constant.

However, this choice would lead to a suboptimal error rate. Therefore, we use a slightly corrected variant of the matrix as S (also see Mixon, Villar and Ward (2017)), called the reference matrix:

$$S_{ij} = -\frac{d_{ab}^2}{2} - \max\left\{0, \frac{d_{ab}^2}{2} + 2(W_i - W_j)^T(\mu_a - \mu_b)\right\} 1(i \in C_a, j \in C_b), \quad (3.3)$$

where $d_{ab} := \|\mu_a - \mu_b\|$, and W_i is defined in Eq (2.2). Note that for i, j in the same cluster, $S_{ij} = 0$. Interestingly, this reference matrix is random itself, which is a deviation from the S used for the network models discussed below. Applying MATR to select θ, we have the following theoretical guarantee, the proof of which can be found in the Supplementary Material, Section S1.2.

Corollary 1. *Let $\hat{S}$ be the negative squared distance matrix, and let S be defined as in Eq* (3.3)*. Let δ_{sep} denote the minimum distance between cluster centers, that is, $\min_{k\neq\ell}\|\mu_k - \mu_\ell\|$. Denote $\alpha = \pi_{max}/\pi_{min}$. As long as there exists $\theta_0 \in \{\theta_1, \ldots, \theta_T\}$, such that $\langle \hat{X}_{\theta_0}, \hat{S}\rangle \geq \langle X_0, S\rangle - n\pi_{min}\epsilon$, Algorithm* 1 *(MATR) will output a $\hat{Z}_{\theta^*}$, such that w.h.p.*

$$\|\hat{X}_{\theta^*} - X_0\|_F^2 \leq C\frac{\epsilon + r\alpha\sigma_{\max}^2(\alpha + \min\{r, d\})}{\delta_{sep}^2},$$

where $\sigma_{\max}$ is the largest operator norm of the covariance matrices of the mixture components, $\hat{X}_{\theta^*}$ is the normalized clustering matrix for $\hat{Z}_{\theta^*}$, and C is a universal constant.

In this setting, ϵ has to be much smaller than δ_{sep}^2 in order to guarantee a small error. As mentioned after Theorem 1, to interpret this trace lower bound involving ϵ, we can set $n\pi_{\min}\epsilon = \langle \hat{X}_{\lambda_0} - X_0, S\rangle + \sup_{X\in\mathcal{X}_r}|\langle X, R\rangle|$, where the second term is absorbed into the noise term in the final bound as usual, and the first term boils down to requiring X_{λ_0} to be close to a computable SDP solution, which is close to X_0 itself. More details and the proof of Corollary 1 can be found in the Supplementary Material, Section S1.2.

3.2. Hyperparameter tuning for SBM

We consider choosing λ in (3.1) for community detection in SBM. Here, the input to MATR, the data $\mathcal{D}$ and the similarity matrix $\hat{S}$, are both the adjacency matrix A. A natural choice of a weakly assortative S is the conditional expectation of A (denoted P) up to diagonal entries, which is blockwise constant. The assortativity condition on S translates naturally to the usual assortativity condition on B, as required by SDP programs. With suitable conditions on the block connectivity separation and estimation error, applying Algorithm 1 (MATR) to tune λ in (3.1) yields a consistent normalized clustering matrix. For brevity, we defer the detailed statement and proofs to the Supplementary Material, Section S1.3.

4. Hyperparameter Tuning with Unknown r

In this section, we adapt MATR to situations where the number of clusters is unknown to perform model selection. Similarly to Section 3, we first explain the general tuning algorithm and state a general theorem that guarantees its performance. Then, applications to specific models will be discussed in the following subsection. Since the applications we focus on are network models, we present our algorithm with the data $\mathcal{D}$ being A for clarity. We present our algorithm using soft membership matrices Θ, which include binary membership matrices as a special case.

We show that MATR can be extended to model selection if we incorporate a CV procedure. In Algorithm 2, we present the general MATR-CV algorithm which takes a clustering algorithm $\mathscr{A}$, adjacency matrix A, and similarity matrix $\hat{S}$ as inputs. Compared with MATR, MATR-CV has two additional parts.

The first part (Algorithm 3) splits the nodes into two subsets for training and testing. This in turn partitions the adjacency matrix A into four submatrices A^{11}, A^{22}, A^{21}, and its transpose, and similarly for $\hat{S}$. MATR-CV makes use of all the submatrices: A^{11} for training, A^{22} for testing, A^{11} and A^{21} for estimating the clustering result for the nodes in A^{22} as shown in Algorithm 4, which is the second additional part. Algorithm 4 clusters testing nodes based on the cluster membership of the training nodes estimated from A^{11} and the connections between the training nodes and the testing nodes A^{21}, the details of which will be explained as we discuss specific models (see Section 4.1 for MMSB and the Supplementary Material, Section S2.3 for SBM).

