Valid Bootstraps for Network Embeddings with Applications to Network Visualisation

Emerald Dilworth[1] **Ed Davis**[1] **Daniel J. Lawson**[1]

[1]School of Mathematics, University of Bristol, Bristol, UK

Abstract

Quantifying uncertainty in networks is an important step in modelling relationships and interactions between entities. We consider the challenge of bootstrapping an inhomogeneous random graph when only a single observation of the network is made and the underlying data generating function is unknown. We address this problem by considering embeddings of the observed and bootstrapped network that are statistically indistinguishable. We utilise an exchangeable network test that can empirically validate bootstrap samples generated by any method. Existing methods fail this test, so we propose a principled, distribution-free network bootstrap using k-nearest neighbour smoothing, that can pass this exchangeable network test in many synthetic and real-data scenarios. We demonstrate the utility of this work in combination with the popular data visualisation method t-SNE, where uncertainty estimates from bootstrapping are used to explain whether visible structures represent real statistically sound structures.

1 INTRODUCTION

Networks are ubiquitous across many applications including cyber-security [Bowman and Huang, 2021, He et al., 2022, Bilot et al., 2023], biology [Rosenthal et al., 2018, Jumper et al., 2021], natural language [Mikolov et al., 2013], and recommendation algorithms [Wu et al., 2022]. Despite the influence of this field, uncertainty quantification for networks remains a challenge.

The bootstrap method introduced by Efron [1979] provides an estimator for the mean and can be adapted to estimate variance, order statistics, or any other functional of the data for i.i.d. samples [Efron and Tibshirani, 1994]. With appropriate quantile corrections, these estimates are provably correct. Handling dependency requires care [Kreiss and Paparoditis, 2011] and so networks, for which edges are far from independent, require special treatment. Many networks can be described as following a limiting distribution [Lovász and Szegedy, 2006, Borgs et al., 2008], where edges are independent given the node cluster labels. Green and Shalizi [2022] apply this to bootstrapping networks using a graphon-based 'histogram bootstrap'. This approach is further developed by Zu and Qin [2024] for valid local network statistics.

More generally, a Random Dot Product Graph (RDPG) places nodes in a latent vector space and edges become independent conditional on location, which generates provably correct bootstraps [Levin and Levina, 2019]. Adjacency Spectral Embedding (ASE) [Hoff et al., 2002, Sussman et al., 2012] provides strong theoretical asymptotic guarantees [Cape et al., 2019]. These can be generalised to multipartite networks [Rubin-Delanchy et al., 2022]. However, inference for these methods relies on ASE, which empirically has lower power relative to other available embedding methods [Qin and Rohe, 2013, Grover and Leskovec, 2016]. Nearest-neighbour methods [Stone, 1977] also provide asymptotic convergence to a latent position [Lian, 2011] and can be applied to nonlinear embeddings such as Node2Vec [Grover and Leskovec, 2016] and ProNE [Zhang et al., 2019].

Further, we care about finite-sample properties. To address this, we apply developments in 'Unfolding' [De Lathauwer et al., 2000, Gallagher et al., 2021] to construct a Bootstrap Exchangeability Test (Section 2.3) for whether the embeddings of a bootstrapped network and the observed network are exchangeable, in a finite dataset. This reveals that the nonlinear models outperform linear ones and are needed for practical usage.

Previous work uses count statistics and U-statistics to summarise the performance of network bootstraps [Bhattacharyya and Bickel, 2015, Epskamp et al., 2018, Levin and Levina, 2019]. These approaches test that the distribution of local features of networks are conserved under bootstrap, e.g.

the counts of triangles. Counting triangles is appropriate for some applications [Prat-Pérez et al., 2014, Gao et al., 2022], but these structures can be averaged-out in embeddings. Complete U-statistics can be impossible to represent in low dimensions [Seshadhri et al., 2020], but fortunately they are not required for downstream tasks such as network visualisation [McInnes et al., 2018, Van der Maaten and Hinton, 2008], anomaly detection [Akoglu et al., 2015], and graph similarity comparisons [Koutra et al., 2013], which are the focus of this paper. For such tasks, even 'useful' bootstrap procedures are often dismissed, with no readily available replacements. We therefore introduce a suitable notion of validity for embeddings, which insists that the observed and bootstrapped network have statistically indistinguishable embedding distributions. This can be formalised into a test for whether network bootstraps are exchangeable with the observed network in a joint embedding. The joint embedding is chosen to have the *stability* property, i.e. nodes that have the same connectivity distribution but belong to different networks still receive the same embedding. Generating bootstraps that remain valid in an embedding space is crucial for many downstream tasks and is the focus of our contributions.

Figure 1 explains our contributions. (a) The Adjacency matrix $\boldsymbol{A}$ is modelled as $\boldsymbol{A}_{ij} \overset{\text{ind}}{\sim} \text{Bernoulli}\,(\boldsymbol{P}_{ij})$, for $i, j \in \{1, ..., n\}$. (b) We then compute an embedding $\hat{\boldsymbol{X}}$ of the network which (c) we utilise to form a broad range of estimators for $\boldsymbol{P}$; we add k-Nearest Neighbours on nonlinear embeddings to the options. (d) Unfolding the observed and bootstrapped adjacency matrices, $\boldsymbol{A}$ and $\tilde{\boldsymbol{A}}$, respectively, allows them to be (e) embedded into a shared space, from which downstream tasks of uncertainty quantification can be performed. These include our Bootstrap Exchangeability Test and (f) visualising uncertainty in t-SNE [Van der Maaten and Hinton, 2008] embeddings.

2 THEORETICAL BACKGROUND

Our proposed bootstrapping procedure requires a method for embedding a single network. For our proposed method to validate the bootstrap, we need a method for embedding multiple networks. We introduce both types of embedding in this section, as well as the concept of exchangeability, which are core concepts in our main contributions.

2.1 EMBEDDING SINGLE NETWORKS

An undirected n node network is represented using a binary and symmetric adjacency matrix $\boldsymbol{A} \in \{0,1\}^{n\times n}$, where $\boldsymbol{A}_{ij} = 1$ if an edge exists between nodes i and j and $\boldsymbol{A}_{ij} = 0$ otherwise for each $i, j \in \{1, \ldots, n\}$. This paper considers networks which can be modelled using a binary inhomogeneous random graph (BIRG) [Söderberg, 2002],

$$\boldsymbol{A}_{ij} \overset{\text{ind}}{\sim} \text{Bernoulli}(\boldsymbol{P}_{ij}),$$

where $\boldsymbol{P}_{ij}$ denotes the probability of an edge between nodes i and j. For the remainder of this paper, unless otherwise stated, we assume that each $\boldsymbol{A}$ is drawn from a BIRG.

It is often useful to consider the graph using a low-dimensional representation $\hat{\boldsymbol{X}} \in \mathbb{R}^{n\times d}$, referred to as embedding. The embedding dimension d is often chosen such that $d \ll n$. While our contributions are not limited to a single embedding method, we will focus on ASE due to its simplicity. For ASE, it is common practice to observe a scree plot of the singular values and find the 'elbow' to decide the value of d [Zhu and Ghodsi, 2006, Nguyen and Holmes, 2019].

Definition 1 (Adjacency spectral embedding) The d-dimensional *adjacency spectral embedding* of a network $\boldsymbol{A}$ is given by

$$\hat{\boldsymbol{X}} = \hat{\boldsymbol{U}}_{\boldsymbol{A}} |\hat{\boldsymbol{\Sigma}}_{\boldsymbol{A}}|^{1/2},$$

where $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{A}} \in \mathbb{R}^{d\times d}$ is a diagonal matrix containing the d largest eigenvalues of $\boldsymbol{A}$ arranged with decreasing magnitude, and $\hat{\boldsymbol{U}}_{\boldsymbol{A}} \in \mathbb{R}^{n\times d}$ is a matrix containing, as columns, the corresponding eigenvectors. Here, the i-th row of the embedding matrix $\hat{\boldsymbol{X}} \in \mathbb{R}^{n\times d}$ is the transposed d-dimensional vector representation of the i-th row of $\boldsymbol{A}$. For single-network embeddings, we use the notation $\hat{\boldsymbol{X}}$.

2.2 EMBEDDING MULTIPLE NETWORKS

To validate a proposed bootstrap, we embed both the observed network and its bootstrap into the same space. For this task, we require a multiple-network embedding. Specifically, we denote the multi-network embedding by $\hat{\boldsymbol{Y}} = (\hat{\boldsymbol{Y}}^{(1)}; \ldots; \hat{\boldsymbol{Y}}^{(M)}) \in \mathbb{R}^{Mn\times d}$, which is a row-wise concatenation of the embeddings of M networks. Here, each $\hat{\boldsymbol{Y}}^{(m)} \in \mathbb{R}^{n\times d}$ represents the embedding of the m-th network $\boldsymbol{A}^{(m)}$ obtained from the multi-network embedding.

One method for multi-network embedding is unfolded adjacency spectral embedding (UASE), which embeds a collection of networks into a single embedding space, allowing for comparisons to be made across the networks [Jones and Rubin-Delanchy, 2020].

Definition 2 (Unfolded adjacency spectral embedding) Let $\boldsymbol{\mathcal{A}} = \left(\boldsymbol{A}^{(1)}, \ldots, \boldsymbol{A}^{(M)}\right) \in \{0,1\}^{n\times Mn}$ (column concatenation) be the unfolding of a collection of M n-node networks. The d-dimensional *unfolded adjacency spectral embedding* of this collection is given by

$$\hat{\boldsymbol{Y}} = \hat{\boldsymbol{V}}_{\boldsymbol{\mathcal{A}}} \hat{\boldsymbol{\Sigma}}_{\boldsymbol{\mathcal{A}}}^{1/2},$$

where $\hat{\boldsymbol{U}}_{\boldsymbol{\mathcal{A}}} \hat{\boldsymbol{\Sigma}}_{\boldsymbol{\mathcal{A}}} \hat{\boldsymbol{V}}_{\boldsymbol{\mathcal{A}}}^T$ is the rank-d truncated singular value decomposition of $\boldsymbol{\mathcal{A}}$, that is, $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{A}} \in \mathbb{R}^{d\times d}$ is a diagonal

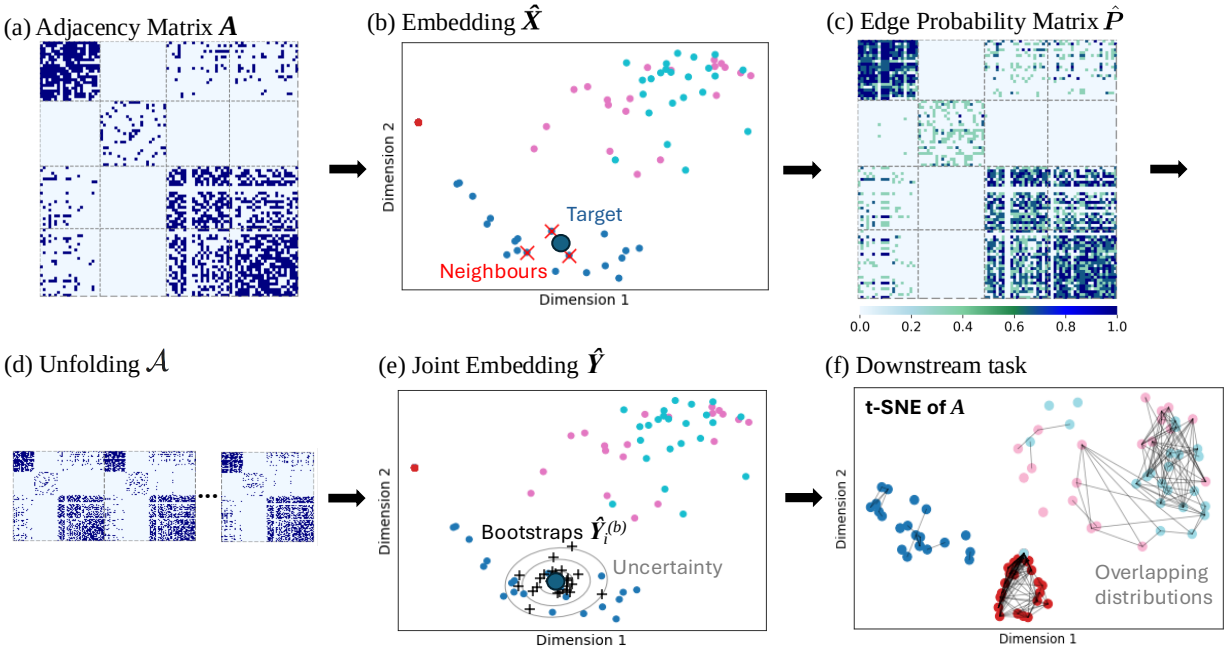

Figure 1: An overview of the proposed graph bootstrap framework. (a): Observe an n node network as an adjacency matrix $\boldsymbol{A} \in \{0,1\}^{n\times n}$, where it is assumed that $\boldsymbol{A}_{ij} \overset{\text{ind}}{\sim} \text{Bernoulli}(\boldsymbol{P}_{ij})$. (b): Embed the network into $d \ll n$ dimensions, e.g. via ASE, to obtain a low-dimension representation of the network $\hat{\boldsymbol{X}} \in \mathbb{R}^{n\times d}$. (c): Find the k-nearest neighbours for each node and use these as in Algorithm 2 to estimate the probability matrix as $\hat{\boldsymbol{P}} \in [0,1]^{n\times n}$. (d): Generate B bootstrap resamples of the observed network by sampling $\widetilde{\boldsymbol{A}}_{ij}^{(b)} \overset{\text{ind}}{\sim} \text{Bernoulli}(\hat{\boldsymbol{P}}_{ij})$ for $b = 1, ..., B$. (e): Use the bootstrap resamples for downstream tasks, such as estimating nodewise variance, as shown. (f): We can link nodes by whether their distributions overlap in an visualisation, here t-SNE.

matrix containing the d largest singular values of $\boldsymbol{\mathcal{A}}$ arranged in decreasing order, and $\hat{\boldsymbol{U}}_{\boldsymbol{A}}, \hat{\boldsymbol{V}}_{\boldsymbol{A}}$ contain the corresponding left and right singular vectors, respectively. Here, $\hat{\boldsymbol{Y}} = (\hat{\boldsymbol{Y}}^{(1)}; \ldots; \hat{\boldsymbol{Y}}^{(M)}) \in \mathbb{R}^{Mn\times d}$, where the i-th row of $\hat{\boldsymbol{Y}}^{(m)}$ contains the transposed d-dimensional vector representation of the i-th row of $\boldsymbol{A}^{(m)}$ for each $m \in \{1, \ldots, M\}$.

