Twinning Complex Networked Systems: Data-Driven Calibration of the mABCD Synthetic Graph Generator

Piotr Bródka[1,2], Michał Czuba[1,2⋆], Bogumił Kamiński[3], Łukasz Kraiński[3], Katarzyna Musial[2], Paweł Prałat[4], and Mateusz Stolarski[1]

[1] Dept. of Artificial Intelligence, Wrocław University of Science and Technology, Wrocław, Poland
[2] Data Science Institute, University of Technology Sydney, Sydney, Australia
[3] Decision Analysis and Support Unit, SGH Warsaw School of Economics, Warsaw, Poland
[4] Dept. of Mathematics, Toronto Metropolitan University, Toronto, Canada

Abstract. The increasing availability of relational data has contributed to a growing reliance on network-based representations of complex systems. Over time, these models have evolved to capture more nuanced properties, such as the heterogeneity of relationships, leading to the concept of multilayer networks. However, the analysis and evaluation of methods for these structures is often hindered by the limited availability of large-scale empirical data. As a result, graph generators are commonly used as a workaround, albeit at the cost of introducing systematic biases. In this paper, we address the inverse-generator problem by inferring the configuration parameters of a multilayer network generator, mABCD, from a real-world system. Our goal is to identify parameter settings that enable the generator to produce synthetic networks that act as digital twins of the original structure. We propose a method for estimating matching configurations and for quantifying the associated error. Our results demonstrate that this task is non-trivial, as strong interdependencies between configuration parameters weaken independent estimation and instead favour a joint-prediction approach.

Keywords— Complex Systems, Digital Twins, Inverse Graph Modelling, mABCD, Multilayer Networks.

1 Introduction

Complex networked systems are typically represented by graphs constructed from relational data. This representation facilitates the encapsulation of topological characteristics, such as relationship structures, connectivity patterns, or node centralities, which are not readily apparent at first glance [8].

To accurately characterise dynamics in complex networked systems, it is necessary to adopt an approach that differentiates between the various types of relationships present. A natural way to achieve this is through the use of *multilayer networks* [15], a system defined as a collection of ℓ layers, where each layer represents a distinct mode of

⋆ Corresponding author: michal.czuba@pwr.edu.pl

interaction encoded by its corresponding graph. The resulting dynamics are governed not only by intra-layer edges, describing interactions within a given relationship type, but also by inter-layer couplings, which connect the same entity across different layers. This structure enables modeling of complex spreading mechanisms that are obscured in monoplex representations [6].

Understanding the topology of a system and the interplay between the relationships it contains is critical for designing effective intervention policies. For instance, in simplified models of human interactions, immunisation strategies (such as node removal or fact-checking) typically target high-degree hubs [21]. However, in multilayer systems, the most influential actors are often not the hubs within individual layers, but rather multiplex bridges, that is, nodes with moderate intra-layer connectivity but high "participation coefficients" across layers. These actors facilitate the spillover of misinformation from fringe, weakly moderated communities (e.g., 4chan) into mainstream discourse (e.g., X or Facebook). Consequently, observing the system through a multilayer lens reveals that resilience in one layer can mask catastrophic fragility in the coupled system. A misinformation cascade that appears sub-critical (dying out) on one platform may be sustained by activity on another, creating a hysteresis loop where the false belief becomes endemic despite platform-specific moderation efforts.

While empirical analysis of real-world datasets is foundational, it faces severe limitations when dissecting the complex dynamics of multilayer systems. Real-world multilayer data is often sparse, proprietary, or plagued by missing links, and crucially, it represents only a single, static realisation of a dynamic process. One cannot "rewind" the internet to see how a misinformation campaign would have unfolded if the user base were 50% larger or if the moderation algorithm were different. To overcome these constraints, researchers must turn to synthetic network generation, that is, creating artificial yet statistically realistic graphs that serve as controllable laboratories for simulation. The primary utility of these synthetic environments lies in their flexibility. By employing generative random graph models, one can systematically vary topological parameters to isolate their effects on dynamical processes.

This approach culminates in the concept of the "*digital twin*" [23]. Digital twins for complex networked systems provide a dynamic virtual counterpart to real-world networks, enabling continuous monitoring, simulation, and prediction of system behaviour. Their defining strength lies in real-time data exchange between the physical system and its digital representation, allowing the model to evolve as the network evolves. By integrating simulation, optimisation, data analytics, and machine learning, digital twins capture both the structure and the dynamics of interconnected systems such as social networks, cyber-physical systems, and blockchain-based architectures. This makes them uniquely suited to exploring scenarios that have never occurred, detecting anomalies, and supporting informed decision-making in complex, distributed environments. Even on the simplest level of modelling, a digital twin is not merely a random graph, but a synthetic replica carefully calibrated to match the specific statistical properties of a target real-world system (such as its degree distribution and distribution of community sizes, etc.). Therefore, the modelling paradigm they convey makes them a powerful approach for understanding and managing complex systems.

While the utility of "digital twins" is clear, constructing one that faithfully mimics a specific real-world system remains a formidable mathematical challenge. This is fundamentally an inverse problem: given an observed set of real-world statistics, one must infer the precise combination of generative parameters that produces a matching graph. In multilayer systems, this difficulty is compounded by the curse of dimensionality. A model rich enough to capture the nuance of real systems, accounting for intra-layer topology and inter-layer interdependence, inevitably possesses a vast parameter space. Developing robust, algorithmic frameworks to automatically calibrate these high-dimensional synthetic models to empirical data is not just a technical refinement; it is a necessary frontier for making network science predictive rather than merely descriptive.