Like other CV procedures, we note that MATR-CV requires specifying a training ratio γ_{train} and the number of repetitions J. Choosing any $\gamma_{\text{train}} = \Theta(1)$ does not affect our asymptotic results. Repetitions of the splits are used empirically to enhance stability; theoretically, we show asymptotic consistency for any random split. The general theoretical guarantee and the role of the trace gap Δ are given in the next theorem.

Theorem 2. *Given a candidate set of cluster numbers $\{r_1, \ldots, r_T\}$ containing the true number of clusters r, let $\hat{X}^{22}_{r_t}$ be the normalized clustering matrix obtained from r_t clusters, as described in MATR-CV. Assume the following are true:*

(i) with probability at least $1 - \delta_{under}$, $\max_{r_t<r}\langle \hat{S}^{22}, \hat{X}^{22}_{r_t}\rangle \le \langle \hat{S}^{22}, X^{22}_0\rangle - \epsilon_{under}$;

(ii) with probability at least $1 - \delta_{over}$, $\max_{r<r_t\le r_T}\langle \hat{S}^{22}, \hat{X}^{22}_{r_t}\rangle \le \langle \hat{S}^{22}, X^{22}_0\rangle + \epsilon_{over}$;

(iii) for the true r, with probability at least $1-\delta_{est}$, $\langle \hat{S}^{22}, \hat{X}^{22}_{r}\rangle \ge \langle \hat{S}^{22}, X^{22}_0\rangle - \epsilon_{est}$;

(iv) there exists $\Delta > 0$, such that $\epsilon_{est} + \epsilon_{over} \le \Delta < \epsilon_{under} - \epsilon_{est}$.

Here, $\epsilon_{under}, \epsilon_{est}, \epsilon_{over} > 0$. Then, with probability at least $1 - \delta_{under} - \delta_{over} - \delta_{est}$, MATR-CV will recover the true r with trace gap Δ.

Algorithm 2: MATR-CV.

Input: clustering algorithm $\mathscr{A}$, adjacency matrix A, similarity matrix $\hat{S}$, candidates $\{r_1, \ldots, r_T\}$, number of repetitions J, training ratio γ_{train}, trace gap Δ;
for $j = 1 : J$ **do**
  for $t = 1 : T$ **do**
    $\hat{S}^{11}, \hat{S}^{21}, \hat{S}^{22} \leftarrow \text{NodeSplitting}(\hat{S}, n, \gamma_{\text{train}})$;
    $A^{11}, A^{21}, A^{22} \leftarrow \text{NodeSplitting}(A, n, \gamma_{\text{train}})$;
    $\hat{\Theta}^{11} = \mathscr{A}(A^{11}, r_t)$;
    $\hat{\Theta}^{22} = \text{ClusterTest}(A^{21}, \hat{\Theta}^{11})$;
    $\hat{X}^{22} = \hat{\Theta}^{22}(\hat{\Theta}^{22^T}\hat{\Theta}^{22})^{-1}\hat{\Theta}^{22^T}$;
    $l_{r_t,j} = \langle \hat{S}^{22}, \hat{X}^{22} \rangle$;
  end for
  $r_j^* = \min\{r_t : l_{r_t,j} \geq \max_t l_{r_t,j} - \Delta\}$;
end for
$\hat{r} = \text{median}\{r_j^*\}$
Output: $\hat{r}$

Algorithm 3: NodeSplitting

Input: A, n, γ_{train};
Randomly split $[n]$ into Q_1, Q_2 of size $n\gamma_{\text{train}}$ and $n(1 - \gamma_{\text{train}})$
$A^{11} \leftarrow A_{Q_1,Q_1}$, $A^{21} \leftarrow A_{Q_2,Q_1}$, $A^{22} \leftarrow A_{Q_2,Q_2}$
Output: A^{11}, A^{21}, A^{22}

Algorithm 4: ClusterTest

Input: A^{21}, $\hat{\Theta}^{11}$;
Estimate testing node memberships using $\hat{\Theta}^{11}$ and A^{21}.
Output: $\hat{\Theta}^{22}$

The proof is deferred to the Supplementary Material, Section S2.

Remark 1.

1. MATR-CV is also compatible with tuning multiple hyperparameters. For example, for (3.1), if the number of clusters is unknown, then for each $\hat{r}$, we can run MATR to find the best λ for the given $\hat{r}$, followed by running a second level MATR-CV to find the best $\hat{r}$. As long as the conditions in Theorems 1 and 2 are met, $\hat{r}$ and the clustering matrix returned will be consistent.