2.3 ACROSS-NETWORK EXCHANGEABILITY

UASE is far from the only method available for embedding multiple networks, e.g. Levin et al. [2017], Chen et al. [2020], Scheinerman and Tucker [2010], Lin et al. [2008]. However, we utilise UASE as unfolding allows for across-network exchangeability [Gallagher et al., 2021], which we leverage for our main contributions.

Definition 3 (Across-network exchangeability) Let $\boldsymbol{P}_i$ denote the i-th row of a probability matrix $\boldsymbol{P} \in [0,1]^{n\times n}$. Let $\hat{\boldsymbol{Y}}^{(1)}, \ldots, \hat{\boldsymbol{Y}}^{(T)} \in \mathbb{R}^{n\times d}$ be a d-dimensional multi-network embedding of a collection of BIRGs with probability matrices $\boldsymbol{P}^{(1)}, \ldots, \boldsymbol{P}^{(M)}$, respectively. This embedding has the property of across-network exchangeability if when $\boldsymbol{P}_i^{(m)} = \boldsymbol{P}_i^{(u)}$, we have that

$$\mathbb{P}\left(\hat{\boldsymbol{Y}}_i^{(m)} = v_1, \hat{\boldsymbol{Y}}_i^{(u)} = v_2\right) = \mathbb{P}\left(\hat{\boldsymbol{Y}}_i^{(m)} = v_2, \hat{\boldsymbol{Y}}_i^{(u)} = v_1\right),$$

for any $m, u \in \{1, \ldots, M\}$ and any $i \in \{1, \ldots, n\}$ [Davis et al., 2023]. In words, this condition states that embeddings $\hat{\boldsymbol{Y}}_i^{(m)}$ and $\hat{\boldsymbol{Y}}_i^{(u)}$ are exchangeable.

Alternative embedding methods can be used and have across-network exchangeability by utilising Dilated Unfolded Embedding [Davis et al., 2023]. For simplicity, we mainly focus on using UASE in this work. See Appendix A.1 for details.

3 CONTRIBUTIONS

Our main contributions are firstly, a novel method to validate bootstraps based on their exchangeability with the observed network, and secondly, two bootstrapping methods. One has theoretical guarantees in the asymptotic regime, and another improved empirical performance in the finite regime.

3.1 BOOTSTRAP VALIDATION PROCEDURE

In the unrealistic case where we know the true probability matrix $\boldsymbol{P}$ from which an observed network $\boldsymbol{A}$ was drawn,

then we could draw a true resample of that network as $\widetilde{\boldsymbol{A}}_{ij} \overset{\text{ind}}{\sim} \text{Bernoulli}(\boldsymbol{P}_{ij})$ for each $i, j \in \{1, \ldots, n\}$. This case represents the best possible bootstrap for our given model assumptions and the entries of $\boldsymbol{A}$ and $\widetilde{\boldsymbol{A}}$ will be exchangeable. Consequently, a good bootstrap must produce an $\widetilde{\boldsymbol{A}}$ with exchangeable entries to that of $\boldsymbol{A}$, based on the finite sample evidence. We therefore propose a method of testing for exchangeability between networks.

Directly comparing adjacency matrices entrywise as n grows is both computationally expensive and conceptually problematic without access to the true $\boldsymbol{P}$, as any errors in estimation can lead to bootstrap resamples which are not valid. To solve this problem, we opt to compare lower-dimensional embeddings of the networks, which can be estimated.

Our procedure for bootstrap validation considers an observed n node network $\boldsymbol{A}$ and one bootstrap $\widetilde{\boldsymbol{A}}$. We first compute a d-dimensional across-network exchangeable embedding $\hat{\boldsymbol{Y}} = (\hat{\boldsymbol{Y}}^{(\text{obs})}; \widetilde{\boldsymbol{Y}}) \in \mathbb{R}^{2n \times d}$, where $\hat{\boldsymbol{Y}}^{(\text{obs})}$ is the embedding of the observed network $\boldsymbol{A}$ and $\widetilde{\boldsymbol{Y}}$ is the embedding of its bootstrap $\widetilde{\boldsymbol{A}}$ in the shared space.

Due to across-network exchangeability, $\hat{\boldsymbol{Y}}^{(\text{obs})}$ and $\widetilde{\boldsymbol{Y}}$ will be exchangeable if $\boldsymbol{A}$ and $\widetilde{\boldsymbol{A}}$ are drawn from the same distribution. Next, we apply the paired displacement test from Davis et al. [2023] (a permutation test for a pair of networks), whose null hypothesis is that the two adjacency matrices are exchangeable, and therefore $\hat{\boldsymbol{Y}}_i^{(\text{obs})}$ and $\widetilde{\boldsymbol{Y}}_i$ have the same latent position, for all $i \in \{1, ..., n\}$. The hypotheses are thus

H_0 : $\hat{\boldsymbol{Y}}^{(\text{obs})}$ and $\widetilde{\boldsymbol{Y}}$ follow the same distribution,

H_1 : $\hat{\boldsymbol{Y}}^{(\text{obs})}$ and $\widetilde{\boldsymbol{Y}}$ do not follow the same distribution.

We exploit this property for the 'Bootstrap Exchangeability Test', which tests the validity of a *single bootstrap sample*. Let $\pi_1, ..., \pi_R$ be R permutations of $\hat{\boldsymbol{Y}}$, where each permutation swaps rows i and $i + n$ with probability $\frac{1}{2}$ for $i = 1, ..., n$. This gives R permuted versions of $\hat{\boldsymbol{Y}}$ denoted by $\pi_1(\hat{\boldsymbol{Y}}), ..., \pi_r(\hat{\boldsymbol{Y}})$. For each permutation, we calculate a test statistic $t(\pi_r(\hat{\boldsymbol{Y}}))$ capturing the displacement of the second network from the first, as:

$$T_{r+1} = t(\pi_r(\hat{\boldsymbol{Y}})) = \left|\left| \sum_{i=1}^{N} \pi_r(\hat{\boldsymbol{Y}})_i - \sum_{i=N+1}^{2N} \pi_r(\hat{\boldsymbol{Y}})_i \right|\right|_2. \tag{1}$$

From these R test statistics we calculate a p-value:

$$\hat{p} = \frac{1}{R+1} \sum_{r=1}^{R+1} \mathbb{1}\{T_r \geq t_{obs}\}. \tag{2}$$

If the p-value provides insufficient evidence for the alternate hypothesis, then the bootstrap is deemed exchangeable with the observed, based on the finite sample evidence, and hence the bootstrap is valid.

A summary of this procedure is displayed in Algorithm 1, which for real data scenarios is applied to an observed matrix with each of its B bootstrap replicates individually, giving B p-values.

In simulated data scenarios we know the true probability matrix $\boldsymbol{P}$, which allows a more robust verification of bootstrap validity by conditioning on $\boldsymbol{P}$ rather than $\boldsymbol{A}$. We therefore take M adjacency matrices draws $\boldsymbol{A}^{(1)}, ..., \boldsymbol{A}^{(M)}$ from the *known* probability matrix $\boldsymbol{P}$. For each matrix, we draw one bootstrap resample given by $\widetilde{\boldsymbol{A}}^{(1)}, ..., \widetilde{\boldsymbol{A}}^{(M)}$, respectively, and apply Algorithm 1 to each pair.

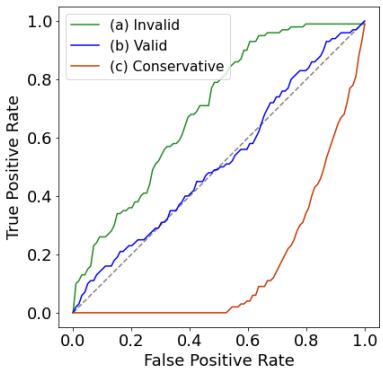

Figure 2: Example QQ-plots. (a): An **invalid resample** has too many small p-values and occurs if the resampled data are centred too far from the true latent position, when compared with the embedding of the observed data, which would lead to overestimation of the variance. (b): A **valid resample** has uniformly distributed p-values and the bootstrap resamples are exchangeable with the observed - we cannot tell which data come from which network. (c): A **conservative resample** has too many large p-values and occurs when the bootstrap resamples are too similar to the observed, which would lead to underestimation of the variance.

Now that we have introduced our bootstrap validation procedure, we state our main theorem.

Theorem 1. Let $\boldsymbol{A}^{(1)}, \ldots, \boldsymbol{A}^{(M)}$ be a collection of BIRGs with probability matrices $\boldsymbol{P}^{(1)}, \ldots, \boldsymbol{P}^{(M)}$, respectively. Let $\widetilde{\boldsymbol{A}}^{(1)}, \ldots, \widetilde{\boldsymbol{A}}^{(M)}$ be a collection of truly resampled networks, that is, $\widetilde{\boldsymbol{A}}_{ij}^{(m)} \overset{\text{ind}}{\sim} \text{Bernoulli}(\boldsymbol{P}_{ij}^{(m)})$ for each $i, j \in \{1, \ldots, n\}$ and $m \in \{1, \ldots, M\}$. Then, the distribution of the M p-values computed by the bootstrap exchangeability test (Algorithm 1) on each pair $(\boldsymbol{A}^{(m)}, \widetilde{\boldsymbol{A}}^{(m)})$ is uniform on $[0, 1]$.

For proof see Appendix A.2. In words, this theorem states that in the case where a bootstrap is truly exchangeable with the observed, our bootstrap validation procedure will provably deem the bootstrap valid. Theorem 1 provides a framework for verifying whether a proposed bootstrapping procedure generates exchangeable networks, offering a ro-

bust tool for validating the effectiveness of the bootstrapping procedure.

When applying Theorem 1, we obtain M p-values, call these $\hat{p}_1, ..., \hat{p}_M$. We can sort these p-values such that $\hat{p}_{1'} \leq ... \leq \hat{p}_{M'}$. Under the null hypothesis, the p-values follow a Uniform[0,1] distribution, thus we can associate the m-th sorted p-value $\hat{p}_{m'}$ with the m-th quantile given by $q_{m'} = \frac{m'}{M+1}$ for $m \in \{1, ..., M\}$. By plotting each sorted p-value with its associated quantile we build a quantile-quantile (QQ)-plot. We construct a 'Bootstrap Validity Score'

$$S = \frac{1}{M} \sum_{m'=1}^{M} |\hat{p}_{m'} - q_{m'}|, \tag{3}$$

where the closer the score is to 0, the closer to uniform the p-values are. A small Bootstrap Validity Score implies the bootstrap procedure used creates exchangeable bootstraps.

Figure 2 illustrates the possible outcomes of the testing procedure for the sorted p-values. A *valid* bootstrap procedure has a QQ-plot close to the diagonal and $S \approx 0$. If $\widetilde{\boldsymbol{A}}$ is far from $\boldsymbol{A}$, then this curve appears super-uniform; we deem such a bootstrap to be *invalid*. Invalid bootstraps may be overdispersed (i.e. the bootstrap embeddings of the node would imply a variance larger than the true variance) or mean-biased (the bootstrapped mean is not centred on the true latent position). On the other hand, if $\widetilde{\boldsymbol{A}}$ is too similar to $\boldsymbol{A}$, then the curve appears sub-uniform; in this case, the hypothesis testing literature would deem the test *conservative*. The embeddings being underdispersed can give a conservative test result. Both invalid and conservative bootstraps are distinguishable from the observed data and node variances would be misestimated. Therefore, we recommend that only valid bootstraps be trusted for such downstream applications.

3.2 NAÏVE BOOTSTRAP

Here, we consider a natural naïve bootstrap method. For the moment, let us consider a different network model, a Random Dot Product Graph (RDPG) [Young and Scheinerman, 2007, Nickel, 2006]. An RDPG is a latent position model, meaning it models each node as having a true position in $\mathbb{R}^d$ latent space, and provides a model-based rationale for spectral embedding. The RDPG assumes that $\boldsymbol{P} = \boldsymbol{X}\boldsymbol{X}^T$, where $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ contains, as rows, the d-dimensional latent position vectors for each node. It is therefore reasonable to consider the ASE estimator $\hat{\boldsymbol{P}} = \hat{\boldsymbol{X}}\hat{\boldsymbol{X}}^T$ as in Lemma 1.

