Biometrika (2020), **107**, 2, *pp.* 277–280
Printed in Great Britain

doi: 10.1093/biomet/asaa017

Discussion of 'Network cross-validation by edge sampling'

By JINYUAN CHANG

School of Statistics, Southwestern University of Finance and Economics, 555 Liutai Avenue, Chengdu, Sichuan 611130, China
changjinyuan@swufe.edu.cn

ERIC D. KOLACZYK

Department of Mathematics and Statistics, Boston University, 111 Cummington Mall, Boston, Massachusetts 02215, U.S.A.
kolaczyk@bu.edu

and QIWEI YAO

Department of Statistics, London School of Economics, Houghton Street, London WC2A 2AE, U.K.
q.yao@lse.ac.uk

1. Edge cross-validation for network model selection

We thank the authors for their new contribution to network modelling. Data reuse, encompassing methods such as bootstrapping and cross-validation, is an area that to date has largely resisted obvious and rapid development in the network context. One of the major reasons is that mimicking the original sampling mechanisms is challenging if not impossible. To avoid deleting edges and destroying some of the network structure, the resampling strategy proposed in Li et al. (2020) based on splitting node pairs rather than nodes is therefore insightful and effective. Matrix completion is the key technique involved, with its use here providing a new perspective for network analysis.

The proposed edge cross-validation procedure operates effectively on an adjacency matrix $A = (A_{ij})_{n\times n}$ instead of on the original network, as described in the following algorithm.

Algorithm 1. The general edge cross-validation procedure.

```
Step 1. Give a loss function $L$ and select the rank $\hat{K}$ for matrix completion.
Step 2. For $m = 1$ to $m = N$:
        (a) Assign to each $(i,j)$ in the learning set $\Omega$ a prespecified probability $p$.
        (b) Obtain $\hat{A}$ from $(A,\Omega)$ by a low-rank matrix completion algorithm with rank $\hat{K}$.
        (c) For each candidate model $q = 1,\ldots,Q$, fit the model on data $\hat{A}$ and evaluate
            its loss $L_q^{(m)}$ by comparing the resulting estimated parameters over the held-out
            set $\{A_{ij} : (i,j) \notin \Omega\}$.
Step 3. Let $L_q = N^{-1}\sum_{m=1}^{N} L_q^{(m)}$ and select the candidate model $\hat{q} = \arg\min_{1\leqslant q\leqslant Q} L_q$.
```

The authors proposed using $p = 0.9$ and replicated the validation $N = 3$ times. Therefore only about 30% of the edges are used for cross-validation. Borrowing from the noisy network setting of Chang et al. (2018), the proposal below will use all of the edges for validation. It is in the spirit of jittered bootstrapping or resampling via jittering. Here jittering means that a small amount of noise is added to every single data point; see, for example, Henning (2007, § 3.3). Interestingly, the low-rank assumption does not appear to be necessary in this approach.

2. Jittering algorithm for network model selection

Let ε_{ij}, for $i \neq j$, be independent and such that

$$\text{pr}(\varepsilon_{ij} = 1) = \alpha, \quad \text{pr}(\varepsilon_{ij} = 0) = 1 - \alpha - \beta, \quad \text{pr}(\varepsilon_{ij} = -1) = \beta, \tag{1}$$

where α and β are two small positive constants. Recall that $M = (M_{ij})_{n\times n}$ with $M_{ij} = \text{pr}(A_{ij} = 1)$. The jittering algorithm is summarized as follows.

```
Algorithm 2. The general jittering algorithm.
Step 1. Give a loss function $L$.
Step 2. For $m = 1$ to $m = N$:
        (a) Draw a jittered sample $Y = (Y_{ij})_{n\times n}$ with $Y_{ij} = A_{ij} I(\varepsilon_{ij} = 0) + I(\varepsilon_{ij} = 1)$.
        (b) For each candidate model $q = 1, \ldots, Q$, estimate the parameters in $M$ based on
            noisy data $Y$, and evaluate its loss $L_q^{(m)}$ by comparing the resulting estimated
            parameters with $A$.
Step 3. Let $L_q = N^{-1} \sum_{m=1}^{N} L_q^{(m)}$ and select the candidate model $\hat{q} = \arg\min_{1\leqslant q\leqslant Q} L_q$.
```

The estimation of parameters in M in Step 2 of Algorithm 2 requires different treatments for different network models. Chang et al. (2018) illustrate how to conduct statistical inference for subgraph densities based on noisy data. A fundamental difference here is that both α and β are known in the present context. Therefore we can use, at least partially, the debiased data

$$A^\star = \frac{Y - \alpha}{1 - \alpha - \beta} \tag{2}$$

to fit a network model. It follows from $Y_{ij} = A_{ij} I(\varepsilon_{ij} = 0) + I(\varepsilon_{ij} = 1)$ that $E(Y) = E\{E(Y \mid A)\} = E\{A(1 - \alpha - \beta) + \alpha\} = M(1 - \alpha - \beta) + \alpha$.