In this paper, we aim to shed light on this problem by proposing a modular approach for inferring the parameters of the mABCD generator from an observed real-world network. In particular, we investigate whether decomposing the task into a sequence of analytical and optimisation sub-problems constitutes a viable strategy. Our results indicate that the proposed approach establishes a strong baseline that could be difficult to surpass. Nevertheless, the experiments suggest that jointly predicting all configuration parameters within a single optimisation procedure may yield superior performance, which delineates an interesting yet challenging new research direction.

The remainder of the paper is organised as follows. Sec. 2 briefly describes the fundamental concepts underpinning the work; in particular, operating principles of the mABCD model. Sec. 3 presents the parameter estimation framework and introduces the discrepancy measures used to quantify approximation error. Sec. 4 discusses the experimental results, while Sec. 5 concludes the paper.

2 Preliminaries

In order to address the problem studied in this paper, we first introduce the fundamental concepts used throughout. In particular, we briefly review multilayer networks, outline the general concept for the configuration inference task, and describe the synthetic graph generator employed.

2.1 Multilayer networks

As stated in the introduction, heterogeneity of complex networked systems is a crucial property which cannot be neglected. Therefore, the problem tackled in this work aims to take into account this property by utilising multilayer networks. Following [15], we formalise the concept of a multilayer network below. Nevertheless, before we do this, we clarify that for a given $n \in \mathbb{N} = \{1, 2, \ldots\}$, $[n]$ is used to denote the set consisting of the first n natural numbers, that is, $[n] = \{1, 2, \ldots, n\}$.

Definition 1 (Multilayer network). *A multilayer network can be described as a quadruple $G = ([n], [\ell], V, E)$, where $[n]$ is a set of n actors, $[\ell]$ is a set of layers, $V \subseteq [n] \times [\ell]$ is a set of nodes, $E \subseteq \bigcup_{i \in [\ell]} [V_i \times V_i]$ is a set of edges.*

When analysing Def. 1, one can note that it is in line with the understanding of multilayer networks as collections of graphs, where each layer $G_i = (V_i, E_i)$ is spanned on a subset of actors existing in the system (v_i) within the relationship (layer) $i \in [\ell]$. For instance, in a system comprising interactions on various social platforms, $[n]$ would denote humans active on the Internet, $[\ell]$ social platforms, V would be a set of all accounts in the systems, i.e. nodes, where node $v = (a, \ell_i) \in V$ represents an actor a in layer ℓ_i, e.g., a LinkedIn profile of a citizen John Smith. On the other hand, the network could be decomposed into ℓ interdependent graphs from different social systems.

2.2 Complex Networked System Analysis with Twinning Approach

Utilising the example already introduced in Sec. 1, a conceptual framework guiding the problem is presented here. Suppose one is to evaluate several "what-if" scenarios that cover interventions on a social network to design effective strategies to control misinformation spread, given an initial dataset of ground truth interactions on the platform.

On the grounds presented in Sec. 1, a twin-based framework can be introduced, as shown in Fig. 1. It assumes that the analysed real-world multirelational system is encoded into a configuration file for a synthetic graph generator, which can then be used for further modelling. By altering selected parameters of the generator, one can explore a variety of what-if scenarios and construct adjusted "digital twins" that reflect possible modifications of the original system. These can be evaluated under the assumed spreading regime, allowing an assessment of which topology is optimal from the perspective of a given influence control task (e.g., maximisation). For instance, one can rigorously test hypotheses by scaling the network size to observe asymptotic behaviours, increasing edge density to identify percolation thresholds, or introducing additional layers to simulate the emergence of new communication channels. This capability is essential for establishing causal links between network structure and process outcomes, something that is impossible with observational data alone.

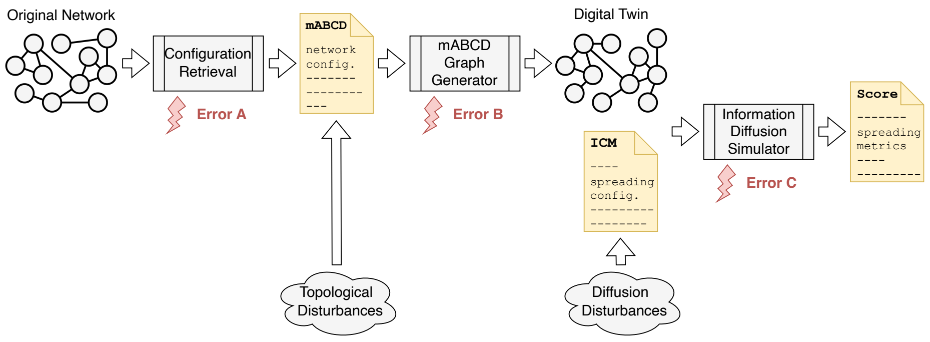

Fig. 1: Spread control pipeline utilising the configuration retrieval approach to create adjusted twins of the evaluated system. The process begins by determining the parameters of `mABCD` that match the analysed system. Its digital twin is then generated and evaluated under the appropriate dynamics. Note that the pipeline's efficacy depends on addressing inherent errors.

The most important factor affecting the validity of the approach involves three types of errors. The first one arises in the *configuration retrieval* process. For models allowing rich modelling space (like `mABCD`), this task is non-trivial, as some parameters cannot be directly inferred from the network and need to be estimated (e.g., a community breakdown). The second type of error relates to imperfections of the synthetic graph generator, as the produced network may deviate from the specified configuration. Finally, the third type of error arises in the simulation of information diffusion, since spreading models remain only approximations of real-world processes. Addressing these errors is essential to ensure the pipeline's correctness. Given the size of the task, we aim to address the first of the presented errors in this work.