Lemma 1. *Let $\hat{\boldsymbol{X}} \in \mathbb{R}^{n \times d}$ be an ASE of an observed adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ and let $\boldsymbol{P} \in [0,1]^{n \times n}$ be the corresponding rank-d probability matrix. Assuming that $O(\|\hat{\boldsymbol{X}}_i\|), O(\|\boldsymbol{X}_i\|) = O(\text{polylog}(n))$ for all i,*

$$\lim_{n \to \infty} ||\hat{\boldsymbol{X}}\hat{\boldsymbol{X}}^T - \boldsymbol{P}|| = 0.$$

Algorithm 1 Bootstrap Exchangeability Test

Input:
Observed network $\mathbf{A} \in \{0,1\}^{n \times n}$
Bootstrap network $\widetilde{\mathbf{A}} \in \{0,1\}^{n \times n}$
Embedding dimension $d \leq n$
Number of permutations R
Test statistic T_{r+1} from Eq. 1.
Compute:
Compute the d-dimensional UASE of $\boldsymbol{A}$ and $\widetilde{\boldsymbol{A}}$, $\hat{\boldsymbol{Y}} = (\hat{\boldsymbol{Y}}^{(\text{obs})}; \widetilde{\boldsymbol{Y}}) \in \mathbb{R}^{2n \times d}$
Compute observed test statistic $T_1 = t_{obs} = t(\hat{\boldsymbol{Y}}) = t(\hat{\boldsymbol{Y}}^{(obs)}, \widetilde{\boldsymbol{Y}})$
for $r = 1$ to R **do**
    for $i = 1$ to n **do**
        Swap rows i and $i + n$ of $\hat{\mathbf{Y}}$ with probability $\frac{1}{2}$ to obtain $\pi_r(\hat{\mathbf{Y}})$
    end for
    Compute permuted test statistic $T_{r+1} = t(\pi_r(\hat{\boldsymbol{Y}}))$
end for
Compute p-value using Eq. 2.
Output: $\hat{p}$

For proof see Appendix A.3. However, the estimator $\hat{\boldsymbol{P}} = \hat{\boldsymbol{X}}\hat{\boldsymbol{X}}^T$ is problematic. Firstly, in the finite-sample case, the entries of $\hat{\boldsymbol{P}} = \hat{\boldsymbol{X}}\hat{\boldsymbol{X}}^T$ can lie outside of the range $[0, 1]$. If this occurs, we set any values less than 0 to 0 and any values greater than 1 to 1. Secondly, the assumptions are strong; for example, if d needs to grow with n then $\|\hat{X}_i\|$ grows as well. In practice Levin and Levina [2019] show this estimator performs badly in U-statistic bootstrap tests, as it is not asymptotically consistent. We will use it as a baseline to show that our test is capable of identifying problematic bootstraps.

3.3 FINITE-SAMPLE KNN BOOTSTRAP

We now propose a new procedure for creating network bootstraps, by estimating the behaviour of each node based on its local k-nearest neighbours (kNN). Using ASE, high-numbers of dimensions can be needed to correctly estimate neighbourhoods, which can make estimation hard. Since kNN relies solely on a locally linear structure that preserves nearest-neighbourhood relationships, alternative embeddings that reduce dimensionality can also be applied.

Specifically, we apply kNN to a single network embedding $\hat{\boldsymbol{X}}$ to generate a k-length neighbourhood $\mathcal{N}_i \in \{1, \ldots, n\}^k$ for each $i \in \{1, \ldots, n\}$, where $i \in \mathcal{N}_i$ for all $i \in \{1, ..., n\}$. For the examples in this paper, Euclidean distance is used as the distance measure. The i-th row of the estimated probability matrix $\boldsymbol{P} \in \mathbb{R}^{n \times n}$ is given by the mean of the rows

of $\boldsymbol{A}$ indexed by the neighbours of node i,

$$\hat{\boldsymbol{P}}_i = \frac{1}{k} \sum_{j \in \mathcal{N}_i} \boldsymbol{A}_j$$

for each $i \in \{1, \ldots, n\}$. Networks are then sampled from a BIRG with a probability matrix $\hat{\boldsymbol{P}}$.

Algorithm 2 ASE-kNN

Input:
Observed network $\boldsymbol{A} \in \{0,1\}^{n \times n}$
Embedding dimension $d \leq n$
Number of nearest neighbours $k > 1$
Number of bootstrapped graphs B
Distance measure for the data and embedding method, e.g. Euclidean
Compute:
Compute the d-dimensional ASE $\hat{\boldsymbol{X}}$ of $\boldsymbol{A}$
for $i = 1, \ldots, n$ **do**
    Find the k nearest neighbours of node i in $\hat{\boldsymbol{X}}$, denoted by the set $\mathcal{N}_i \in \{1, \ldots, n\}^k$ (including node i), using specified distance measure
    Set $\hat{\boldsymbol{P}}_i = \frac{1}{k} \sum_{j \in \mathcal{N}_i} \boldsymbol{A}_j$
end for
for $b = 1, \ldots, B$ **do**
    Sample $\widetilde{\boldsymbol{A}}^{(b)}_{ij} \overset{\text{ind}}{\sim} \text{Bernoulli}(\hat{\boldsymbol{P}}_{ij})$
for each $i, j \in \{1, \ldots, n\}$
end for
Output: $\widetilde{\boldsymbol{A}}^{(1)}, \ldots, \widetilde{\boldsymbol{A}}^{(B)}$

For spectral embedding, an alternative form of $\hat{\boldsymbol{X}}$, given in Appendix A.4, can be used, which is less sensitive to the choice of d.

When choosing k there is a trade-off. If k is too small we may be ignoring neighbours with similar behaviour, resulting in a bootstrap that is too similar to the observation. However, if k is too large the model will use nodes embedded far away from the node of interest to estimate the latent position of the node of interest (see Figure 2). Appendix A.5.1 explores the sensitivity of this procedure to the choice of k. We check the appropriateness of the choice of k by considering the Bootstrap Validity Score S.

4 SYNTHETIC DATA EXAMPLE

In this section we empirically evaluate our proposed network procedures alongside other bootstrapping procedures using the Bootstrap Exchangeability Test given in Section 3.1. We use synthetic data generated from a Mixed Membership Stochastic Block Model (MMSBM) [Airoldi et al., 2008]. MMSBMs are an extension of the stochastic block model (SBM) [Holland et al., 1983] that allow nodes to have partial membership to multiple communities. In an MMSBM, each node is associated with a distribution over communities, rather than being assigned to a single community. A vector $\boldsymbol{\alpha} \in \mathbb{R}^C_+$ controls the mixed-membership allowed per node. For each node $i \in \{1, ..., n\}$ there is a C-dimensional mixed membership vector $\boldsymbol{\pi}_i \overset{\text{ind}}{\sim} \text{Dirichlet}(\boldsymbol{\alpha})$.

An indicator vector $\boldsymbol{z}_{i \to j}$ denotes the specific block membership of node i when it connects to node j and $\boldsymbol{z}_{i \to j}$ denotes the specific block membership of node j when it is connected from node i. For each pair of nodes (i, j), a membership indicator for the initiator, $\boldsymbol{z}_{i \to j} \overset{\text{ind}}{\sim} \text{Multinomial}(\boldsymbol{\pi}_i)$, is drawn, and a membership indicator for the receiver, $\boldsymbol{z}_{j \to i} \overset{\text{ind}}{\sim} \text{Multinomial}(\boldsymbol{\pi}_j)$, is drawn. A block probability matrix $\boldsymbol{B} \in [0,1]^{C \times C}$ defines the probability of interactions between the C communities, where $\boldsymbol{B}_{gh}$ is the probability of there being a connection from a node in community g to a node in community h, for $g, h \in \{1, ..., C\}$. By defining $\boldsymbol{B}$ to be symmetric, $\boldsymbol{B}_{gh} = \boldsymbol{B}_{hg}$ for all $g, h \in \{1, ..., C\}$, we have that $\mathbb{P}(\boldsymbol{A}_{ij}) = \mathbb{P}(\boldsymbol{A}_{ji})$ for all $i, j \in \{1, ..., n\}$, however this does not guarantee that $\boldsymbol{A}_{ij} = \boldsymbol{A}_{ji}$. To ensure sampled matrices are symmetric, whenever $i > j$, we set $\boldsymbol{A}_{ij} = \boldsymbol{A}_{ji}$. The value of the interaction from node i to node j is sampled as

$$\boldsymbol{A}_{ij} \overset{\text{ind}}{\sim} \text{Bernoulli}(\boldsymbol{z}_{i \to j} \boldsymbol{B} \boldsymbol{z}_{j \to i}), \tag{4}$$

for all $i, j \in \{1, ..., n\}$. In this example, there are $n = 300$ nodes and $C = 3$ communities. We define $\boldsymbol{\alpha} = \mathbf{1}_C$, that is, each node is equally likely to belong to each community. We define the block probability matrix $\boldsymbol{B}$ as:

$$\boldsymbol{B} = \begin{bmatrix} 0.3 & 0.2 & 0.2 \\ 0.2 & 0.6 & 0.2 \\ 0.2 & 0.2 & 0.9 \end{bmatrix}. \tag{5}$$

Here we demonstrate the method is applicable to non clustered data; see Appendix A.5 for a synthetic example on clustered data[1].

As introduced in Section 3.1, we can use Algorithm 1 $M \in \mathbb{N}$ times to verify if a procedure for generating bootstrapped networks creates exchangeable networks. For this example we take $M = 300$ random samples from the model specified by Equations 4 and 5 to create the adjacency matrices $\boldsymbol{A}^{(1)}, ..., \boldsymbol{A}^{(M)}$, and choose $d = 3$ for ASE since $\text{rank}(\boldsymbol{B}) = 3$. For each matrix one bootstrap resample is generated as in Algorithm 2 with $k = 5$, given by $\widetilde{\boldsymbol{A}}^{(1)}, ..., \widetilde{\boldsymbol{A}}^{(M)}$, respectively. To each pair $[\boldsymbol{A}^{(m)}, \widetilde{\boldsymbol{A}}^{(m)}]$, for $m = 1, ..., M$, we apply Algorithm 1, and use the M $\hat{p}$-values to give a QQ-plot and a Bootstrap Validity Score. We repeat the kNN-based bootstrap with different values of k to demonstrate the effect of k in the model. Other bootstrap

[1] We provide all data and implementation code for the synthetic and real-data experiments described in this paper at: `https://github.com/0emerald/ValidBootstrapsForNetworkEmbeddings`

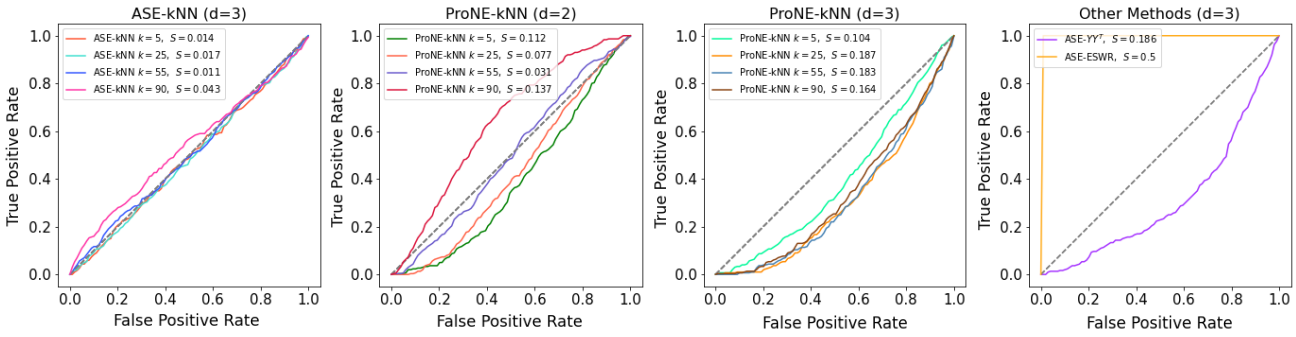

Figure 3: Simulated results for the 3 community MMSBM ($n = 300$) with different embedding and bootstrap resampling methods applied. Each curve uses $M = 300$ draws from the known model, each paired with a single bootstrap from the model. The area between the curve and the $x = y$ line is the Bootstrap Validity Score S.

methods are also evaluated on this data (see Appendix A.6), in addition to using ProNE [Zhang et al., 2019] to create non-linear low-dimensional embeddings to apply bootstrap methods to.

In this example, we use a variety of methods to estimate $\hat{\boldsymbol{P}}$ from one network observation and draw bootstraps, which we then evaluate with the validity test (observed visually with the QQ-plots in Figure 3) to decide which method for estimating $\hat{\boldsymbol{P}}$ gives exchangeable embeddings with the one observation. We are then free to use any embedding bootstraps that do pass the test for any downstream tasks.

Figure 3 shows that only our kNN methods achieve validity, though non-linear embeddings require more tuning as they compress information into fewer dimensions. Appendix A.7 considers the effects of changing n in this model.

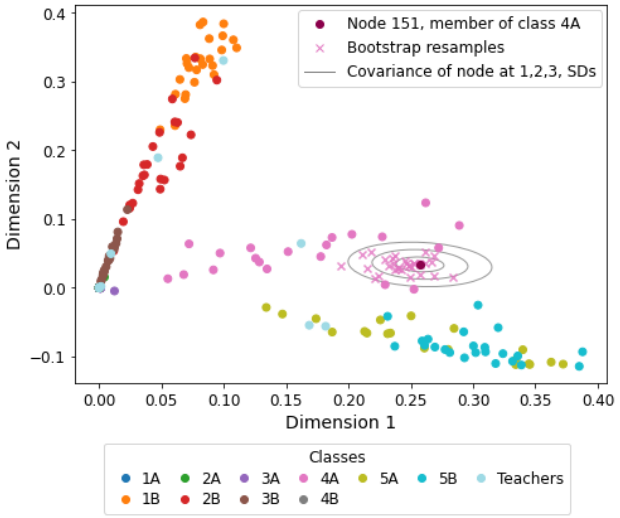

Figure 4: $\hat{\boldsymbol{Y}}^{(\text{obs})}$ the first 2 dimensions of the School data for all nodes. For a student node in class 4A, we visualise $B = 20$ bootstrap embeddings, with the covariance ellipse at 1, 2, and 3 standard deviations.