2. As shown in the applications below, the derivations of ϵ_{under} and ϵ_{over} are general and only depend on the properties of $\hat{S}$. On the other hand, ϵ_{est} measures the estimation error associated with the algorithm of interest and depends on its performance.

In what follows, we demonstrate MATR-CV can be applied to do model selection for MMSB, which includes SBM as a sub-model.

4.1. Model selection for MMSB

In this section, we consider model selection for the MMSB as introduced in Section 2.2, with a soft membership matrix Θ. As an example of estimation algorithms, we consider the SPACL algorithm proposed by Mao, Sarkar and Chakrabarti (2017), which gives consistent parameter estimation when given the correct r. As mentioned in Section 2.2, a normalized clustering matrix in this case is defined analogously as $X = \Theta(\Theta^T\Theta)^{-1}\Theta^T$ for any Θ. X is still a projection matrix, and $X\mathbf{1}_n = \Theta(\Theta^T\Theta)^{-1}\Theta^T\mathbf{1}_n = \Theta(\Theta^T\Theta)^{-1}\Theta^T\Theta\mathbf{1}_r = \mathbf{1}_n$, since $\Theta\mathbf{1}_r = \mathbf{1}_n$. Following Mao, Sarkar and Chakrabarti (2017), we consider a Bayesian setting for Θ: each row of Θ, $\Theta_i \sim \text{Dirichlet}(\boldsymbol{\alpha}), \boldsymbol{\alpha} \in \mathbb{R}_+^r$. We assume r, $\boldsymbol{\alpha}$ are all fixed constants. Note that the Bayesian setting here is only for convenience, and can be replaced with equivalent assumptions bounding the eigenvalues of $\Theta^T\Theta$. We also assume there is at least one pure node for each of the r communities for consistent estimation at the correct r.

MATR-CV can be applied to the MMSB model by noting the following two points. First, take $\hat{S} = A^2 - \text{diag}(A^2)$ and $S = P^2 - \text{diag}(P^2)$. This allows us to remove the assortativity requirement on P and replace it with a full rank condition on B, which is commonly assumed in the MMSB literature. The fact that P^2 is always positive semi-definite is used in the proof. The removal of $\text{diag}(A^2)$ and $\text{diag}(P^2)$ leads to better concentration, because $\text{diag}(A^2)$ is centered around a different mean. Second, noting that $P^{12} = \Theta^{11}B(\Theta^{22})^T$, we can view the estimation of Θ^{22} as a regression problem with plug-in estimators of Θ^{11} and B. In Algorithm 4, we use an estimate of the form $\hat{\Theta}^{22} = A^{21}\hat{\Theta}^{11}((\hat{\Theta}^{11})^T\hat{\Theta}^{11})^{-1}\hat{B}^{-1}$, where $\hat{B}$ and $\hat{\Theta}^{11}$ are estimated from A^{11}.

We have the following guarantee for $\hat{r}$ returned by MATR-CV.

Theorem 3. *Let A be generated from an MMSB (see Section 2.2) satisfying $\lambda^*(B) = \Omega(\rho)$, where $\lambda^*(B)$ is the smallest singular value of B. We assume $\sqrt{n\rho}/(\log n)^{1+\xi} \to \infty$, for some arbitrarily small $\xi > 0$. Given a candidate set of $\{r_1, \ldots, r_T\}$ containing r and $r_T = \Theta(1)$, with high probability for large n, MATR-CV returns the true cluster number r if $\Delta = O((n\rho)^{3/2}(\log n)^{1.01})$.*

Proof sketch. We first show w.h.p., the underfitting and overfitting errors in Theorem 2 are $\epsilon_{\text{under}} = \Omega(n^2\rho^2)$ and $\epsilon_{\text{over}} = O(n\rho\sqrt{\log n})$, respectively. To obtain ϵ_{est}, we show that given the true cluster number, the convergence rate of the parameter estimates for the testing nodes obtained from the regression algorithm is the same as the convergence rate for the training nodes. This leads to $\epsilon_{\text{est}} = O((n\rho)^{3/2}(\log n)^{1+\xi})$. For convenience, we pick $\xi = 0.01$. For details, see Section S2.2 of the Supplementary Material.

Remark 2.

1. The new choice of S and $\hat{S}$ allows our framework to work for more general B, which can have negative eigenvalues in Theorem 3. If B is positive semi-definite with full rank, a common assumption in many MMSB papers, we can still use A and P as $\hat{S}$ and S, respectively. A similar analysis applies, and the same type of consistency result holds.

2. Compared with Fan et al. (2022), who consider the more general degree-corrected MMSB model, our result holds for $\rho \to 0$ at a faster rate.