Configuration retrieval can be carried out in different ways depending on needs and availabilities. The most natural approach is to estimate each parameter independently by designing appropriate algorithms and metrics to quantify estimation quality. This, however, is not the only method. Another, perhaps complementary, natural way of measuring the "distance" between the original network G and its produced twin $\hat{G}$ would be to embed both graphs in the same high-dimensional space and use the distance between the two representations to measure the quality (*Open Problem 1*). Given the recent advancements in machine learning, this approach also feels natural, and some fast algorithms were recently introduced [7]. Nevertheless, approaching the problem without trying the more straightforward methods deprives us of a reliable baseline. Therefore, this paper tackles the problem from the former position.

2.3 The Synthetic Network Generators — mABCD

Synthetic graph generators with explicit community structure play a central role in the development and evaluation of analysis methods for complex networked systems, particularly in settings where large-scale empirical datasets with reliably annotated ground-truth communities are scarce. To address this limitation, several benchmark models have been proposed to generate artificial networks that resemble real-world systems while retaining full control over their structural properties. A prominent example is the LFR benchmark model [18,17], which allows for heterogeneous degree distributions and community sizes and has become a standard tool for evaluating community detection algorithms.

More recently, the Artificial Benchmark for Community Detection (ABCD) model [12] has been introduced as a scalable [10] and theoretically tractable alternative to LFR. Undirected variants of ABCD generate networks with comparable structural properties, while offering improved computational performance and greater flexibility in interpolating between well-defined community structure and random graphs [10]. Owing to these properties, ABCD has been extensively analysed from a theoretical perspective, including studies of its modularity behaviour [11] and self-similarity properties [3]. The modular design of the model has further enabled a range of extensions, including outliers [13], overlapping communities [2], hypergraphs [14], and, most relevant to this work, multilayer networks [16]. The multilayer extension, mABCD, generalises the ABCD framework to systems in which multiple interaction layers jointly shape network dynamics. Rather than aiming to exactly reproduce a specific empirical network, mABCD is intended to generate ensembles of multilayer graphs that preserve key structural characteristics of a target system, and can therefore be interpreted as coarse-grained digital twins of complex networked systems.

In this work, we adopt mABCD as the synthetic generator underlying our experimental analysis. Therefore, to focus on the aspects most relevant to the configuration inference problem considered here, a brief review of its core principles is necessary. In principle, the mABCD network generation comprises of six subsequent phases. The process is governed by global and local (independent for each layer) parameters (Tab. 1). which affect various properties of the model, see Sec. 3 for details.

The first five steps are carried out independently for each of the ℓ layers, while the sixth one binds the generated graphs into a single multilayer network. In the first phase, the so-called active nodes (i.e., those that will not be isolated) are determined.

Table 1: Global and local parameters of `mABCD`.

Parameter	Range	Description
Global parameters		
n	$\mathbb{N}$	Number of actors
ℓ	$\mathbb{N}$	Number of layers
d	$\mathbb{N}$	Dimension of reference layer
$\mathbf{R}$	$[0,1]^{\ell\times\ell}$	Inter-layer edge correlation matrix
Local parameters (independent for each layer)		
q_i	$(0,1]$	Fraction of active actors
τ_i	$[-1,1]$	Correlation coefficient between degrees and labels
r_i	$[0,1]$	Correlation strength between communities and the reference layer
γ_i	$(2,3)$	Power-law degree distr. with exponent γ_i
δ_i	$\mathbb{N}$	Min. degree at least δ_i
Δ_i	$\mathbb{N}$ $(1 \leq \delta_i \leq \Delta_i < n)$	Max. degree at most Δ_i
β_i	$(1,2)$	Power-law community size distr. with exponent β_i
s_i	$\mathbb{N}$	Min community size at least s_i
S_i	$\mathbb{N}$ $(\delta < s_i \leq S_i \leq n)$	Max community size at most S_i
ξ_i	$(0,1)$	Level of noise

Each active node is endowed its degree in the second step. During the third step, communities are created and populated with nodes based on the latent layer, which is implicitly generated by the routine to serve as a proxy for actors' features that typically shape connections in complex networked systems. Phase four focuses on connecting nodes within other members of the same community as well as establishing inter-community (i.e., background) connections. These graphs are subsequently simplified to remove possible self-loops and multiedges while preserving the intended structural properties. The final phase performs a series of edge rewirings to achieve the targeted edge correlations between each pair of layers. For a detailed description, please see [16].

3 Proposed `mABCD` Configuration Retrieval Method

Suppose that we are given a multilayered network following Def. 1. Our goal is to generate a digital twin, a multilayered network $\hat{G}$ generated by `mABCD` with associated graphs $\hat{G}_i = (\hat{V}_i, \hat{E}_i)$ for $i \in [\ell]$, that mimics the original network G as best as possible. In order to do it, one needs to appropriately select the parameters of the `mABCD` model and then measure the "success", i.e., how close G to $\hat{G}$ is. We will measure the success by computing a *divergence score* $\mathcal{D}(\hat{G}, G)$; the smaller the score, the better the fit is obtained.

As already noted, `mABCD` consists of several parameters, but not all of them are used directly to model the target network's properties. Some should be treated rather like hyperparameters, by manipulating which the accuracy of transforming a given configuration setup into the final network can be controlled. These were intentionally omitted in Sec. 2.3, as they are out of our interest due to a loose connection with the structural properties of the produced network, but we acknowledge them to show the complexity of the problem. In the proposed approach, we fix them to the values recommended in [16].