5 REAL-WORLD DATA

5.1 SCHOOL SOCIAL INTERACTION NETWORK

For a concrete analysis on real data we consider a social interaction network. The Lyon school dataset captures face-to-face interactions between members of a French primary school over two days in October 2009, from 08:00 to 18:00 each day [Stehlé et al., 2011, Rossi and Ahmed, 2015]. The network tracks the interactions of 242 participants (10 teachers and 232 students) across 5 school years, each year group having two classes. Each person wore a Radio Frequency Identification badge which recorded an interaction between two people if they were in close proximity for 20 seconds.

The data is binned into hour-long time windows over the two days to give a time series of networks. When two participants had one or more interactions in the hour-long window, an edge is present between them in the corresponding network. This gives a series of graphs $\boldsymbol{A}^{(1)}, ..., \boldsymbol{A}^{(20)} \in \{0,1\}^{n\times n}$, where $n = 242$.

To highlight the performance and a useful application of the methodology in this paper, we consider a single from the series, 9:00-10:00 of Day 1. This is $\boldsymbol{A}^{(2)}$, which we shall henceforward denote as $\boldsymbol{A}^{(obs)}$. We apply our bootstrap method to this data as in Algorithm 2 to obtain $B = 500$ bootstrap replicates of $\boldsymbol{A}^{(obs)}$. We embed the observed matrix along with the $B = 500$ bootstraps into a common $d = 12$ dimensional space via UASE of $\boldsymbol{\mathcal{A}} = \left(\boldsymbol{A}^{(obs)}, \widetilde{\boldsymbol{A}}^{(1)}, ..., \widetilde{\boldsymbol{A}}^{(B)}\right)$, to get an embedding for the observed matrix and the B bootstrap matrices, denoted by $\hat{\boldsymbol{Y}}^{(\text{obs})}, \widetilde{\boldsymbol{Y}}^{(1)}, ..., \widetilde{\boldsymbol{Y}}^{(B)} \in \mathbb{R}^{n\times d}$, respectively. From these, we are able to estimate the covariance of each node in the embedding space. Figure 4 shows the first 2 dimensions of $\hat{\boldsymbol{Y}}^{(\text{obs})}$ with bootstrap embeddings and Normal-approximation uncertainty for one node.

5.2 APPLYING DIFFERENT EMBEDDING METHODS TO THE SCHOOL DATA

Figure 5a shows that the empirical performance of different estimators of $\hat{\boldsymbol{P}}$ varies considerably. Despite asymptotic guarantees, ASE-based $\boldsymbol{X}\boldsymbol{X}^T$ approaches create bootstraps which the Bootstrap Exchangeability Test do not deem exchangeable. Bootstraps from nonlinear methods perform better; both kNN applied to the ASE embedding (ASE-kNN) and kNN applied to the ProNE embedding (ProNE-kNN).

Our bootstrap procedure generates uncertainty for each point in the embedding, which we have providing a formal test to ensure validity. One important application is to score common network embeddings in terms of whether they represent this uncertainty. Dimension reduction techniques that preserve global structure in 2-3D such as t-SNE [Van der Maaten and Hinton, 2008] and UMAP [McInnes et al., 2018] are often applied. See Appendix A.8 for a brief overview of t-SNE. We apply t-SNE to $\boldsymbol{A}^{(obs)}$ to visualise the adjacency matrix in 2-dimensions. Bootstrapping $\boldsymbol{A}^{(obs)}$ allows us to estimate the covariance of each node in the $d = 10$ dimensional embedding $\hat{\boldsymbol{Y}}^{(obs)}$ which shows whether dimension reduction method such as t-SNE is appropriate. To do this we construct a symmetric "fuzziness" matrix $\boldsymbol{F} \in \{0,1\}^{n\times n}$, where

$$\boldsymbol{F}_{ij} = \begin{cases} 1 & \text{if node } i \text{ is within 3 standard deviations} \\ & \text{of node } j \text{ and node } j \text{ is within 3 standard} \\ & \text{deviations of node } i, \\ 0 & \text{otherwise.} \end{cases}$$

An edge is drawn between nodes i and j in the t-SNE plot if $\boldsymbol{F}_{ij} = 1$, with the interpretation that these nodes have overlapping probability distributions. Appendix A.9 conducts sensitivity analysis to ensure robustness to t-SNE hyperparameters and the number of bootstraps B.

To use network uncertainty to quantify the performance of some (externally chosen) embedding, we can then provide a score for a particular (standardised) node visualisation layout $\boldsymbol{V}$, which rewards placing nodes with overlapping distributions close to one another, whilst also retaining structure within uncertain nodes by penalising nodes that are placed too close. For this we construct the 'Fuzziness Score' as the standard deviation of the edge length

$$F(\boldsymbol{V}) = \mathrm{Var}(|\boldsymbol{V}_i - \boldsymbol{V}_j|_2; \boldsymbol{F}_{ij} = 1). \tag{6}$$

This penalises both misplacing nodes in the wrong cluster, and giving uncertain nodes 'structure'. To select a value for the perplexity hyperparameter we minimise the 'Fuzziness score'. This implies an optimal perplexity hyperparameter for t-SNE on the School data (of 125±75; Figure 12).

Figure 5b-c shows $\boldsymbol{F}$ as edges highlighting which nodes probabilistically overlap. Whilst the latent positions $\boldsymbol{Y}_i$ and $\boldsymbol{Y}_j$ may not be close, if $\boldsymbol{F}_{ij} = 1$ then it is possible that $\hat{\boldsymbol{Y}}_i$ and $\hat{\boldsymbol{Y}}_j$ could be the same.

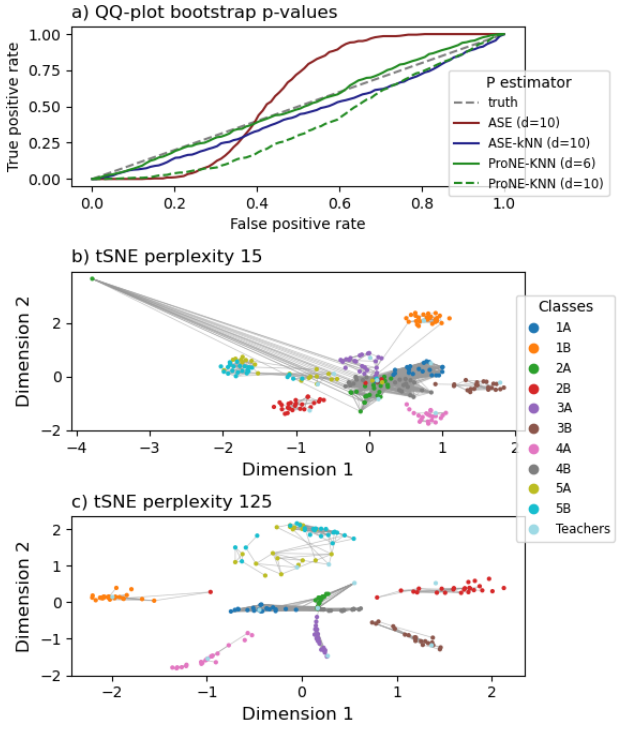

Figure 5: Insights from uncertainty in the School data. (a): QQ-plots from applying the Bootstrap Exchangeability Test to different bootstrap methods which use different estimators of $\hat{\boldsymbol{P}}$. (b): t-SNE of $\boldsymbol{A}^{(obs)}$ using too-low perplexity results in distant points having overlapping uncertainty, shown as lines connecting nodes within 3 standard deviations of one another ($\hat{\boldsymbol{P}}$ from $d = 6$ ProNE embedding). (c): t-SNE of $\boldsymbol{A}^{(obs)}$ using higher perplexity results in fewer poorly placed nodes.

Using kNN bootstraps, we quantify the performance of t-SNE embeddings (Figure 5b-c) using node-wise variance. Annotating the fuzziness highlights across-cluster interactions that t-SNE alone does not show. The low perplexity visualisation is misleading - e.g. it places a class 2A student (green) away from students with overlapping distributions under bootstrap, and similarly a class 2B student (red) is placed centrally. Conversely, the high perplexity solution avoids this problem, whilst uncertainty highlights that the red 2B student's distribution overlaps with 1B.

6 DISCUSSION

Obtaining valid bootstrap resamples from a network embedding requires only a single substantial assumption, that nodes are exchangeable. This permits versatile resampling from a single network observation, validated by a reliable Bootstrap Exchangeability Test to confirm resample exchangeability with the observed network. Our approach relies on $\boldsymbol{P}$ being a sufficient statistic for $\boldsymbol{A}^{(obs)}$, so extend-

ing the methodology to work for weighted networks would require additional distributional assumptions. Experimental evaluation demonstrates the method is able to resample networks of various structures. The bootstrap validity test supports the flexibility to incorporate various embedding methods, including those optimised for scalability with large datasets, ensuring applicability to networks with increasing complexity and size.

In the common case where only one instance of a network is available, care must be taken not to overfit. The fuzziness score and other metrics should be viewed as a sensitivity analysis, and not subject to premature optimisation. The number of neighbours k should be chosen by a prior based on community sizes, and run with other k used to ensure a lack of sensitivity to this parameter.

Previous studies found that using the estimate $\hat{\boldsymbol{P}} = \hat{\boldsymbol{X}}\hat{\boldsymbol{X}}^T$ introduces bias in count statistics. Our findings extend this by showing it does not provide a valid bootstrap under the exchangeable bootstrap framework proposed in this paper. In contrast, the ASE-kNN bootstrap we introduce generates exchangeable embeddings, but it underestimates the variability in metrics such as average node degree and triangle density in an SBM example.

Future work aimed at developing network bootstrapping methods that satisfy all validity criteria could enable broader applications in downstream tasks. By offering a novel perspective on evaluating bootstrap validity, we address a range of network analysis applications that rely on low-dimensional representations rather than entire networks.

References

Edo M Airoldi, David Blei, Stephen Fienberg, and Eric Xing. Mixed membership stochastic blockmodels. *Advances in neural information processing systems*, 21, 2008.

Leman Akoglu, Hanghang Tong, and Danai Koutra. Graph based anomaly detection and description: a survey. *Data mining and knowledge discovery*, 29:626–688, 2015. doi: 10.1007/s10618-014-0365-y.

Sharmodeep Bhattacharyya and Peter J. Bickel. Subsampling bootstrap of count features of networks. *The Annals of Statistics*, 43(6):2384 – 2411, 2015. doi: 10.1214/15-AOS1338.

Tristan Bilot, Nour El Madhoun, Khaldoun Al Agha, and Anis Zouaoui. Graph neural networks for intrusion detection: A survey. *IEEE Access*, 2023. doi: 10.1109/ACCESS.2023.3275789.

Christian Borgs, Jennifer T Chayes, László Lovász, Vera T Sós, and Katalin Vesztergombi. Convergent sequences of dense graphs i: Subgraph frequencies, metric properties and testing. *Advances in Mathematics*, 219(6): 1801–1851, 2008. doi: 10.1016/j.aim.2008.07.008.

Benjamin Bowman and H Howie Huang. Towards next-generation cybersecurity with graph ai. *ACM SIGOPS Operating Systems Review*, 55(1):61–67, 2021. doi: 10.1145/3469379.3469386.

Joshua Cape, Minh Tang, and Carey E. Priebe. The two-to-infinity norm and singular subspace geometry with applications to high-dimensional statistics. *The Annals of Statistics*, 47(5):2405–2439, 2019. ISSN 00905364, 21688966. doi: 10.48550/arXiv.1705.10735.

Guodong Chen, Jesús Arroyo, Avanti Athreya, Joshua Cape, Joshua T Vogelstein, Youngser Park, Chris White, Jonathan Larson, Weiwei Yang, and Carey E Priebe. Multiple network embedding for anomaly detection in time series of graphs. *arXiv preprint arXiv:2008.10055*, 2020. doi: 10.48550/arXiv.2008.10055.

Ed Davis, Ian Gallagher, Daniel John Lawson, and Patrick Rubin-Delanchy. A simple and powerful framework for stable dynamic network embedding. *arXiv preprint arXiv:2311.09251*, 2023. doi: 10.48550/arXiv.2311.09251.

Ed Davis, Ian Gallagher, Daniel John Lawson, and Patrick Rubin-Delanchy. Valid conformal prediction for dynamic gnns. *arXiv preprint arXiv:2405.19230*, 2024. doi: 10.48550/arXiv.2405.19230.

Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000. doi: 10.1137/S0895479896305696.

B. Efron. Bootstrap Methods: Another Look at the Jackknife. *The Annals of Statistics*, 7(1):1 – 26, 1979. doi: 10.1214/aos/1176344552.

Bradley Efron and Robert J Tibshirani. *An introduction to the bootstrap*. Chapman and Hall/CRC, 1994. doi: 10.1201/9780429246593.

Sacha Epskamp, Denny Borsboom, and Eiko I Fried. Estimating psychological networks and their accuracy: A tutorial paper. *Behavior research methods*, 50:195–212, 2018. doi: 10.3758/s13428-017-0862-1.

Ian Gallagher, Andrew Jones, and Patrick Rubin-Delanchy. Spectral embedding for dynamic networks with stability guarantees. *Advances in Neural Information Processing Systems*, 34:10158–10170, 2021. doi: 10.48550/arXiv.2106.01282.

Yang Gao, Xiangzhan Yu, and Hongli Zhang. Graph clustering using triangle-aware measures in large networks. *Information Sciences*, 584:618–632, 2022. ISSN 0020-0255. doi: 10.1016/j.ins.2021.11.008.