3. On a practical note: due to the constant in the estimation error being tedious to determine, in this case we only know the asymptotic order of the gap Δ. As has been observed in many other methods based on asymptotic properties (e.g., Bickel and Sarkar (2016); Lei (2016); Wang and Bickel (2017); Hu et al. (2017)), performing an adjustment for finite samples often improves the empirical performance. In practice, we find that if the constant factor in Δ is too large, we tend to underfit. To guard against this, note that at the correct r, the trace difference $\delta_{r,r-1} := \langle \hat{S}, \hat{X}_r \rangle - \langle \hat{S}, \hat{X}_{r-1} \rangle$ should be much larger than Δ. We start with $\Delta = (n\rho)^{3/2}(\log n)^{1.01}$ and find $\hat{r}$ by Algorithm 2; if $\delta_{\hat{r},\hat{r}-1}$ is smaller than Δ, we reduce Δ by half and repeat the step of finding r_j^* in Algorithm 2 until $\delta_{\hat{r},\hat{r}-1} > \Delta$. This adjustment is much faster than bootstrap corrections and works well empirically.

4. As an example of applying Algorithm 2 to the SBM, we consider a different type of SDP algorithm introduced in Peng and Wei (2007) and Yan, Sarkar and Cheng (2017), where the model selection problem is embedded in the algorithm as a hyperparameter tuning problem. In this case, $\hat{S}$ is simply A itself, and the estimation error ϵ_{est} can achieve zero. The detailed statement and proofs can be found in Section S2.3 of the Supplementary Material.

5. Numerical Experiments

In this section, we present extensive numerical results on simulated and real data by applying MATR and MATR-CV to the settings considered in Sections 3 and 4.

5.1. MATR with known number of clusters

Spectral clustering for mixture models. We use MATR-CV to select the bandwidth parameter θ in spectral clustering applied to mixture data, when given the correct number of clusters. In all the examples, our candidate set of θ is $\{t\alpha/20\}$, for $t = 1, \ldots, 20$ and $\alpha = \max_{i,j} \|Y_i - Y_j\|_2$. We compare MATR with three other well-known heuristic methods. The first one was proposed by Shi, Belkin and Yu (2008) (DS), where, for each data point Y_i, the 5%

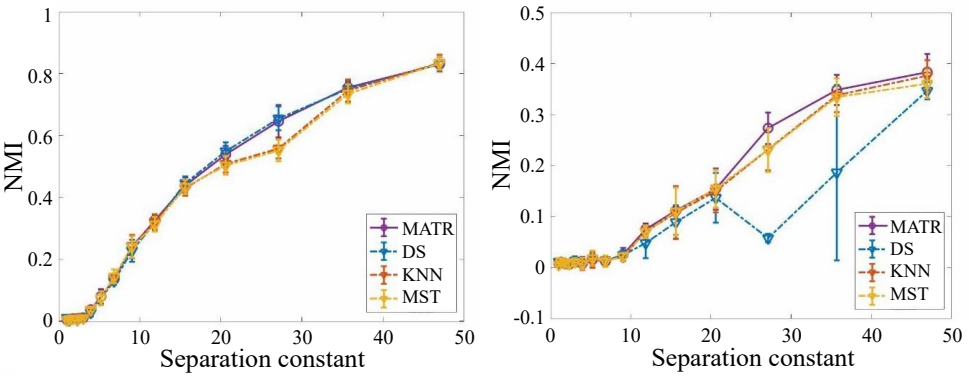
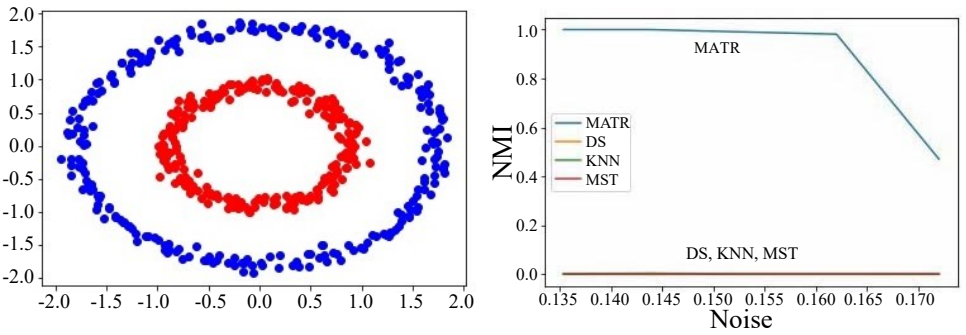


Figure 2. Comparison of NMI for tuning the bandwidth in spectral clustering for mixture models with (a) equal and (b) unequal mixing coefficients.

quantile of $\{\|Y_i - Y_j\|_2, j = 1, \ldots, n\}$ is denoted q_i, and then θ is set to be 95% quantile of $\{q_1, \ldots, q_n\}/\sqrt{95\% \text{ quantile of } \chi^2_d}$. The other two methods are presented in von Luxburg (2007): a method based on k-nearest neighbor (KNN) and a method based on minimal spanned tree (MST). For KNN, θ is chosen in the order of the mean distance of a point to its k-th nearest neighbor, where $k \sim \log(n) + 1$. For MST, θ is set as the length of the longest edge in a minimal spanning tree of the fully connected graph on the data points.