The remaining mABCD parameters (Tab. 1) explicitly provide expected structural properties of the network and can be grouped by their proximity. Based on the observation, we propose a method that can independently predict each parameter group by leveraging an algorithmic approach. The only exemption is r, which cannot be feasibly estimated as it employs the latent layer used to model actor features — neither returned by the model nor available in real-world networks. That is why a solution-search-browsing technique was employed, i.e., the Bayesian optimisation.

Parameters such as n, ℓ, δ_i, and Δ_i are trivial to extract from G. Similarly, the fraction of active actors is easy to obtain: $q_i = |V_i|/|\bigcup_{i\in[\ell]} V_i|$.

Dimension of reference layer (d). Communities in mABCD are independently generated in each layer, but the desired correlations are obtained via a hidden, low-dimensional geometric reference layer. Nodes close to each other in the reference layer end up in the same community with a larger probability. In particular, the dimension of a reference layer d affects many properties of the generated network. However, it seems not so easy to guess the right value so we leave it as a parameter that eventually can be tuned for the best outcome, that is, the smallest divergence score $\mathcal{D}(\hat{G}, G)$. Understanding the influence of this hyperparameter and selecting the optimal value is left as an open problem (*Open Problem 2*).

This geometric approach to community structure is justified by the fact that latent geometric spaces are widely believed to shape complex networks (e.g., social media networks shaped by users' opinions, education, knowledge, interests, etc.). These latent spaces have been successfully employed for many years to model and explain network properties such as self-similarity [20], homophily, and aversion [9]. For more details, we direct the reader to the survey [4] or the book [19].

Inter-layer edge correlation (R). To measure correlations between edges in different layers, we define $\mathbf{R}$, a $\ell \times \ell$ matrix in which elements $r_{i,j} \in [0,1]$ $(i,j \in [\ell])$ capture correlation between edges present in layers i and j. For any $i,j \in [\ell]$, let E_i^j be the set of edges that are present in layer i, involving actors that are also active in layer j. Entries $r_{i,j}$ in $\mathbf{R}$ are computed using the following formula:

$$r_{i,j} = \frac{|E_i^j \cap E_j^i|}{\min\{|E_i^j|, |E_j^i|\}}.$$

If $\min\{|E_i^j|, |E_j^i|\} = 0$, then we leave $r_{i,j}$ undefined.

Note that $r_{i,i} = 1$ for any $i \in [\ell]$ and $r_{i,j} = r_{j,i}$ for $1 \le i < j \le \ell$. The maximum value of 1 is attained when edges in one of the layers form a subset of edges in the other layer. The minimum value of 0 is attained when the two sets of edges in the corresponding layers are completely disjoint.

Extracting $A = \mathbf{R}$ from the original network G is straightforward. The mABCD model aims to produce a synthetic network with the desired edge correlation matrix A, but this is not guaranteed, and some small error always occurs. Let B be the corresponding edge correlation matrix for the obtained digital twin $\hat{G}$. The divergence score is simply the normalised to $[0,1]$ Frobenius norm between the two matrices:

$$\mathcal{D}_{\mathbf{R}}(\hat{G}, G) = \frac{\|A-B\|_F}{\sqrt{\ell(\ell-1)}} = \sqrt{\frac{1}{\ell(\ell-1)} \sum_{i,j\in[\ell]} (a_{ij} - b_{ij})^2} = \sqrt{\frac{1}{\binom{\ell}{2}} \sum_{1\le i<j\le\ell} (a_{ij} - b_{ij})^2}.$$

Correlation coefficient between degrees and labels (τ_i). This family of parameters models correlations between sequences of node degrees that are active in two different layers. The `mABCD` model assumes that the labels of actors and nodes representing them are natural numbers. The parameter $\tau_i \in [-1, 1]$ models the correlation between these labels and the corresponding degree sequence in layer i.

To estimate τ_i, which relies on the ordering of actor labels, we first sort nodes with respect to their total degree (sum of degrees over all layers), breaking ties randomly if needed; that is, the node with the highest total degree is assigned label n, the highest possible label. Then, for any $i \in [\ell]$, we compute the Kendall rank correlation coefficient τ_i between these labels and the corresponding degree sequence in layer i. Moreover, only nodes with a positive degree are taken into account to obtain the correlation value in order to reduce the noise. In particular, inactive nodes are ignored as they should be.

To measure the quality of the fit, one can compute the matrix $A = (a_{ij})$, where $a_{ij} \in [-1, 1]$ is the Kendall rank correlation coefficient between layers i and j for the real network G. Clearly, only nodes that are active in both layers (i.e., nodes in $V_i \cap V_j$) are considered. Once a digital twin is generated, matrix B is computed analogously, but this time for the twin $\hat{G}$. As before, the divergence score is simply the (normalised) Frobenius norm between the two matrices:

$$\mathcal{D}_\tau(\hat{G}, G) = \frac{\|A - B\|_F}{\sqrt{4\ell(\ell-1)}} = \sqrt{\frac{1}{4\binom{\ell}{2}} \sum_{1 \le i < j \le \ell} (a_{ij} - b_{ij})^2}.$$

Correlation strength between communities and the reference layer (r_i). This family of parameters models correlations between communities formed by nodes that are active in two different layers. These parameters seem to be the most challenging to retrieve, as we do not have access to the latent biscuit-like reference layer used during network generation. However, once a digital twin is generated, one can easily measure the success by computing the corresponding divergence score $\mathcal{D}_r(\hat{G}, G)$. The relationship between the parameters r_i and the divergence score is not clear, but one can minimise $\mathcal{D}_r(\hat{G}, G)$ by efficiently searching the parameter space.