Alden Green and Cosma Rohilla Shalizi. Bootstrapping exchangeable random graphs. *Electronic Journal of Statistics*, 16(1):1058–1095, 2022. doi: 10.1214/21-EJS1896.

Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016. doi: 10.48550/arXiv.1607.00653.

Haoyu He, Yuede Ji, and H Howie Huang. Illuminati: Towards explaining graph neural networks for cybersecurity analysis. In *2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P)*, pages 74–89. IEEE, 2022. doi: 10.1109/EuroSP53844.2022.00013.

Peter D Hoff, Adrian E Raftery, and Mark S Handcock. Latent space approaches to social network analysis. *Journal of the american Statistical association*, 97(460):1090–1098, 2002. doi: 10.1198/016214502388618906.

Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983. doi: 10.1016/0378-8733(83)90021-7.

Andrew Jones and Patrick Rubin-Delanchy. The multilayer random dot product graph. *arXiv*, page arXiv:2007.10455, 2020. doi: 10.48550/ARXIV.2007.10455.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021. doi: 10.1038/s41586-021-03819-2.

Danai Koutra, Joshua T Vogelstein, and Christos Faloutsos. Deltacon: A principled massive-graph similarity function. In *Proceedings of the 2013 SIAM international conference on data mining*, pages 162–170. SIAM, 2013. doi: 10.1137/1.9781611972832.18.

Jens-Peter Kreiss and Efstathios Paparoditis. Bootstrap methods for dependent data: A review. *Journal of the Korean Statistical Society*, 40(4):357–378, 2011. doi: 10.1016/j.jkss.2011.08.009.

Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22 (1):79–86, 1951.

Keith Levin and Elizaveta Levina. Bootstrapping networks with latent space structure. *arXiv preprint arXiv:1907.10821*, 2019. doi: 10.48550/arXiv.1907.10821.

Keith Levin, Avanti Athreya, Minh Tang, Vince Lyzinski, and Carey E Priebe. A central limit theorem for an omnibus embedding of multiple random dot product graphs. In *2017 IEEE international conference on data mining workshops (ICDMW)*, pages 964–967. IEEE, 2017. doi: 10.48550/arXiv.1705.09355.

Heng Lian. Convergence of functional k-nearest neighbor regression estimate with functional responses. *Electronic Journal of Statistics*, 5:31–40, 2011. ISSN 1935-7524, 1935-7524. doi: 10.1214/11-EJS595.

Yu-Ru Lin, Yun Chi, Shenghuo Zhu, Hari Sundaram, and Belle L Tseng. Facetnet: a framework for analyzing communities and their evolutions in dynamic networks. In *Proceedings of the 17th international conference on World Wide Web*, pages 685–694, 2008. doi: 10.1145/1367497.1367590.

László Lovász and Balázs Szegedy. Limits of dense graph sequences. *Journal of Combinatorial Theory, Series B*, 96(6):933–957, 2006. doi: 10.1016/j.jctb.2006.05.002.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018. doi: 10.48550/arXiv.1802.03426.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013. doi: 10.48550/arXiv.1301.3781.

Lan Nguyen and Susan Holmes. Ten quick tips for effective dimensionality reduction. *PLOS Computational Biology*, 15:e1006907, 06 2019. doi: 10.1371/journal.pcbi.1006907.

Christine Leigh Myers Nickel. *Random dot product graphs a model for social networks*. PhD thesis, Johns Hopkins University, 2006.

Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2 (11):559–572, 1901. doi: 10.1080/14786440109462720.

Arnau Prat-Pérez, David Dominguez-Sal, and Josep-Lluis Larriba-Pey. High quality, scalable and parallel community detection for large real graphs. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14, page 225–236, New York, NY, USA, 2014. Association for Computing Machinery. ISBN 9781450327442. doi: 10.1145/2566486.2568010.

Tai Qin and Karl Rohe. Regularized spectral clustering under the degree-corrected stochastic blockmodel. *Advances in neural information processing systems*, 26, 2013. doi: 10.48550/arXiv.1309.4111.

Gideon Rosenthal, František Váša, Alessandra Griffa, Patric Hagmann, Enrico Amico, Joaquín Goñi, Galia Avidan,

and Olaf Sporns. Mapping higher-order relations between brain structure and function with embedded vector representations of connectomes. *Nature communications*, 9(1): 2178, 2018. doi: 10.1038/s41467-018-04614-w.

Ryan Rossi and Nesreen Ahmed. The network data repository with interactive graph analytics and visualization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29, 2015. doi: 10.1609/aaai.v29i1.9277.

Patrick Rubin-Delanchy, Joshua Cape, Minh Tang, and Carey E. Priebe. A Statistical Interpretation of Spectral Embedding: The Generalised Random Dot Product Graph. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 84(4):1446–1473, 06 2022. ISSN 1369-7412. doi: 10.1111/rssb.12509.

Edward R Scheinerman and Kimberly Tucker. Modeling graphs using dot product representations. *Computational statistics*, 25:1–16, 2010. doi: 10.1007/s00180-009-0158-8.

Comandur Seshadhri, Aneesh Sharma, Andrew Stolman, and Ashish Goel. The impossibility of low-rank representations for triangle-rich complex networks. *Proceedings of the National Academy of Sciences*, 117(11):5631–5637, 2020.

Bo Söderberg. General formalism for inhomogeneous random graphs. *Physical review E*, 66(6):066121, 2002. doi: 10.1103/PhysRevE.66.066121.

Juliette Stehlé, Nicolas Voirin, Alain Barrat, Ciro Cattuto, Lorenzo Isella, Jean-François Pinton, Marco Quaggiotto, Wouter Van den Broeck, Corinne Régis, Bruno Lina, et al. High-resolution measurements of face-to-face contact patterns in a primary school. *PloS one*, 6(8):e23176, 2011. doi: 10.1371/journal.pone.0023176.

Charles J. Stone. Consistent Nonparametric Regression. *The Annals of Statistics*, 5(4):595–620, 1977. ISSN 0090-5364, 2168-8966. doi: 10.1214/aos/1176343886.

Daniel L. Sussman, Minh Tang, Donniell E. Fishkind, and Carey E. Priebe. A consistent adjacency spectral embedding for stochastic blockmodel graphs. *Journal of the American Statistical Association*, 107(499):1119–1128, 2012. doi: 10.1080/01621459.2012.699795.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of machine learning research*, 9(11), 2008.

Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. Graph neural networks in recommender systems: a survey. *ACM Computing Surveys*, 55(5):1–37, 2022. doi: 10.1145/3535101.

Stephen J Young and Edward R Scheinerman. Random dot product graph models for social networks. In *International Workshop on Algorithms and Models for the Web-Graph*, pages 138–149. Springer, 2007. doi: 10.1007/978-3-540-77004-6_11.

Jie Zhang, Yuxiao Dong, Yan Wang, Jie Tang, and Ming Ding. Prone: Fast and scalable network representation learning. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 4278–4284. International Joint Conferences on Artificial Intelligence Organization, 7 2019. doi: 10.24963/ijcai.2019/594.

Mu Zhu and Ali Ghodsi. Automatic dimensionality selection from the scree plot via the use of profile likelihood. *Computational Statistics & Data Analysis*, 51(2):918–930, 2006. ISSN 0167-9473. doi: 10.1016/j.csda.2005.09.010.

Tianhai Zu and Yichen Qin. Local bootstrap for network data. *Biometrika*, 112(1):asae046, 09 2024. ISSN 1464-3510. doi: 10.1093/biomet/asae046.

Valid Bootstraps for Network Embeddings with Applications to Network Visualisation

Emerald Dilworth[1] **Ed Davis**[1] **Daniel J. Lawson**[1]

[1]School of Mathematics, University of Bristol, Bristol, UK

A APPENDIX

A.1 OTHER ACROSS-NETWORK EXCHANGEABLE EMBEDDINGS

In Section 2.2 we introduced UASE, which is a multi-network embedding that has the property of across-network exchangeability. This property is utilised in our bootstrap exchangeability test (stated in Algorithm 1), in order to prove Theorem 1. UASE is just one of many possible *unfolded* embeddings [Davis et al., 2023, 2024], which have across-network exchangeability.

The standard unfolding, used by UASE, is defined as a column concatenation of the collection of networks $\boldsymbol{\mathcal{A}} = (\boldsymbol{A}^{(1)}, ..., \boldsymbol{A}^{(M)}) \in \mathbb{R}^{Mn \times n}$. However, it is possible to define a more general *dilated unfolding*,

$$\mathscr{D}(\boldsymbol{\mathcal{A}}) = \begin{bmatrix} \mathbf{0} & \boldsymbol{\mathcal{A}} \\ \boldsymbol{\mathcal{A}}^T & \mathbf{0} \end{bmatrix} \in \{0,1\}^{n(M+1) \times n(M+1)}. \tag{7}$$

Let $\mathcal{G} : \{\boldsymbol{A} \in \{0,1\}^{n \times n} : \boldsymbol{A} = \boldsymbol{A}^T\} \times \Omega \to \mathbb{R}^{n \times d}$ be a general function for single-network embedding of an adjacency matrix $\boldsymbol{A}$ with a seed $\omega \in \Omega$. We require this seed due to the random nature of most network embedding methods; for example, the specific choice of eigenvalues and eigenvectors in ASE is random. The inclusion of some random seed serves the purpose of eliminating this randomness, allowing for comparisons to be made across multiple runs of the embedding function. This is required for the following definition. $\mathcal{G}$ is a label-invariant embedding in the case where

$$\mathbb{P}(\mathcal{G}(\mathbf{a}, \omega) = \mathbf{v}) = \mathbb{P}\left(\mathcal{G}(\mathbf{\Pi a \Pi}^\top, \omega) = \mathbf{\Pi v}\right),$$

for any permutation matrix $\Pi \in \{0,1\}^{n \times n}$, $\mathbf{a} \in \mathbb{R}^{n \times n}$ and $\mathbf{v} \in \mathbb{R}^{n \times d}$. Davis et al. [2023] proves that any label-invariant $\mathcal{G}$, applied to $\mathscr{D}(\boldsymbol{\mathcal{A}})$, returns

$$\begin{bmatrix} \hat{\boldsymbol{X}} \\ \hat{\boldsymbol{Y}} \end{bmatrix} = \mathcal{G}\left(\mathscr{D}(\boldsymbol{\mathcal{A}}), \omega\right),$$

where $\hat{\boldsymbol{X}} \in \mathbb{R}^{n \times d}$ is a single embedding that summarises the collection of networks, and $\hat{\boldsymbol{Y}} \in \mathbb{R}^{Mn \times d}$ is an across-network exchangeable multi-network embedding. The authors show that when $\mathcal{G}$ is simply ASE, then the returned $\hat{\boldsymbol{Y}}$ is equivalent to UASE.

Using this more general definition of an unfolded embedding, we now have access to many other across-network exchangeable embeddings through the choice of $\mathcal{G}$. For example, we define unfolded ProNE, by setting $\mathcal{G}$ to be the ProNE embedding [Zhang et al., 2019].

A.2 PROOF OF THEOREM 1

Proof. UASE is a multi-network embedding that satisfies across-network exchangeability (Definition 3) regardless of embedding dimension (Gallagher et al. [2021], Davis et al. [2023] Theorem 5). Let $(\hat{\boldsymbol{Y}}^{(m)}; \hat{\boldsymbol{Y}}^{(m)}) \in \mathbb{R}^{2n \times d}$ denote a

d-dimensional UASE of $\boldsymbol{\mathcal{A}} = (\boldsymbol{A}^{(m)}, \widetilde{\boldsymbol{A}}^{(m)}) \in \mathbb{R}^{2n \times n}$ for all $m \in \{1, ..., M\}$. Both $\boldsymbol{A}^{(m)}$ and $\widetilde{\boldsymbol{A}}^{(m)}$ are BIRGs drawn the same probability matrix, $\boldsymbol{P}^{(m)}$. Since UASE is across-network exchangeable, it is true that

$$\mathbb{P}\left(\hat{\boldsymbol{Y}}_i^{(m)} = v_1, \widetilde{\boldsymbol{Y}}_i^{(m)} = v_2\right) = \mathbb{P}\left(\hat{\boldsymbol{Y}}_i^{(m)} = v_2, \widetilde{\boldsymbol{Y}}_i^{(m)} = v_1\right).$$

for all $m \in \{1, ..., M\}$. In words, the embedding of the observed network $\boldsymbol{A}^{(m)}$ is exchangeable with the embedding of the truly resampled network $\widetilde{\boldsymbol{A}}^{(m)}$ for each $m \in \{1, \ldots, M\}$.

Due to the exchangeability of each $\hat{\boldsymbol{Y}}^{(m)}, \widetilde{\boldsymbol{Y}}^{(m)}$, the permutations applied at each step $r \in \{1, ..., R\}$ does not alter the distribution of $t(\hat{\boldsymbol{Y}}^{(m)}, \widetilde{\boldsymbol{Y}}^{(m)})$. Therefore, the sequence $T_1, T_2 \ldots, T_{R+1}$ is exchangeable, that is

$$\mathbb{P}(T_1 \leq v_1, \ldots, T_{R+1} \leq v_{R+1}) = \mathbb{P}(T_1 \leq v_{\sigma(1)}, \ldots, T_{R+1} \leq v_{\sigma(R+1)})$$

for any $v_k \in \mathbb{R}$ and any permutation σ on $\{1, \ldots, R+1\}$.