Simulated data. We first conduct experiments on simulated data generated from a three-component Gaussian mixture with $d = 20$. The means are multiplied by a separation constant that controls the clustering difficulty (a larger constant implies less difficulty). Detailed descriptions of the parameter settings can be found in Section S3.2 of the Supplementary Material. $n = 500$ data points are generated for each mixture model, and random runs are used to calculate the standard deviations for each parameter setting. Figures 2(a) and (b) show the NMI of different methods against the separation constant for equal and unequal mixing proportions, respectively. For all these settings, MATR performs the best or comparably to DS, KNN and MST.

To illustrate the robustness of our method on non-Gaussian data, we also apply MATR to tune the bandwidth θ for the two rings data set (Figure 3(a)) by setting the similarity matrix $\hat{S}$ to be an RBF kernel to account for nonlinearity. To alleviate the problem that the trace objective is now also dependent on θ via $\hat{S}$, we use a rough guess, for example, 10th percentile of the pairwise distances in $\hat{S}$. A rough guess here is enough to pick up the right trend. We then apply MATR to select θ in spectral clustering. As seen in Figure 3(b), MATR outperforms the other methods by a large margin.



(a) The two-ring data set (b) NMI comparison

Figure 3. Results on the ring data set.

Real data. We also test MATR for tuning θ on a real data set, namely, the Olivetti faces data set, provided by scikit-learn (Pedregosa et al. (2011)). The data consist of 40 classes with 10 examples in each class. We standardize the dataset before clustering. MATR achieves the highest NMI value of 0.83. Both KNN and MST obtain NMI values around 0.82, while DS yields a θ much smaller than those of the other methods, leading to similarity matrices that are highly unstable when spectral clustering is applied.

Additional results for SBM. We apply MATR to tune λ in (3.1) for known r and compare with two existing data driven methods (Cai and Li (2015) and Li, Levina and Zhu (2020)) using simulated and real networks. The details can be found in Section S3.3 of the Supplementary Material.

5.2. Model selection with MATR-CV

MMSB. We compare MATR-CV with universal singular value thresholding (USVT) (Chatterjee (2015)), ECV (Li, Levina and Zhu (2020)) and SIMPLE (Fan et al. (2022)) in terms of model selection with MMSB. For ECV and MATR-CV, we consider the candidate set $r \in \{1, 2, \ldots, \lfloor \hat{\rho} n \rfloor\}$, where $\hat{\rho} = \sum_{i<j} A_{ij} / \binom{n}{2}$.

Simulated data. We first apply all methods to simulated data. We consider $B = \rho \times \{(p-q)I_r + qE_r\}$. Following (Mao, Sarkar and Chakrabarti (2018)), we sample $\Theta_i \sim \text{Dirichlet}(\boldsymbol{\alpha})$ and $\boldsymbol{\alpha} = \mathbf{1}_r/r$. We generate networks with $n = 2000$ nodes with $r = 4$ and $r = 8$. We set $p = 1, q = 0.1$ for $r = 4$ and $p = 1, q = 0.01$ for $r = 8$ for a range of ρ. In Tables 1a and 1b, we report the fractions of exactly recovering the true cluster number r over 40 runs for each method across different average degrees. We observe that for both $r = 4$ and $r = 8$, MATR-CV outperforms the other three methods by a large margin on sparse graphs. We find that SIMPLE tends to underfit in our sparsity regime, since their theoretical guarantees hold for a regime with a denser degree in order to generalize to a broader model than the MMSB. An example generated from a non-assortative B

	ρ	0.01	0.02	0.06	0.08	0.11	0.13
	MATR-CV	0.35	0.83	0.93	1	1	1
(a)	USVT	0	0	1	1	1	1
	ECV	0	0	1	0.95	1	1
	ρ	0.02	0.05	0.09	0.12	0.16	0.21
	MATR-CV	0.10	0.43	0.95	0.93	0.95	1
(b)	USVT	0	0	0.58	1	1	1
	ECV	0	0	0	0.93	1	1

Table 1. Model selection on simulated MMSB. Exact recovery fractions for (a) 4 clusters; (b) 8 clusters.

can be found in the Supplementary Material, Section S3.4.

Real data. We also test MATR-CV with the MMSB on a real network, the political books network, which contains 105 nodes in three clusters. Here, fitting a MMSB model is reasonable since each book can have mixed political inclinations, for example, a "conservative" book may actually be a mix of neutral and conservative views. Using MATR-CV, we found three clusters, agreeing with the ground truth. USVT, ECV and SIMPLE found fewer than three clusters.