The proposed method for predicting r bases on the access to the previously estimated remaining part of the `mABCD` configuration and the A matrix of the original network. The objective function creates a given number of candidate twins according to the evaluated r supplemented by the already known rest of the parameters. Next, it computes a B matrix for each of them and returns the mean divergence score $\mathcal{D}_r$ over them. This value is then minimised during the Bayesian optimisation process. By sequentially evaluating the objective function at values of r selected based on the outcomes of previous evaluations, the method seeks to identify the best-matching value of r. The implementation employed, provided by the `skopt` library, accounts for the non-deterministic behaviour of `mABCD` by incorporating Gaussian noise into the surrogate model. A typical optimisation trajectory for fitting r is shown in Fig. 2.

Let us then concentrate on the divergence score. For each layer $i \in [\ell]$ in the real network G, we independently run some stable clustering algorithm (in experiments, the Greedy Modularity Optimisation [5]) to get a partition $\mathcal{P}_i$ that identifies its community structure. Then, one can compute the matrix $A = (a_{ij})$, where $a_{ij} \in [0, 1]$ is the adjusted mutual information (AMI) between partitions $\mathcal{P}_i$ and $\mathcal{P}_j$ induced by nodes that are active in both layers, that is, partitions induced by $V_i \cap V_j$. Matrix B is

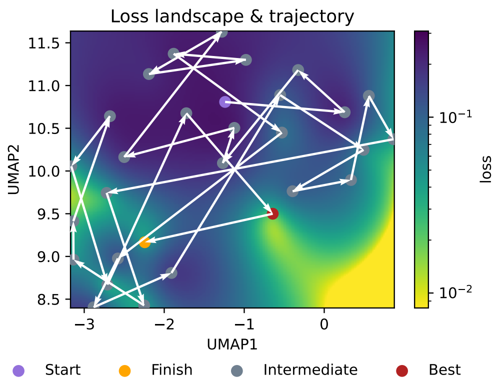

Fig. 2: A typical optimisation trajectory projected on $\mathfrak{R}^2$ space of r obtained during the experimental part. Note the trade-off between exploration and exploitation that the employed method preserves, by which it avoids getting stuck in local minima.

computed analogously but for the twin $\hat{G}$ and the divergence score is defined as follows:

$$\mathcal{D}_r(\hat{G}, G) = \frac{\|A - B\|_F}{\sqrt{\ell(\ell-1)}} = \sqrt{\frac{1}{\binom{\ell}{2}} \sum_{1 \le i < j \le \ell} (a_{ij} - b_{ij})^2}.$$

Degree distributions ($\gamma_i, \delta_i, \Delta_i$). This family of parameters models the degree sequences in the corresponding layers. Among them, only the retrieval of γ_i is non-trivial, as it involves the general challenge of fitting an observed distribution to a specific model. To estimate this parameter, we employ the `powerlaw` library [1], implicitly assuming that the degree distribution follows a power-law.

The `mABCD` model generates, by design, sequences following a power-law with parameters provided as the input. Hence, the only discrepancy between the original network G and its synthetic digital twin $\hat{G}$ comes from fitting the power-law distribution into a given real network degree sequence. To measure the quality of the fit, we might use the Kolmogorov–Smirnov test. This test is a nonparametric statistical test used to determine if a sample comes from a specific distribution (one-sample) or if two samples come from the same distribution (two-sample) by measuring the maximum difference between their cumulative distribution functions (CDFs). In our scenario, it implies the following divergence score of the distribution from the real network from the theoretical distribution:

$$\mathcal{D}_\gamma(\hat{G}, G) = \frac{1}{\ell} \sum_{i \in [\ell]} \max_{k \in [\delta_i, \Delta_i]} \left| \frac{N_i(k)}{N_i(\delta_i)} - \frac{\int_k^\infty x^{-\gamma_i} dx}{\int_{\delta_i}^\infty x^{-\gamma_i} dx} \right|,$$

where $N_i(k)$ is the number of nodes of degree at least k in layer i (as a result, as usual, we ignore inactive nodes in this layer).

For this short paper, for simplicity, we assume that the degree sequence of a real multilayered network G is close to a power law. But not all networks have degree sequences of this nature. To solve this issue, the `mABCD` model is flexible and allows the degree sequences to be directly injected into the model. Since `mABCD` is using the classical configuration model to generate edges in each layer, with this more advanced option, the degree sequence of the digital twin $\hat{G}$ would match *exactly* the original network G. We leave an implementation of this extension into our framework and an investigation of its consequences on the quality of the digital twin for the future (*Open Problem 3*).

Distributions of community sizes (β_i, s_i, S_i). The next family of parameters governs distributions of community sizes in the corresponding layers. Recall that we already extracted the community structure in each layer $i \in [\ell]$ of the real network G by running some stable clustering algorithm, partitions $\mathcal{P}_i$. We may now compute the corresponding sequences of community sizes and extract the values of s_i and S_i from these sequences. Following the same procedure as for dealing with the degree distributions, one can estimate the parameters β_i and measure the success by computing the corresponding divergence score $\mathcal{D}_\beta(\hat{G}, G)$. As before, we leave it for future work to investigate how much one gains if the sequence of community sizes is directly injected into the `mABCD` model (*Open Problem 4*).