As the sequence of test statistics is exchangeable, it follows that the p-value,

$$\hat{p} = \frac{1}{R+1} \sum_{r=1}^{R+1} \mathbb{1}\{T_r \geq t_{obs}\},$$

will be uniformly distributed on $[0, 1]$. □

A.3 PROOF OF LEMMA 1

Proof. From Gallagher et al. [2021] (Proposition 2) we have that 'there exists a sequence of orthogonal matrices $\boldsymbol{Q} \in O(d)$ such that

$$\max_{i=1,\ldots,n} ||\hat{\boldsymbol{X}}_i \boldsymbol{Q} - \boldsymbol{X}_i|| = O\left(\frac{\sqrt{\log(n)}}{\sqrt{\rho_n n}}\right) \tag{8}$$

with high probability, where it is assumed that the sparsity factor ρ_n satisfies $\rho_n = \omega\left(\frac{1}{n}\log^c(n)\right)$ for some universal constant $c > 1$'. This states that after applying an orthogonal transformation to an embedding $\hat{\boldsymbol{X}}$, which leaves the structure of the embedding unchanged, the embedding will converge in the Euclidean norm to the noise-free embedding $\boldsymbol{X}$ as the number of nodes increases. For a dense graph $\rho_n = 1$. To prove the lemma, it is enough to assume that $O(\|\hat{\boldsymbol{X}}_i\|), O(\|\boldsymbol{X}_i\|) = O(\text{polylog}(n))$ for all i. Under this assumption and using Equation 8 we obtain:

$$\begin{aligned}
\max_{i=1,\ldots,n} \|\hat{\boldsymbol{X}}_i \hat{\boldsymbol{X}}_i^T - \boldsymbol{X}_i \boldsymbol{X}_i^T\| &\leq \max_{i=1,\ldots,n} \left(\|\hat{\boldsymbol{X}}_i - \boldsymbol{X}_i\| \cdot \|\hat{\boldsymbol{X}}_i^T\| + \|\boldsymbol{X}_i\| \cdot \|\hat{\boldsymbol{X}}_i - \boldsymbol{X}_i\|\right) \\
&= O\left(\frac{\sqrt{\log(n)}}{\sqrt{\rho_n n}}\right) \cdot \max_{i=1,\ldots,n} \left(\|\hat{\boldsymbol{X}}_i\| + \|\boldsymbol{X}_i\|\right) \\
&= O\left(\frac{\sqrt{\log(n)} \cdot \text{polylog}(n)}{\sqrt{\rho_n n}}\right)
\end{aligned} \tag{9}$$

The limit $\lim_{n\to\infty} \frac{\sqrt{\log(n)} \cdot \text{polylog}(n)}{\sqrt{\rho_n n}} = 0$, from which we get $\lim_{n\to\infty} \max_{i=1,\ldots,n} ||\hat{\boldsymbol{X}}_i \hat{\boldsymbol{X}}_i^T - \boldsymbol{X}_i \boldsymbol{X}_i^T|| = 0$ which implies that $\lim_{n\to\infty} ||\hat{\boldsymbol{X}} \hat{\boldsymbol{X}}^T - \boldsymbol{X} \boldsymbol{X}^T|| = 0$. □

In general, the assumption $O(\|\hat{\boldsymbol{X}}_i\|), O(\|\boldsymbol{X}_i\|) = O(\text{polylog}(n))$ only holds under certain conditions.

A.4 ALTERNATIVE EMBEDDING DEFINITION WITH ASE

For an adjacency matrix $\boldsymbol{A} \in \{0,1\}^{n\times n}$ which represents a binary n node network, typically when this is embedded spectrally via a truncated SVD, it is as:

$$\boldsymbol{A} \approx \boldsymbol{U_A}\boldsymbol{\Sigma_A}\boldsymbol{V_A}^T, \tag{10}$$

where the largest $d \ll n$ singular values are kept. Here, $\boldsymbol{\Sigma_{\mathcal{A}}} \in \mathbb{R}^{d\times d}$ is a diagonal matrix of the d largest singular values of $\boldsymbol{\mathcal{A}}$ in descending order, with $\boldsymbol{U_{\mathcal{A}}} \in \mathbb{R}^{n\times d}$ and $\boldsymbol{V_{\mathcal{A}}} \in \mathbb{R}^{(B+1)n\times d}$ matrices containing, as columns, corresponding orthonormal left and right singular vectors respectively.

The right spectral embedding is then given by $\hat{\boldsymbol{Y}} = \boldsymbol{V_A}|\boldsymbol{\Sigma_A}|^{\frac{1}{2}} \in \mathbb{R}^{n\times d}$ and the left spectral embedding by $\hat{\boldsymbol{X}}_{\boldsymbol{A}} = \boldsymbol{U_A}|\boldsymbol{\Sigma_A}|^{\frac{1}{2}} \in \mathbb{R}^{n\times d}$. However, we could define an alternative embedding as $\hat{\boldsymbol{Y}}_{alt} = \boldsymbol{V_A}\boldsymbol{\Sigma_A} = \hat{\boldsymbol{Y}}_{\boldsymbol{A}}|\boldsymbol{\Sigma_A}|^{\frac{1}{2}} \in \mathbb{R}^{n\times d}$.

The values of the diagonal matrix $\boldsymbol{\Sigma_A} \in \mathbb{R}^{d\times d}$ are in descending order, and give information about how much variance is explained by each dimension of the variance, as in Principle Component Analysis (PCA) [Pearson, 1901]. Thus by using $\hat{\boldsymbol{Y}}_{alt}$ in Algorithm 2 when using a spectral embedding to locate each node's k-nearest neighbours, the variance captured by each dimension is better represented, and the algorithm is less sensitive to the choice of d.

A.5 4 COMMUNITY STOCHASTIC BLOCK MODEL EXAMPLE

For simulated data, we generate a $C = 4$ community symmetric Stochastic Block Model (SBM) [Holland et al., 1983], an example of a BIRG, of $n = 1000$ nodes, where nodes are assigned to each community with equal random chance. The communities can be stored in a community allocation vector $\boldsymbol{\tau} \in \{1, ..., C\}^n$. The probability of an edge between two nodes depends on their respective community memberships. Specifically, we sampled from an SBM defined by the block probability matrix $\boldsymbol{B} \in [0,1]^{C\times C}$, where $\boldsymbol{B}$ is given by:

$$\boldsymbol{B} = \begin{bmatrix} 0.7 & 0.4 & 0.2 & 0.5 \\ 0.4 & 0.6 & 0.3 & 0.2 \\ 0.2 & 0.3 & 0.8 & 0.4 \\ 0.5 & 0.2 & 0.4 & 0.9 \end{bmatrix}. \tag{11}$$

An adjacency matrix $\boldsymbol{A} \in \{0,1\}^{n\times n}$ is sampled as:

$$\boldsymbol{A}_{ij} \overset{\text{ind}}{\sim} \text{Bernoulli}(\boldsymbol{B}_{\tau_i,\tau_j}), \tag{12}$$

where $\tau_i, \tau_j \in \{1, ..., C\}$ denote the communities that nodes i and j belong to respectively. A block model $\boldsymbol{B}$ is specified the same as for a MMSBM (see Section 4). By defining $\boldsymbol{B}$ to be symmetric, $\boldsymbol{B}_{g,h} = \boldsymbol{B}_{h,g}$ for all $g, h \in \{1, ..., C\}$, we have that $\mathbb{P}(\boldsymbol{A}_{ij}) = \mathbb{P}(\boldsymbol{A}_{ji})$ for all $i, j \in \{1, ..., n\}$, however this does not guarantee that $\boldsymbol{A}_{ij} = \boldsymbol{A}_{ji}$. To ensure symmetric samples, whenever $i > j$, i.e. $\boldsymbol{A}_{ij}$ falls into the lower triangle of $\boldsymbol{A}$, then set $\boldsymbol{A}_{ij} = \boldsymbol{A}_{ji}$ to ensure symmetry. In this way, we ensure that the resulting adjacency matrix $\boldsymbol{A}$ is symmetric.

As introduced in Section 3.1, we can use Algorithm 1 $M \in \mathbb{N}$ times to verify if a procedure for generating bootstrapped networks creates exchangeable networks. For this example we take $M = 200$ random samples from the model specified in Equation 12 to create the adjacency matrices $\boldsymbol{A}^{(1)}, ..., \boldsymbol{A}^{(M)}$. For each matrix one bootstrap replicate is created as in Algorithm 2 with $k = 5$ chosen, given by $\widetilde{\boldsymbol{A}}^{(1)}, ..., \widetilde{\boldsymbol{A}}^{(M)}$, respectively. To each pair $[\boldsymbol{A}^{(m)}, \widetilde{\boldsymbol{A}}^{(m)}]$, for $m = 1, ..., M$, we apply Algorithm 1, and use the M $\hat{p}$-values to create a QQ-plot. We repeat the above kNN-based bootstrap (as in Section 3.3) and change k to be $k = 25$ and $k = 240$, as well as applying other bootstrap methods (see Appendix A.6 for more details), with different embedding dimension d. In Figure 6 we show the QQ-plots and Bootstrap Validity Scores produced by each bootstrapping method.

In Appendix A.5.1, we perform a sensitivity analysis to demonstrate the procedure is robust to the choice of k for ASE-kNN, and principled in how it behaves.

A.5.1 Sensitivity of kNN-bootstrap Procedure to k

To demonstrate how the choice of k influences the performance of the kNN algorithm, we uses the $M = 200$ draws given by $\boldsymbol{A}^{(1)}, ..., \boldsymbol{A}^{(M)}$ from the $n = 1000$ node 4 community SBM with block probability matrix $\boldsymbol{B}$ given in Equation 11

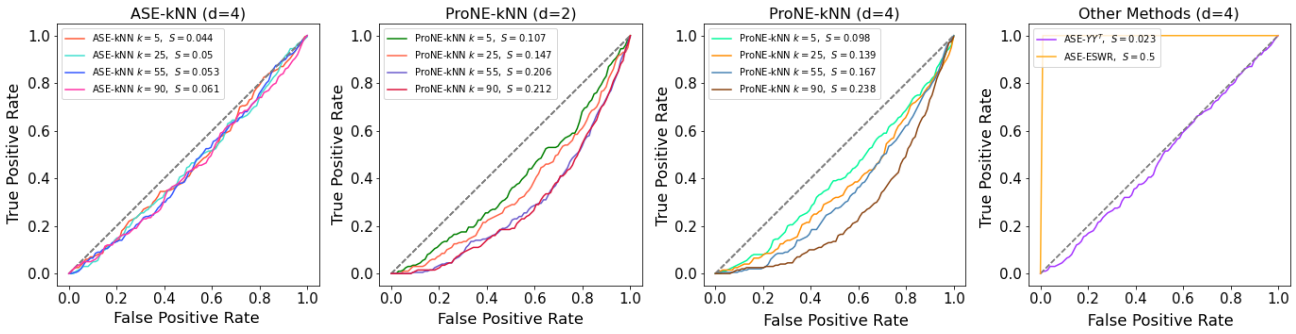

Figure 6: Simulated results for the $n = 1000$ node 4 community SBM. Each curve is found using $M = 200$ draws from the known model, each paired with a single bootstrap from the model. The area between the curve and the $x = y$ line is the Bootstrap Validity Score.

introduced in Section A.5. For all values of $k \in \{2, ..., \frac{n}{2}\}$, each of the M matrices are bootstrapped once as in Algorithm 2 (ASE-kNN) with $d = 4$, giving M pairs for each k value. To each pair $[\boldsymbol{A}^{(m)}, \widetilde{\boldsymbol{A}}^{(m)}]$, for $m = 1, ..., M$, we apply Algorithm 1, and use the M p-values to create a QQ-plot. The Bootstrap Validity Score is calculated for each value of $k \in \{2, ..., \frac{n}{2}\}$, and gives a measure of how well the bootstrap algorithm performs with this data example for different choices of k. In Figure 7 the choices of k are plotted against their scores. By observing the score values, we see that there is a wide range of choices of k that the algorithm performs well for with this data. Since in this example we know that $n = 1000$ for 4 communities, where all communities have equal probability of a node belonging to it (i.e. the expected size of each community is 250), we see the algorithm begins to perform poorly when k is chosen to be larger than the size of the smallest community in an observed network.

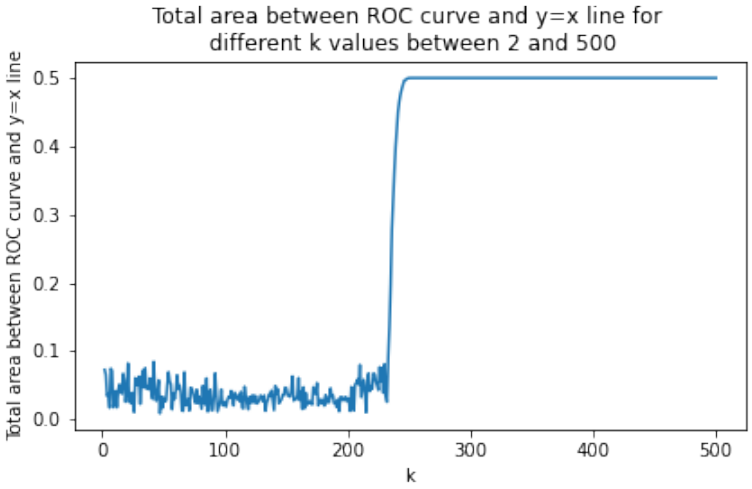

Figure 7: Plot showing possible choices of k against the 'Bootstrap Validity Score' (area between the curve and the $x = y$ line) for a 4 community SBM model of $n = 1000$ nodes. The smaller the score value, the better the bootstrap method performs at producing exchangeable networks. From the plot we see that choices of $k < 200$ all have reasonable score values. Around the $k = 230$ mark the score values increase rapidly and plateau at a Bootstrap Validity Score of 0.5. As each of the $n = 1000$ nodes are equally likely to belong to each of the 4 communities, the expected size of each community is 250. The kNN-based algorithm performs poorly when k is chosen larger than the size of the smallest community, but this is sensible, as a node is not well represented by a node outside of its own community in this SBM setting.