Additional results for SBM. We apply MATR-CV to tune the SDP in Yan, Sarkar and Cheng (2017) for model selection. Comparisons with existing methods on simulated and real networks can be found in the Supplementary Material, Section S3.5.

6. Discussion

Clustering data, both in i.i.d. and network structured settings, have received a lot of attention both from applied and theoretical communities. However, methods for tuning hyperparameters involved in clustering problems are mostly heuristic. In this paper, we present MATR, a provable MAx-TRace based hyperparameter tuning framework for general clustering problems. We prove the effectiveness of this framework for tuning SDP relaxations for community detection under the block model, and for learning the bandwidth parameter of the Gaussian kernel in spectral clustering over a mixture of sub-Gaussians. Our framework can also be used to perform model selection using a CV-based extension (MATR-CV) to consistently estimate the number of clusters in the SBM and the MMSB. Using a variety of simulation and real experiments, we have shown the advantage of our method over existing heuristics.

The framework presented in this paper is general, and can be applied to model selection or tuning for broader model classes such as the degree-corrected block models (Karrer and Newman (2011)), since there are many exact recovery-based algorithms for estimation in these settings (Chen, Li and Xu (2018)). We believe

that our framework can be extended to the broader class of degree-corrected mixed membership block models (Jin, Ke and Luo (2017)), which includes the topic model (Mao, Sarkar and Chakrabarti (2018)). However, the derivation of the estimation error ϵ_{est} involves tedious derivations of the parameter estimation error, which has not been done by existing works. Furthermore, even though our work uses node sampling, we believe we can extend the MATR-CV framework to obtain consistent model selection for other sampling procedures, such as edge sampling (Li, Levina and Zhu (2020)).

Supplementary Material

The online Supplementary Material contains detailed proofs of the main results, together with additional theoretical and numerical results.

Acknowledgments

Wang was partially supported by the ARC DECRA fellowship DE180101252. Sakar was partially supported by NSF DMS 1713082. The authors would like to thank the editor, the associate editor and two anonymous referees for their valuable time and comments.

References

Abbe, E., Bandeira, A. S. and Hall, G. (2015). Exact recovery in the stochastic block model. *IEEE Transactions on Information Theory* **62**, 471–487.

Abbe, E. and Sandon, C. (2015). Recovering communities in the general stochastic block model without knowing the parameters. In *Proceedings of the 28th International Conference on Neural Information Processing Systems* **1**, 676–684.

Airoldi, E. M., Blei, D. M., Fienberg, S. E. and Xing, E. P. (2008). Mixed membership stochastic blockmodels. *J. Mach. Learn. Res.* **9**, 1981–2014.

Amini, A. A. and Levina, E. (2018). On semidefinite relaxations for the block model. *Ann. Statist.* **46**, 149–179.

Athreya, A., Fishkind, D. E., Tang, M., Priebe, C. E., Park, Y., Vogelstein, J. T. et al. (2017). Statistical inference on random dot product graphs: A survey. *J. Mach. Learn. Res.* **18**, 8393–8484.

Bach, F. R. (2008). Bolasso: Model consistent Lasso estimation through the bootstrap. In *Proceedings of the 25th International Conference on Machine Learning*, 33–40. ACM.

Belkin, M. and Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comput.* **15**, 1373–1396.

Bengio, Y. (2000). Gradient-based optimization of hyperparameters. *Neural Comput.* **12**, 1889–1900.

Bergstra, J. and Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research* **13**, 281–305.

Bergstra, J. S., Bardenet, R., Bengio, Y. and Kégl, B. (2011). Algorithms for hyper-parameter optimization. In *Advances in Neural Information Porcessing System* **24**, 2546–2554.

Bickel, P. J. and Sarkar, P. (2016). Hypothesis testing for automated community detection in networks. *J. R. Stat. Soc. Ser. B. Stat. Methodol.* **78**, 253–273.

Birgé, L. and Massart, P. (2001). Gaussian model selection. *Journal of the European Mathematical Society* **3**, 203–268.

Bozdogan, H. (1987). Model selection and Akaike's information criterion (AIC): The general theory and its analytical extensions. *Psychometrika* **52**, 345–370.

Breiman, L. (1996). Heuristics of instability and stabilization in model selection. *Ann. Statist.* **24**, 2350–2383.

Cai, T. T. and Li, X. (2015). Robust and computationally feasible community detection in the presence of arbitrary outlier nodes. *Ann. Statist.* **43**, 1027–1059.

Chatterjee, S. (2015). Matrix estimation by universal singular value thresholding. *Ann. Statist.* **43**, 177–214.