Level of noise (ξ_i). Finally, to estimate the layer-wise noise level between communities, we again analyse the community structure $\mathcal{P}_i$ of each network layer $i \in [\ell]$. For each layer i of the original multilayered network G, the parameter ξ_i is extracted by simply looking at the fraction of inter-community edges (edges connecting nodes from different parts of the partition $\mathcal{P}_i$) relative to the total number of edges in the layer i. The `mABCD` model aims to preserve this property, but some very small error could be introduced. More importantly, the model preserves the level of noise between the ground-truth partition, which might not be exactly the same as the partition found by the clustering algorithm (especially for noisy graphs). Hence, once the twin $\hat{G}$ is generated, we extract the associated parameters $\hat{\xi}_i$ and compute the divergence score:

$$\mathcal{D}_\xi(\hat{G}, G) = \sqrt{\frac{1}{\ell} \sum_{i \in [\ell]} (\xi_i - \hat{\xi}_i)^2}.$$

Cumulative Divergence Score. The parameters of `mABCD` affect some important properties of multilayered networks; hence, it is desired for the digital twin $\hat{G}$ to match these properties as best as possible. For each parameter of the model, we introduced above some natural ways to measure how well the corresponding property is preserved via their divergence scores. To estimate an overall quality of the digital twin, one needs to somehow combine these scores. Their definitions guarantee that each of the scores is in $[0, 1]$, but they are not comparable. Hence, before we develop a framework for generating good digital twins, we need to come up with a good measure of quality that combines all aspects together (*Open Problem 5*).

4 Experiments

To test the proposed configuration retrieval method, two experiments using the goodness-of-fit framework proposed in Sec. 3 were performed. In both experiments, a real-world multilayer network, namely *Freebase* [22], was employed. The network consists of $\ell = 3$ layers and includes 3,492 actors. The numbers of nodes in the consecutive layers are

3,479, 2,091, and 1,865, respectively. The corresponding numbers of edges per layer are 129,097, 7,099, and 5,948. The mean degree is 75.41, and the average closeness centrality is 0.1.

4.1 Experiment 1 – Dimensionality of the Reference Layer *d*

The objective of the first experiment was to evaluate the influence of the dimension of the reference layer (d) on the divergence scores between the original network and its produced twins. As such, the estimation procedure was performed independently for four different values of d, i.e. $d \in \{1, 2, 4, 8\}$. Note that it directly affects the Bayesian optimisation component of the proposed method that is used to estimate r.

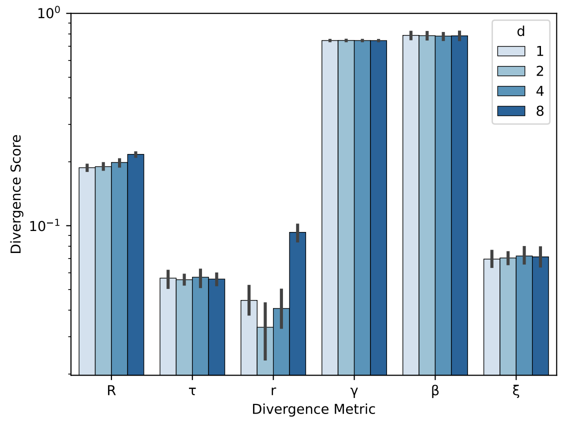

Fig. 3: Divergence scores (log scale) for configuration retrieval with fixed $d = 2^k$ dimension, $k \in \{0, 1, 2, 3\}$; estimation with tuned r and loss $\mathcal{D}_r(\hat{G}, G)$.

Fig. 3 presents a comparison of the average values of the corresponding divergence metrics across the tested d. The scores associated with γ, β, ξ, and τ remain virtually unchanged, as expected, since these parameters were fixed during the optimisation of r. Their stability only confirms the reproducibility of the proposed approach. In contrast, a pronounced effect of d is observed for the AMI-based interlayer correlation (r), resulting in non-monotonic divergence values with respect to increasing d, which is consistent with the hypothesis of parameter interdependence. Nevertheless, the lowest divergence is attained for $d = 2$, which coincides with the design choice recommended by the authors of `mABCD`.

An additional, perhaps less anticipated, effect of a high d value is the increased divergence in edge correlation (R). The increasing misalignment of communities between layers, when comparing the original network to the digital twin, creates unfavourable conditions for efficient link rewiring (phase 6 of the `mABCD` process). It should also be noted that relatively high values of γ and β can stem from the non necessarily power-law-like distribution of degrees and communities in *Freebase*. A potential improvement for that could be injecting sequences of them directly into `mABCD` and, by that, to pass over some of its phases.

4.2 Experiment 2 – Configuration Retrieval Methods

The objective of the second experiment was to evaluate various `mABCD` configuration retrieval methods within the Bayesian framework outlined in Sec. 3. Specifically, we are interested in the following research question. Does increasing the fidelity of the produced twin in one measured dimension trade off for a decrease in other scores (*Open Problem 6*)? To make a step toward this direction, we investigated whether a specific Bayesian tuning configuration exists that outperforms the others across all divergence dimensions.

Based on the considerations, four configuration retrieval runs were executed with the following specifications:

1. decision variable r with loss based on $\mathcal{D}_r(\hat{G}, G)$;
2. decision variables r and τ with loss based on $\mathcal{D}_r(\hat{G}, G)$;
3. decision variables r and τ with loss based on $\mathcal{D}_\tau(\hat{G}, G)$;
4. decision variables r and τ with loss based on $(\mathcal{D}_r(\hat{G}, G) + \mathcal{D}_\tau(\hat{G}, G))/2$.

Based on the outcomes of Experiment 1, we set $d = 2$ for all configuration retrieval specifications. Subsequently, ten digital twins were generated for each set of derived `mABCD` parameters, and the divergence metrics were calculated (Fig. 4).