As an illustration, we consider the stochastic block model $M = ZBZ^{\mathrm{T}}$, using the notation of Li et al. (2020), where B is a $K \times K$ probability matrix. The goal is to determine the number of communities K using the proposed jittering algorithm. Following the lead of Li et al. (2020), we apply the spectral clustering method, but with the jittered data $Y = (Y_{ij})_{n\times n}$, i.e., performing the eigenanalysis on the Laplacian $L = D^{-1/2} Y D^{-1/2}$, where D is the diagonal matrix with the node degrees of Y as its main diagonal elements. Let $K_m > K$ be a prespecified integer. For each $k = 2, \ldots, K_m$, the k-means algorithm is applied to the n rows of the $n \times k$ matrix consisting of the k orthonormal eigenvectors of L associated with its k largest eigenvalues, leading to the estimated memberships for the n nodes. A $k \times k$ estimated probability matrix $\hat{B}_k = (\hat{B}_{ij}^k)_{k\times k}$ is then obtained using the debiased data $A^\star$ in (2). The loss is measured by the mean squared error

$$W(k) = \left\{ \sum_{1\leqslant i<j\leqslant n} (A_{ij} - \hat{B}_{\hat{c}_i \hat{c}_j}^k)^2 \right\}^{1/2},$$

where $\hat{c}_i$ is the estimated membership for node i. Repeating the jittered sampling N times yields $W_1(k), \ldots, W_N(k)$. The estimated number of communities is then defined as

$$\hat{K} = \arg\min_{1<k\leqslant K_m} \frac{1}{N} \sum_{j=1}^{N} W_j(k).$$

Table 1. *Relative frequencies for events* ($\hat{K} = k$) *with* $k = K - 1$, K *and* $K + 1$ *in a simulation with* 200 *replications*

n	K	π_{in}	π_{bg}	EVD	$(\hat{K} = K - 1)$	$(\hat{K} = K)$	$(\hat{K} = K + 1)$
60	3	0.6	0.06	13.8	0	1	0
			0.12	16.2	0	1	0
			0.30	23.4	0.760	0.240	0
	5		0.06	9.48	0	0.995	0.005
			0.12	12.36	0	0.750	0.155
			0.30	21	0.010	0.005	0.005
300	3	0.6	0.06	71.4	0	1	0
			0.12	83.4	0	1	0
			0.30	119.4	0.795	0.205	0
	5		0.06	49.8	0	1	0
			0.12	64.2	0	1	0
			0.30	107.4	0	0	0
600	3	0.6	0.06	143.4	0	1	0
			0.12	167.4	0	1	0
			0.30	239.4	0.805	0.195	0
	5		0.06	100.2	0	1	0
			0.12	129	0	1	0
			0.30	215.4	0	0	0
300	3	0.1	0.02	13.9	0	1	0
	5			10.7	0	0	0
600	3			27.9	0	1	0
	5			21.6	0	1	0

EVD, average expected node degree.

3. A numerical study

Although we are still working on a formal theoretical justification for this procedure, results from simulations are positive. We report some illustrative results in Table 1. We take all the main diagonal elements of B to equal π_{in} and all the off-diagonal elements to equal π_{bg}, where $0 < \pi_{\text{bg}} < \pi_{\text{in}} < 1$. We set $\alpha = \beta = 0.05$ in (1) and repeat the jittered sampling $N = 200$ times. For each setting we repeat the estimation 200 times. The relative frequencies of the events $\hat{K} = K$, $\hat{K} = K - 1$ and $\hat{K} = K + 1$ in the 200 replications are reported in Table 1. Also included are the average expected node degrees, EVD, which reflect the sparseness of the models. Li et al. (2020) used β to denote EVD in their paper, which differs from the usage here.

With a within-block probability of $\pi_{\text{in}} = 0.6$ and a between-block probability of $\pi_{\text{bg}} = 0.06$ or 0.12, the community number K can be determined correctly with little error. Minor inefficiency only occurs when $n = 10$, $\pi_{\text{bg}} = 0.12$ and $K = 5$. The results hardly change when α and β are increased from 0.05 to 0.1. However, when $\pi_{\text{bg}} = 0.3$, $\hat{K}$ almost always underestimates K. When the out-in ratio $\pi_{\text{bg}}/\pi_{\text{in}}$ is 0.5, the spectral clustering algorithm has difficulties identifying different communities.

We now consider sparse models with $\pi_{\text{in}} = 0.1$ and $\pi_{\text{bg}} = 0.02$, for which the out-in ratio is $\pi_{\text{bg}}/\pi_{\text{in}} = 0.2$. The results are reported in the last section of Table 1. The networks are very sparse now. For example, with $n = 300$ and $K = 3$ the EVD is merely 13.9. Still, the proposed method leads to 100% correctness in determining K, which is also the case for $n = 600$ and $K = 3$ or 5. However, the method fails completely when $n = 300$ and $K = 5$; the EVD is now 10.7, and it seems that the signal for communities is too weak to be picked up for reasonable clustering. Using smaller values for α and β, such as 0.025 or 0.01, does not help either.

Acknowledgement

Chang was supported in part by the Fundamental Research Funds for the Central Universities of China, the National Natural Science Foundation of China, the Fok Ying-Tong Education Foundation, and the Center of Statistical Research and the Joint Lab of Data Science and Business Intelligence at Southwestern University of Finance and Economics. Kolaczyk was supported in part by the U.S. Army Research Office.

References

Chang, J., Kolaczyk, E. D. & Yao, Q. (2018). Estimation of subgraph densities in noisy networks. *arXiv:* 1803.02488v2.

Henning, C. (2007). Cluster-wise assessment of cluster stability. *Comp. Statist. Data Anal.* **52**, 258–71.

Li, T., Levina, E. & Zhu, J. (2020). Network cross-validation by edge sampling. *Biometrika*, **107**, 257–76.

[*Received on* 11 *February* 2020. *Editorial decision on* 12 *February* 2020]