Chen, K. and Lei, J. (2018). Network cross-validation for determining the number of communities in network data. *J. Amer. Statist. Assoc.* **113**, 241–251.

Chen, Y., Li, X. and Xu, J. (2018). Convexified modularity maximization for degree-corrected stochastic block models. *Ann. Statist.* **46**, 1573–1602.

Coifman, R. R., Shkolnisky, Y., Sigworth, F. J. and Singer, A. (2008). Graph Laplacian tomography from unknown random projections. *Trans. Img. Proc.* **17**, 1891–1899.

Drton, M. and Plummer, M. (2017). A Bayesian information criterion for singular models. *J. R. Stat. Soc. Ser. B. Stat. Methodol.* **79**, 323–380.

Fan, J., Fan, Y., Han, X. and Lv, J. (2022). Simple: Statistical inference on membership profiles in large networks. *J. R. Stat. Soc. Ser. B. Stat. Methodol.* **84**, 630–653.

Fan, X., Yue, Y., Sarkar, P. and Wang, Y. X. R. (2020). On hyperparameter tuning in general clustering problems. In *Proceedings of the 37th International Conference on Machine Learning* **Article 281**, 2996–3007.

Fang, Y. and Wang, J. (2012). Selection of the number of clusters via the bootstrap method. *Computational Statistics & Data Analysis* **56**, 468–477.

Feng, J. and Simon, N. (2020). An analysis of the cost of hyperparameter selection via split-sample validation, with applications to penalized regression. *Statist. Sinica* **30**, 511–530.

Figueiredo, M. A. T. and Jain, A. K. (2002). Unsupervised learning of finite mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**, 381–396.

Giné, E. and Koltchinskii, V. (2006). Empirical graph Laplacian approximation of Laplace–Beltrami operators: Large sample results. *Lecture Notes–Monograph Series*, 238–259. IMS.

Guédon, O. and Vershynin, R. (2016). Community detection in sparse networks via Grothendieck's inequality. *Probability Theory and Related Fields* **165**, 1025–1049.

Hajek, B., Wu, Y. and Xu, J. (2016). Achieving exact cluster recovery threshold via semidefinite programming. *IEEE Transactions on Information Theory* **62**, 2788–2797.

Hastie, T., Tibshirani, R. and Friedman, J.(2001). *The Elements of Statistical Learning.* Springer, New York.

Hein, M. (2006). Uniform convergence of adaptive graph-based regularization. In *COLT 2006: Learning Theory*, 50–64. Springer, Berlin, Heidelberg.

Hein, M., Audibert, J.-Y. and von Luxburg, U. (2005). From graphs to manifolds – weak and strong pointwise consistency of graph Laplacians. In *COLT 2005: Learning Theory*, 470–485. Springer, Berlin, Heidelberg.

Hu, J., Qin, H., Yan, T., Zhang, J. and Zhu, J. (2017). Mixedentry-wise deviation to test the goodness-of-fit for stochastic block models. *arXiv:1703.06558*.

Jin, J., Ke, Z. T. and Luo, S. (2017). Mixed memberships estimation for social networks. *arXiv:1708.07852*.

Karrer, B. and Newman, M. E. J. (2011). Stochastic blockmodels and community structure in networks. *Phys. Rev. E* **83**, 016107.

Keribin, C. (2000). Consistent estimate of the order of mixture models. *Sankhy, Series A* **62**, 49–66.

Le, C. M. and Levina, E. (2015). Estimating the number of communities in networks by spectral methods. *arXiv:1507.00827*.

Lei, J. (2016). A goodness-of-fit test for stochastic block models. *Ann. Statist.* **44**, 401–424.

Leroux, B. G. (1992). Consistent estimation of a mixing distribution. *Ann. Statist.* **20**, 1350–1360.

Löffler, M., Zhang, A. Y. and Zhou, H. H. (2019). Optimality of spectral clustering in the Gaussian mixture model. arXiv:1911.00538.

Li, T., Levina, E. and Zhu, J. (2020). Network cross-validation by edge sampling. *Biometrika* **107**, 257–276.

Li, X., Chen, Y. and Xu, J. (2018). Convex relaxation methods for community detection. *arXiv:1810.00315*.

Lim, C. and Yu, B. (2016). Estimation stability with cross-validation (ESCV). *Journal of Computational and Graphical Statistics* **25**, 464–492.

Little, A., Maggioni, M. and Murphy, J. M. (2017). Path-based spectral clustering: Guarantees, robustness to outliers, and fast algorithms. *Journal of Machine Learning Research* **21**, 1–66.

Maggioni, M. and Murphy, J. M. (2018). Learning by unsupervised nonlinear diffusion. *arXiv:1810.06702*.