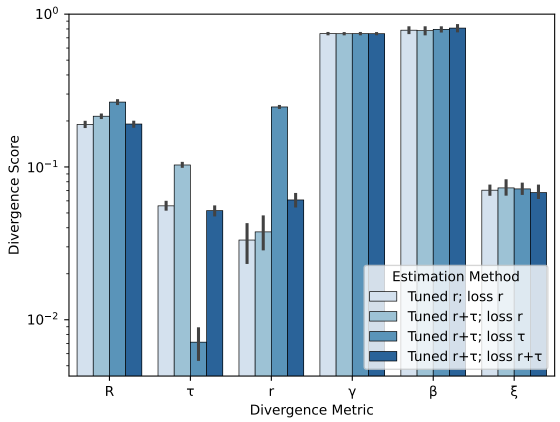

Fig. 4: Divergence scores for four configuration estimation methods; fixed $d = 2$.

The resulting scores for γ, β, and ξ once again validate correct implementation of the configuration retrieval: their minimal variation is attributed to the stochastic nature of `mABCD` and the community extraction algorithm. More significant findings, consistent with expectations, are observed in the divergence metrics $\mathcal{D}_\tau(\hat{G}, G)$ and $\mathcal{D}_r(\hat{G}, G)$. Both scores are minimised when the optimisation technique targets the corresponding loss function (either $\mathcal{D}_\tau(\hat{G}, G)$ or $\mathcal{D}_r(\hat{G}, G)$). However, a trade-off is evident in the form of increased divergence for the alternative metric. This is particularly pronounced for the τ-based loss method, which achieves a divergence score of 0.0071, compared to an average of 0.0701 across other specifications. Conversely, for the $\mathcal{D}_r(\hat{G}, G)$ criterion, the τ-based loss yields 0.2469, indicating suboptimal performance compared to the average of 0.0438 for other losses. Interestingly, expanding the set of decision variables from a single r to include both r and τ (while maintaining $\mathcal{D}_r(\hat{G}, G)$ loss) does not decrease the $\mathcal{D}_r(\hat{G}, G)$ score, but notably exacerbates the $\mathcal{D}_\tau(\hat{G}, G)$. This phenomenon may be attributed to the curse of dimensionality and insufficient exploration of the parameter space. Variations in the R score are difficult to attribute definitively; however, given their small magnitude, it can be concluded that the specific choice of decision variables and loss functions has a negligible impact on edge correlation fidelity.

To complement the analysis of Experiment 2, we present the loss trajectories for the optimisation model utilising both r and τ as decision variables. Fig. 5 illustrates the instantaneous loss calculated at each iteration of the search. It should be noted that we report $\mathcal{D}_\tau(\hat{G}, G)$, $\mathcal{D}_r(\hat{G}, G)$, and $(\mathcal{D}_r(\hat{G}, G) + \mathcal{D}_\tau(\hat{G}, G))/2$ for each loss used as the objective function in the optimisation problem. Consistent reductions in the loss associated with the primary objective are observable, highlighting the efficacy of the Bayesian optimisation. Conversely, the trajectories for the non-target criterion predominantly oscillate around the average, indicating that any reduction in the alternative loss occurs incidentally rather than through directed optimisation.

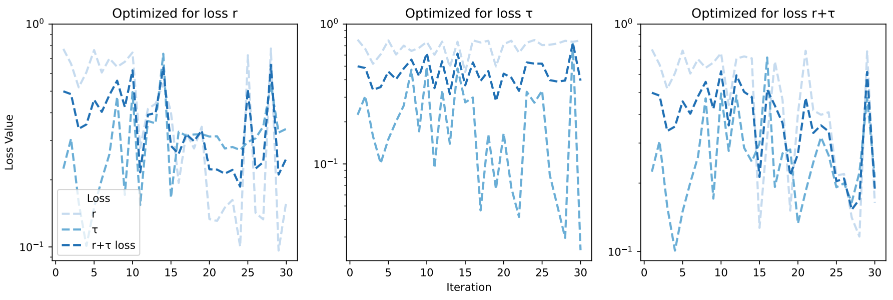

Fig. 5: $\mathcal{D}_\tau(\hat{G}, G)$, $\mathcal{D}_r(\hat{G}, G)$, and $(\mathcal{D}_r(\hat{G}, G) + \mathcal{D}_\tau(\hat{G}, G))/2$ scores as functions of the optimisation step in finding the best matching r or τ.

5 Conclusions

In this work, we discussed a problem of configuration retrieval for multilayer networks with the synthetic network generator — mABCD. Addressing that issue is vital for twinning complex networked systems and opens new possibilities for graph-based data augmentation in machine learning applications.

The main contribution of this study is the proposal of a method for creating statistically faithful network twins from empirical multilayer datasets. We also introduce several divergence measures for assessing the quality of parameter estimation. The method is evaluated on a real-world network representing cooperation within the movie industry. The results provide preliminary evidence supporting the validity of the proposed approach; however, a key finding is that independent parameter estimation techniques encounter intrinsic accuracy limits, as the configuration parameters collectively determine the structural properties of networks generated by mABCD. This observation naturally motivates the development of joint prediction methods as a promising direction for future work.

Nevertheless, as this work is rather a perspective paper, we left readers with six *O*pen *P*roblems that can serve as a roadmap for deeper investigation and which (we believe) will open a fruitful discussion.