A.6 STATEMENT OF CONSIDERED BOOTSTRAP ALGORITHMS

A naïve way to bootstrap a network is to consider the edge list E of adjacency matrix $\boldsymbol{A} \in \{0, 1\}^{n \times n}$, and sample $|E|$ edges with replacement. As we consider binary graphs, any edge sampled more than once will have its duplicates removed from the final sample. The resulting edge list $\widetilde{E}$ will populate adjacency matrix $\widetilde{\boldsymbol{A}} \in \{0, 1\}^{n \times n}$. This is given by Algorithm 4.

Algorithm 3 $\boldsymbol{X}\boldsymbol{X}^T$ Bootstrap of an Unweighted Network

Input:
  Observed adjacency matrix $\boldsymbol{A} \in \{0,1\}^{n \times n}$
  Embedding dimension $d \leq n$
  Number of bootstrapped graphs B
Compute the d-dimensional adjacency spectral embedding $\hat{\boldsymbol{X}}_{\boldsymbol{A}}$ of $\boldsymbol{A}$
Set $\hat{\boldsymbol{P}} = \hat{\boldsymbol{X}}_{\boldsymbol{A}} \hat{\boldsymbol{X}}_{\boldsymbol{A}}^T$
Set any values in $\hat{\boldsymbol{P}} < 0$ to be 0, and set any values in $\hat{\boldsymbol{P}} > 1$ to be 1
for $b = 1, \ldots, B$ **do**
  Sample

$$\widetilde{\boldsymbol{A}}_{ij}^{(b)} \overset{\text{ind}}{\sim} \text{Bernoulli}(\hat{\boldsymbol{P}}_{ij})$$

end for
Output: $\widetilde{\boldsymbol{A}}^{(1)}, \ldots, \widetilde{\boldsymbol{A}}^{(B)}$

Algorithm 4 Edge List Sample with Replacement (ESWR) Bootstrap of an Unweighted Network

Input:
  Observed adjacency matrix $\boldsymbol{A} \in \{0,1\}^{n \times n}$
  Number of bootstrapped graphs B
Create the edge list E of edges in $\boldsymbol{A}$
for $b = 1, \ldots, B$ **do**
  Sample with replacement $|E|$ edges from E to give an edge list $\widetilde{E}^{(b)}$
  Update $\widetilde{E}^{(b)}$ by removing any duplicate edges
  Use $\widetilde{E}^{(b)}$ to construct a binary adjacency matrix $\widetilde{A}^{(b)}$
end for
Output: $\widetilde{\boldsymbol{A}}^{(1)}, \ldots, \widetilde{\boldsymbol{A}}^{(B)}$

We can extend Algorithm 4, by adding random edges to the bootstrapped $\widetilde{\boldsymbol{A}}^{(b)}$ for $b = 1, ..., B$ outputted by Algorithm 4, such that all bootstrap resamples have the same number of edges as the observed adjacency matrix $\boldsymbol{A}$. See Algorithm 5.

Algorithm 5 Edge List Sample with Replacement Bootstrap + Random Edges of an Unweighted Network

Input:
  Observed adjacency matrix $\boldsymbol{A} \in \{0,1\}^{n \times n}$
  Number of bootstrapped graphs B
Create the edge list E of edges in $\boldsymbol{A}$
for $b = 1, \ldots, B$ **do**
  Sample with replacement $|E|$ edges from E to give an edge list $\widetilde{E}^{(b)}$
  Update $\widetilde{E}^{(b)}$ by removing any duplicate edges
  Find $a \in \mathbb{N}_0$ such that $|\widetilde{E}^{(b)}| + a = |E|$
  Sample without replacement a edges from $(\widetilde{E}^{(b)})^c$ and add these to $\widetilde{E}^{(b)}$
  Use $\widetilde{E}^{(b)}$ to construct a binary adjacency matrix $\widetilde{\boldsymbol{A}}^{(b)}$
end for
Output: $\widetilde{\boldsymbol{A}}^{(1)}, \ldots, \widetilde{\boldsymbol{A}}^{(B)}$

A.7 MMSBM - EFFECTS OF CHANGING N TO THE BOOTSTRAP VALIDITY SCORE WITH DIFFERENT METHODS

We look to see how the network size n changes the Bootstrap Validity Score for different methods in Figure 8. We see for this synthetic data example, ASE-XX^T performs worse than all other methods we tried. We apply the kNN bootstrap procedure with ASE into $d = 3$ dimensions (as in Algorithm 2) and use ProNE embedding (into $d = 2$ dimensions as Figure 3 showed $d = 2$ performed better than $d = 3$ with ProNE) with kNN bootstrapping. For the same values of k, the ASE-kNN and ProNE-kNN perform fairly similarly across all values of n, as shown by Figure 8. We see for $k = 90$, both methods

yield a fairly high Bootstrap Validity Score, likely that k is chosen too large. For $k = 5$ and $k = 25$ we see fairly similar performance, however when $n > 750$, it appears that $k = 25$ is a better choice than $k = 5$ for both embedding method choices.

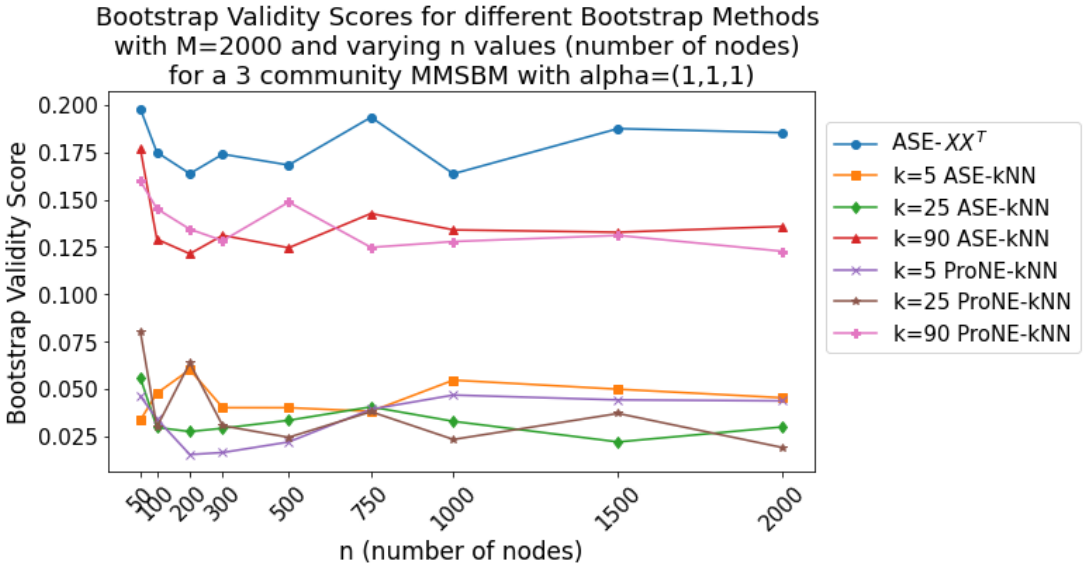

Figure 8: MMSBM, $C = 3$ communities, with $\boldsymbol{\alpha} = \mathbf{1}_C$ for different values of n and different embedding methods and bootstrap procedures applied.

A.8 OVERVIEW OF T-SNE

t-SNE (t-distributed Stochastic Neighbour Embedding) was introduced in Van der Maaten and Hinton [2008] as a tool for visualising high-dimensional data in two or three dimensions. In the original high-dimensional space, pairwise similarities between data points are calculated using a probability distribution, which preserves local structures by assigning higher probabilities to points that are close together. t-SNE tries to replicate the high-dimensional similarities in a low-dimension space by optimising the Kullback-Leibler (KL) divergence [Kullback and Leibler, 1951] between the high-dimension and low-dimension probability distributions, using gradient descent.

t-SNE is good at clustering and separating groups with different local relationship structure. However, it does not preserve global structures well. The algorithm can separate clusters well, but where the clusters are embedded in the low-dimensional t-SNE space is not necessarily representative of how similar certain clusters are in behaviour, or where there are across cluster local relationships. The perplexity hyperparameter, which controls the balance between local and global aspects of the data, thus needs tuning carefully. It is also of note that t-SNE is non-deterministic, so results may vary across runs. A fixed seed can be used to create reproducible t-SNE embeddings.

A.9 SENSITIVITY ANALYSIS CHECKS OF EMBEDDING OF THE SCHOOL DATA EXAMPLE

To choose k in the kNN model, we computed the Bootstrap Validity Score for each $k \in \{2, ..., 12\}$. We find that k being too small or too large leads to poor bootstraps, but in between there is tolerance between $k = 3$ and 8 (Figure 9). So as not to overfit, we choose the middle value $k = 5$, even though $k = 3$ or 7 are possibly better. For both ProNE and ASE, this leads to very similar $\hat{\boldsymbol{P}}$ values (Figure 10).

When estimating the covariance of each node in the d dimensional embedding space created via UASE, we have B bootstrap embeddings and 1 observed embedding to compute the estimate. Here we show that the method is not sensitive to the number of bootstraps used, by plotting the t-SNE embedding of $\boldsymbol{A}^{(obs)}$, with the t-SNE perplexity value fixed at perplexity = 125. We set $B = 25, 50, 100, 150, 200, 250, 300, 400, 500$ in our sensitivity analysis. We set $d = 10$ as the ASE embedding dimension and $k = 5$ for the kNN to compute $\hat{\boldsymbol{P}}$. The t-SNE embeddings are standardised to be mean 0, standard deviation 1. Figure 11 shows that the method is robust to changing the value of B, the number of bootstraps, as the lines between

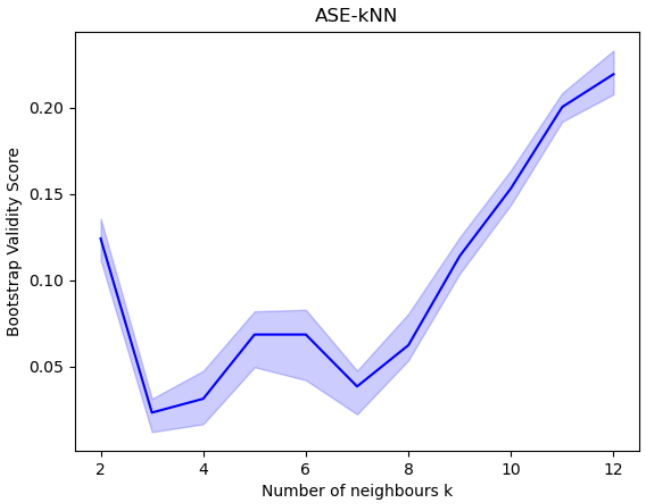

Figure 9: Bootstrap Validity Score for the School Data averaged over 20 runs, with 90% confidence intervals, for $k \in \{2, ..., 12\}$.

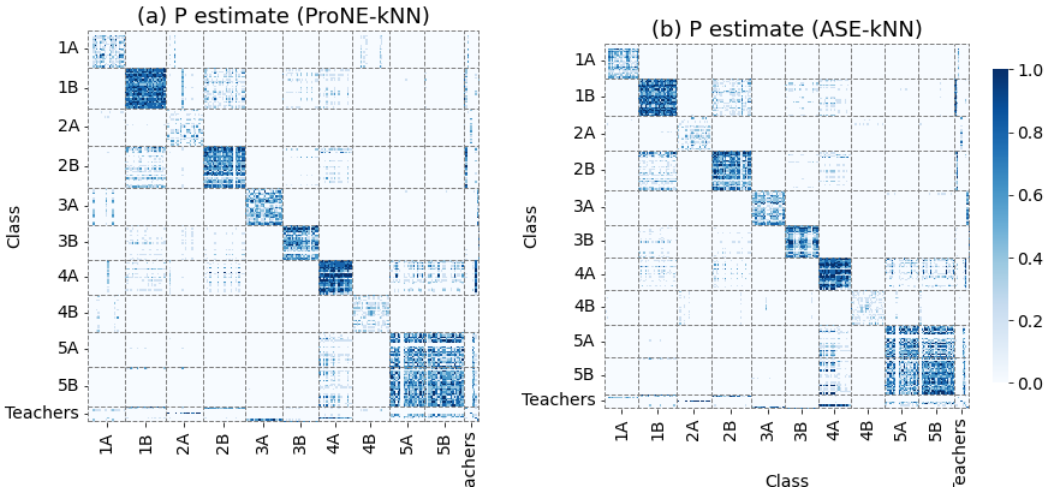

Figure 10: $\hat{\boldsymbol{P}}$ for the School Data ASE-kNN ($d = 10$) or ProNE-kNN ($d = 6$) embedding (both $k = 5$). Both are very similar, having passed the Bootstrap Exchangeability Test.