Mao, X., Sarkar, P. and Chakrabarti, D. (2017). Estimating mixed memberships with sharp eigenvector deviations. *arXiv:1709.00407*.

Mao, X., Sarkar, P. and Chakrabarti, D. (2018). Overlapping clustering models, and one (class) SVM to bind them all. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 2130–2140.

Meila, M. (2018). How to tell when a clustering is (approximately) correct using convex relaxations. In *Advances in Neural Information Processing Systems*, 7407–7418.

Meinshausen, N. and Bühlmann, P. (2010). Stability selection. *J. R. Stat. Soc. Ser. B. Stat. Methodol.* **72**, 417–473.

Mixon, D. G., Villar, S. and Ward, R. (2017). Clustering subgaussian mixtures by semidefinite programming. *Information and Inference: A Journal of the IMA* **6**, 389–415.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O. et al. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* **12**, 2825–2830.

Peng, J. and Wei, Y. (2007). Approximating k-means-type clustering via semidefinite programming. *SIAM J. on Optimization* **18**, 186–205.

Perry, A. and Wein, A. S. (2017). A semidefinite program for unbalanced multisection in the stochastic block model. In *2017 International Conference on Sampling Theory and Applications (SampTA)*, 64–67. IEEE.

Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics* **20**, 53–65.

Schiebinger, G., Wainwright, M. J. and Yu, B. (2015). The geometry of kernelized spectral clustering. *Ann. Statist.* **43**, 819–846.

Shao, J. (1993). Linear model selection by cross-validation. *J. Amer. Statist. Assoc.* **88**, 486–494.

Shi, T., Belkin, M. and Yu, B. (2008). Data spectroscopy: Learning mixture models using eigenspaces of convolution operators. In *Proceedings of the 25th International Conference on Machine Learning*, 936–943. ACM.

Snoek, J., Larochelle, H. and Adams, R. P. (2012). Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*, 2951–2959.

Srivastava, P. R., Sarkar, P. and Hanasusanto, G. A. (2023). A robust spectral clustering algorithm for sub-Gaussian mixture models with outliers. *Operations Research* **71**, 224–244.

Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions. *J. R. Stat. Soc. Ser. B. Stat. Methodol.* **36**, 111–133.

Tibshirani, R., Wainwright, M. and Hastie, T. (2015). *Statistical Learning with Sparsity: The Lasso and Generalizations*. Chapman and Hall/CRC.

Tibshirani, R., Walther, G. and Hastie, T. (2001). Estimating the number of clusters in a data set via the gap statistic. *J. R. Stat. Soc. Ser. B. Stat. Methodol.* **63**, 411–423.

von Luxburg, U. (2007). A tutorial on spectral clustering. *Stat. & Comput.* **17**, 395–416.

von Luxburg, U. (2010). Clustering stability: An overview. *Foundations and Trends® in Machine Learning* **2**, 235–274.

Wang, J. (2010). Consistent selection of the number of clusters via crossvalidation. *Biometrika* **97**, 893–904.

Wang, Y. X. R. and Bickel, P. J. (2017). Likelihood-based model selection for stochastic block models. *Ann. Statist.* **45**, 500–528.

Wasserman, L. (2006). *All of Nonparametric Statistics*. Springer Science & Business Media.

Wasserman, L. and Roeder, K. (2009). High dimensional variable selection. *Ann. Statist.* **37**, 2178.

Yan, B. and Sarkar, P. (2021). Covariate regularized community detection in sparse graphs. *J. Amer. Statist. Assoc.* **116**, 734–745.

Yan, B., Sarkar, P. and Cheng, X. (2017). Provable estimation of the number of blocks in block models. *arXiv:1705.08580*.

Yang, Y. (2007). Consistency of cross validation for comparing regression procedures. *Ann. Statist.* **35**, 2450–2473.

Young, S. J. and Scheinerman, E. R. (2007). Random dot product graph models for social networks. In *International Workshop on Algorithms and Models for the Web-Graph*, 138–149. Springer.

Zhang, P. (1993). Model selection via multifold cross validation. *Ann. Statist.* **21**, 299–313.

Xinjie Fan

Department of Statistics and Data Sciences, The University of Texas at Austin, Austin, TX 78705, USA.

E-mail: xfan@utexas.edu

Y. X. Rachel Wang

School of Mathematics and Statistics, University of Sydney, NSW 2006, Australia.

E-mail: rachel.wang@sydney.edu.au

Purnamrita Sarkar

Department of Statistics and Data Sciences, The University of Texas at Austin, Austin, TX 78705, USA.

E-mail: purna.sarkar@austin.utexas.edu

Yuguang Yue
Department of Statistics and Data Sciences, The University of Texas at Austin, Austin, TX 78705, USA.
E-mail: yuguang@utexas.edu

(Received January 2022; accepted September 2022)