- *OP-1* assessing whether utilising deep embedding methods for joint-parameter estimation is feasible;
- *OP-2* understanding the influence of the hyper-parameter d and selecting its optimal value;
- *OP-3* handling networks with degree sequences that do not follow the power-law by injecting them into `mABCD`;
- *OP-4* handling networks with community size sequences that do not follow the power-law by injecting them into `mABCD`;
- *OP-5* providing a cumulative divergence score, a scalar value which encapsulates all structural discrepancies between the original network and the digital twin;
- *OP-6* assess if increasing the fidelity of the produced twin with respect to one parameter trades off with a decrease in other scores.

Acknowledgments. This research was partially supported by: (1) EU under the Horizon Europe, grant no. 101086321 OMINO; (2) Polish Ministry of Science and Higher Education, International Projects Co-Funded programme; (3) National Science Centre, Poland, grant no. 2022/45/B/ST6/04145; (4) Polish National Agency for Academic Exchange, Strategic Partnerships programme, grant no. BPI/PST/2024/1/00129/U/00001; (5) Wrocław University of Science and Technology, Academia Profesorum Iuniorum programme. Views and opinions expressed are, however, those of the authors only and do not necessarily reflect those of the founding agencies.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

Code and Data The methods presented in this manuscript have been implemented in Python. The source code and installation instructions are available at: `https://github.com/network-science-lab/mabcd-for-digital-twins`.

References

1. Alstott, J., Bullmore, E., Plenz, D.: powerlaw: A python package for analysis of heavy-tailed distributions. PLOS ONE **9**(1), 1–11 (01 2014)
2. Barrett, J., DeWolfe, R., Kamiński, B., Prałat, P., Smith, A., Théberge, F.: The artificial benchmark for community detection with outliers and overlapping communities (abcd+o2). In: International Workshop on Modelling and Mining Networks. pp. 125–140. Springer (2025)
3. Barrett, J., Kamiński, B., Prałat, P., Théberge, F.: Self-similarity of communities of the ABCD model. Theoretical Computer Science **1026**, 115012 (2025)
4. Boguna, M., Bonamassa, I., De Domenico, M., Havlin, S., Krioukov, D., Serrano, M.Á.: Network geometry. Nature Reviews Physics **3**(2), 114–135 (2021)
5. Clauset, A., Newman, M.E.J., Moore, C.: Finding community structure in very large networks. Physical Review E **70**(6) (Dec 2004)

6. De Domenico, M., Granell, C., Porter, M.A., Arenas, A.: The physics of spreading processes in multilayer networks. Nature Physics **12**(10), 901–911 (2016)
7. Dehghan, A., Prałat, P., Théberge, F.: Network embedding exploration tool (neext). arXiv preprint arXiv:2503.15853 (2025)
8. Easley, D., Kleinberg, J.: Networks, Crowds, and Markets: Reasoning About a Highly Connected World. Cambridge University Press, Cambridge (2010)
9. Henry, A.D., Prałat, P., Zhang, C.Q.: Emergence of segregation in evolving social networks. PNAS **108**(21), 8605–8610 (2011)
10. Kamiński, B., Olczak, T., Pankratz, B., Prałat, P., Théberge, F.: Properties and performance of the ABCDe random graph model with community structure. Big Data Research **30**, 100348 (2022)
11. Kamiński, B., Pankratz, B., Prałat, P., Théberge, F.: Modularity of the ABCD random graph model with community structure. Journal of Complex Networks **10**(6), cnac050 (2022)
12. Kamiński, B., Prałat, P., Théberge, F.: Artificial benchmark for community detection (ABCD) - Fast random graph model with community structure. Network Science pp. 1–26 (2021)
13. Kamiński, B., Prałat, P., Théberge, F.: Artificial benchmark for community detection with outliers (ABCD+o). Applied Network Science **8**(1), 25 (2023)
14. Kamiński, B., Prałat, P., Théberge, F.: Hypergraph artificial benchmark for community detection (h–ABCD). Journal of Complex Networks **11**(4), cnad028 (2023)
15. Kivelä, M., Arenas, A., Barthelemy, M., Gleeson, J.P., Moreno, Y., Porter, M.A.: Multilayer networks. Journal of Complex Networks **2**(3), 203–271 (07 2014)
16. Kraiński, Ł., Czuba, M., Bródka, P., Prałat, P., Kamiński, B., Théberge, F.: Multilayer artificial benchmark for community detection (mabcd). Expert Systems with Applications **307**, 130920 (2026)
17. Lancichinetti, A., Fortunato, S.: Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. Physical Review E **80**(1), 016118 (2009)
18. Lancichinetti, A., Fortunato, S., Radicchi, F.: Benchmark graphs for testing community detection algorithms. Physical review E **78**(4), 046110 (2008)
19. Serrano, M.Á., Boguñá, M.: The Shortest Path to Network Geometry: A Practical Guide to Basic Models and Applications. Cambridge University Press (2021)
20. Serrano, M.Á., Krioukov, D., Boguná, M.: Self-similarity of complex networks and hidden metric spaces. Physical review letters **100**(7), 078701 (2008)
21. Tambuscio, M., Ruffo, G., Flammini, A., Menczer, F.: Fact-checking effect on viral hoaxes: A model of misinformation spread in social networks. In: Proceedings of the 24th International Conference on World Wide Web. pp. 977–982 (2015)
22. Wang, X., Liu, N., Han, H., Shi, C.: Self-supervised heterogeneous graph neural network with co-contrastive learning. In: Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining. pp. 1726–1736 (2021)
23. Wen, J., Gabrys, B., Musial, K.: DTCNS: A python toolbox for digital twin-oriented complex networked systems. SoftwareX **27**, 101818 (2024)