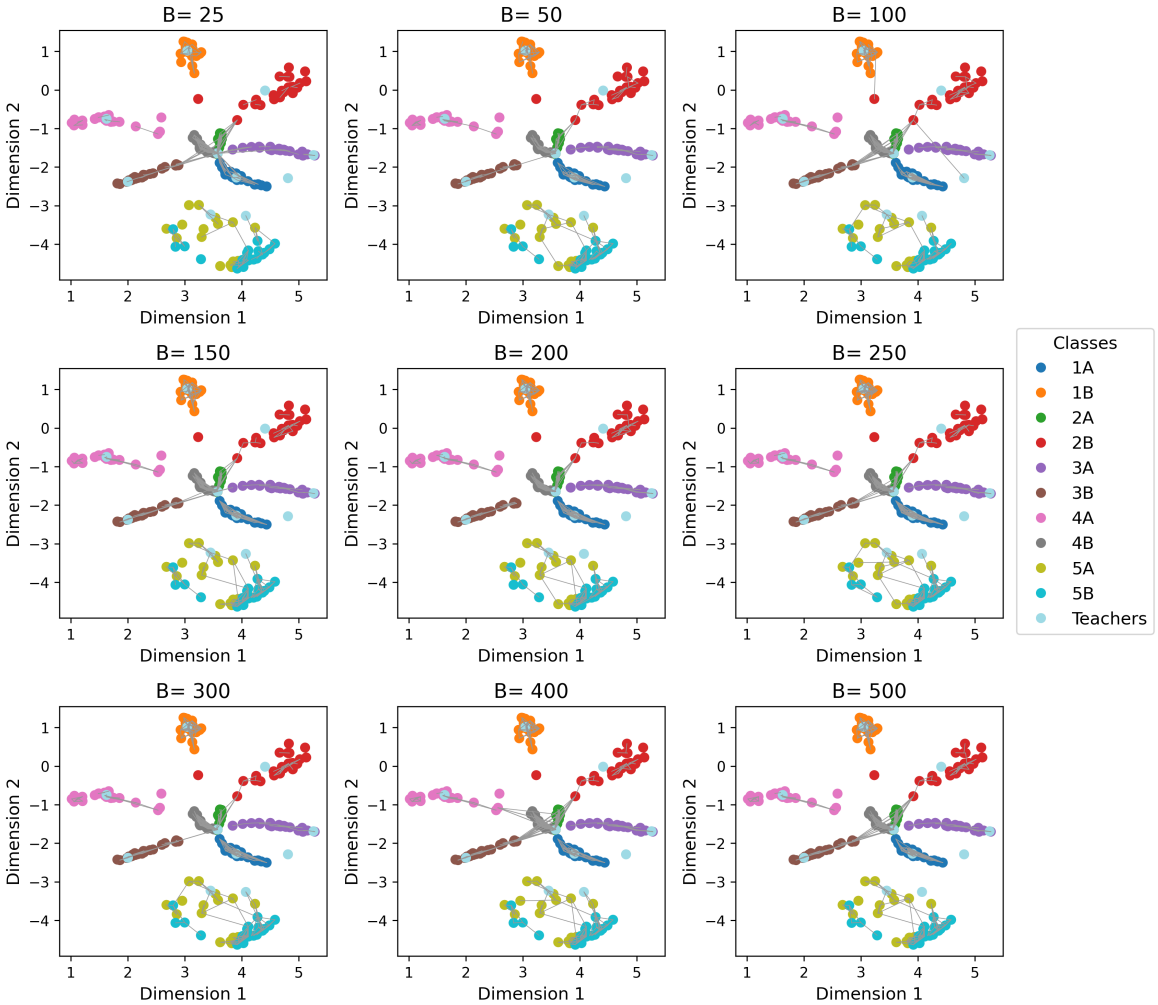

Figure 11: Sensitivity to B shown on a shared t-SNE visualisation applied to the observed adjacency matrix with Perplexity 125, $\boldsymbol{A}^{(obs)}$ is shown. The edges connecting nodes are from within 3 SDs of one another in the d dimensional embedding space using ASE-kNN ($k = 5$, $d = 10$, varied B).

clusters are stable. For each value of B, the "fuzziness" matrix $\boldsymbol{F}$ is calculated, and a line is plotted between nodes i and j when $\boldsymbol{F}_{ij} = 1$. We have $\boldsymbol{F}_{ij} = 1$ when the embeddings of nodes i and j in the d dimensional embedding space are both within 3 standard deviations of one another, using the B bootstrap embeddings alongside the observed network's embedding to estimate the covariance of each node.

The t-SNE algorithm has a parameter called perplexity, which can be interpreted as a smooth measure of the effective number of neighbours used by the algorithm to calculate a nodes similarity to other nodes [Van der Maaten and Hinton, 2008]. For a fixed number of bootstraps, $B = 500$, we calculate the covariance of each node, and thus the "fuzziness" matrix $\boldsymbol{F}$. A systematic scan of perplexity is shown in Figure 12, summarised with the Fuzziness score. We then visualise the corresponding t-SNE of $\boldsymbol{A}^{(obs)}$ for perplexity $= 15, 25, 35, 55, 75, 95, 125, 155, 185$, with edges from $\boldsymbol{F}$, in Figure 13. The method is robust to perplexity choice, however when perplexity is chosen to be larger than the largest community size, the t-SNE algorithm will certainly explore relationships outside of the cluster each node belongs to, which is valuable.

For the school data example, each class had between 22 and 26 students, with one teacher per class [Stehlé et al., 2011]. The largest cluster size is therefore 27, if we consider the largest class and their teacher. When perplexity is increased to a value larger than 27, we see that the t-SNE algorithm does a better job at separating nodes, especially near the origin. In Figure 5 we can see that when perplexity is increased from 15 to 125 the algorithm does a much better job at clustering the data, as it is able to look outside of the members of the cluster to explore the global data structure, yielding a better visualisation.

Our uncertainty estimates are relatively robust to the choice of $\boldsymbol{P}$ estimate. Figure 14 shows the inferred $\boldsymbol{F}$ matrices using

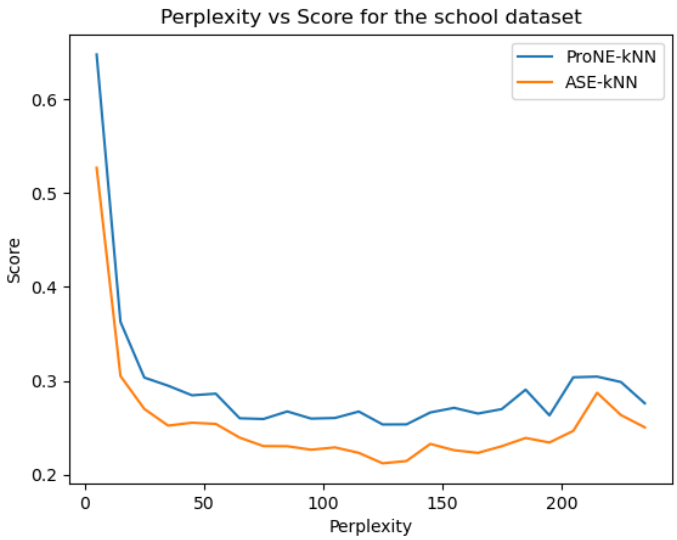

Figure 12: Fuzziness Score as a function of t-SNE Perplexity for the School Data ASE-kNN ($d = 10$) or ProNE-kNN ($d = 6$) embedding (both $k = 5$). Both methods agree that 'good' values are around 125 with a wide tolerance of ± 75.

the two best bootstraps, either ProNE-kNN (d=6) or ASE-kNN (d=10). Whilst differences exist, these are within-cluster structures and the important cross-cluster features are present similarly in both.

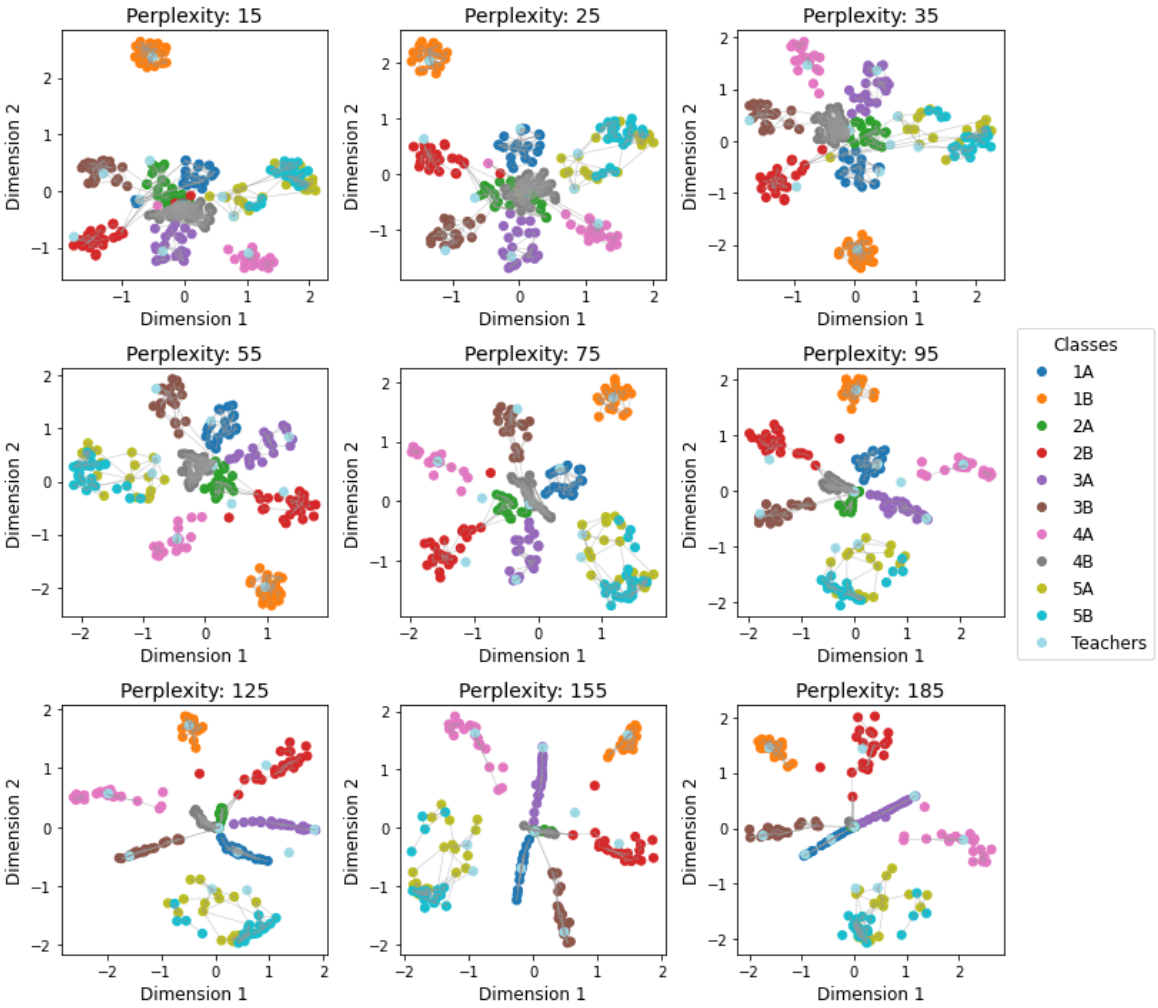

Figure 13: Plots of t-SNE applied to the observed adjacency matrix $\boldsymbol{A}^{(obs)}$ for different perplexity values. For $B = 500$ bootstraps the "fuzziness" matrix $\boldsymbol{F}$ is calculated, and a line is plotted between nodes i and j if $\boldsymbol{F}_{ij} = 1$, i.e. nodes i and j have overlapping distributions. We see the method is robust to different choices of perplexity.

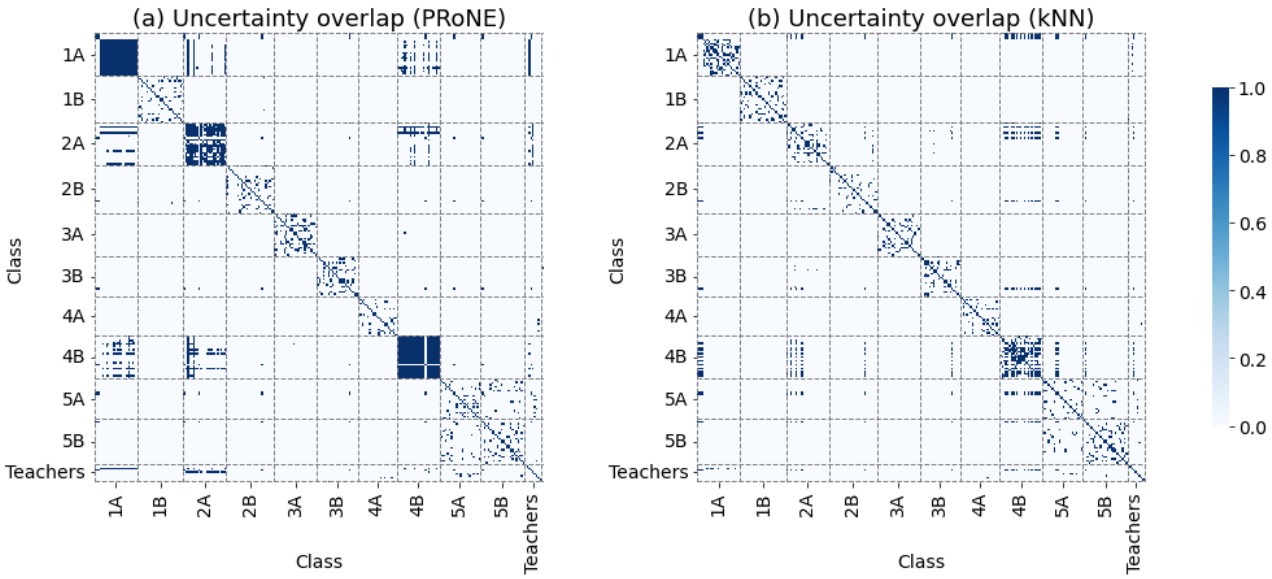

Figure 14: Visualisation of the "fuzziness" matrix $\boldsymbol{F}$ calculated using $B = 500$ bootstraps, for ProNE-kNN ($d = 6$) and ASE-kNN ($d = 10$). Where $\boldsymbol{F}_{ij} = 1$, nodes i and j are within 3 SDs of one another, i.e. the nodes have overlapping distributions. Both methods highlight the same within cluster and across cluster structures.