

A latent linear model for nonlinear coupled oscillators on graphs

Agam Goyal,^{1,2, a)} Zhaoxing Wu,^{3, a)} Richard P. Yim,⁴ Binhao Chen,⁵ Zihong Xu,² and Hanbaek Lyu^{2, b)}

¹⁾Department of Computer Science, University of Wisconsin - Madison, WI 53706

²⁾Department of Mathematics, University of Wisconsin - Madison, WI 53706

³⁾Department of Statistics, University of Washington, Seattle, WA 98195

⁴⁾Bakar Computational Health Sciences Institute, University of California, San Francisco, CA 94143

⁵⁾Department of Computer Science, Brown University, Providence, RI 02912

A system of coupled oscillators on an arbitrary graph is locally driven by the tendency to mutual synchronization between nearby oscillators, but can and often exhibit nonlinear behavior on the whole graph. Understanding such nonlinear behavior has been a key challenge in predicting whether all oscillators in such a system will eventually synchronize. In this paper, we demonstrate that, surprisingly, such nonlinear behavior of coupled oscillators can be effectively linearized in certain latent dynamic spaces. The key insight is that there is a small number of ‘latent dynamics filters’, each with a specific association with synchronizing and non-synchronizing dynamics on subgraphs so that any observed dynamics on subgraphs can be approximated by a suitable linear combination of such elementary dynamic patterns. Taking an ensemble of subgraph-level predictions provides an interpretable predictor for whether the system on the whole graph reaches global synchronization. We propose algorithms based on supervised matrix factorization to learn such latent dynamics filters. We demonstrate that our method performs competitively in synchronization prediction tasks against baselines and black-box classification algorithms, despite its simple and interpretable architecture.

I. INTRODUCTION

If a group of people is given local clocks with arbitrarily set times, and there is no global reference (for example, GPS), is it possible for the group to synchronize all clocks by only communicating with nearby members? In order for a distributed system to be able to perform high-level tasks that may go beyond the capability of an individual agent, the system must first solve a “clock synchronization” problem to establish a shared notion of time. The study of synchronization of coupled oscillators has been an important subject of research in mathematics and various areas of science for decades^{1,2}, with fruitful applications in many areas, including wireless sensor networks, wildfire monitoring, electric power networks, robotic vehicle networks, and large-scale information fusion^{3–5}.

A system of coupled oscillators is said to be (globally) *synchronized* if all oscillators are at a consensus in terms of their phase or oscillation frequency. In this work, we consider oscillators of identical frequencies and only phase synchronization. Such a global state may or may not emerge depending on how the oscillators interact along the edges of the graph, how such local interaction leads to larger-scale interactions, and so on. In spite of several sufficient conditions on model parameters (e.g., large coupling strength⁶) or initial configuration (e.g., phase concentration⁷ within an open semicircle), it is usually analytically intractable to predict whether a given system of coupled oscillators with arbitrary underlying graph structures will eventually synchronize, more so when the underlying graph is heterogeneous and the initial phase

configuration is not confined in a small arc of the phase space. Furthermore, the interplay between the nonlinear dynamics and network topology can often give rise to highly nonlinear phenomena^{8,9}, which makes it intriguingly hard to study and understand their properties.

With the revolutionary success of machine learning methods in various tasks such as image classification and natural language processing, there has been a surge of interest in employing these methods to study scientific problems that have been previously believed to be extremely difficult^{10–12}. This is also the case for the problem of synchronization prediction, where a number of tools in machine learning have been applied to study the properties of coupled oscillator systems^{13–15}. The recent work of Bassi et al.¹⁶ in particular demonstrated that after a proper reformulation, the synchronization prediction problem on randomly generated graphs can be effectively solved by training binary classification algorithms on a large dataset of synchronizing and non-synchronizing examples, in the sense that the resulting prediction accuracy significantly outperforms a baseline predictor that uses the concentration principle in coupled oscillator theory^{7,17,18}.

However, a key question that remains unanswered is whether we can actually gain any scientific insight into coupled oscillator systems from the advantages of analyzing a massive amount of data using machine learning methods. For instance, we would like to understand ‘how’ a well-performing model is able to make these predictions on coupled oscillator systems and what features in graphs or dynamics it considers most important for this task. Toward this goal, in this paper, we propose an *interpretable* model for synchronization prediction that we call the *Latent Linear Dynamics Model* (LLDM). The logic behind the model is very simple: *Given an observed dynamics on a subgraph, first compute ‘proximity scores’ for how much a set of prescribed patterns*

^{a)}Co-first author

^{b)}Corresponding author; hlyu@math.wisc.edu

we observe there, and then use the proximity scores for those patterns as an input to a logistic classifier.

The key challenge in our approach is to figure out what fundamental patterns of dynamics on subgraphs we seek to observe for the purpose of synchronization prediction. For example, if the underlying graph is very dense, then it will be likely that the dynamics will eventually synchronize. Also, if the observed phase configuration is confined in an open semicircle, then we know the system will eventually synchronize. If the graph is sparse and contains a long cycle, then it would be hard to see eventual synchronization. While such ‘patterned behaviors’ are informed by the existing knowledge on coupled oscillators, our novel approach here is to learn such ‘critical patterns for synchronization’ directly from the data.

We summarize our key contributions through this work:

1. We propose a novel and interpretable framework for the prediction of synchronization in coupled oscillators, leveraging the feature learning capabilities of matrix factorization techniques to learn latent linear representations of underlying network dynamics.
2. We propose various ways of approaching the synchronization prediction problem using LLDM, by the use of data-informed and computationally efficient, theory-informed approaches, in addition to using both supervised and unsupervised matrix factorization techniques.
3. We propose a compute-efficient and novel method for the prediction of dynamics synchronization on large-scale graphs by the use of LLDM on a set of subgraphs sampled by motif sampling techniques, followed by recursive averaging of predicted probabilities.

To the best of our knowledge, this is the first work to study the synchronization of small and large-scale coupled oscillator dynamical systems on graphs through the lens of representation learning techniques that also focus on interpretability—a critical aspect of modeling dynamical systems.

A. Related Works

1. Machine Learning for Synchronization Prediction

There has been a surge in studies that have approached the study the dynamic oscillator systems by using machine learning techniques. Some studies such as Thiem et al.¹⁹ used Feed Forward Neural Networks²⁰ (FFNNs) to analyze Kuramoto dynamics in specific, while Hefny et al.²¹ use LASSO regression for modeling independent subsystems of dynamical systems. Itabashi et al.²² use features derived from early-stage topological dynamics to classify Kuramoto oscillator dynamics. Furthermore, Bassi et al.¹⁶ show that various classical machine learning algorithms can be used for the synchronization prediction problem by training them on a large dataset of

synchronizing and non-synchronizing coupled oscillator systems on randomly generated graphs. Their method was applied to the Kuramoto oscillators as well as discrete oscillator models such as the Firefly Cellular Automata (FCA²³) and Greenberg-Hastings Model (GHM²⁴). Recently, Chen et al.²⁵ proposed to use reinforcement learning to find an optimal pulse-interaction mechanism that optimizes the probability of synchronization of pulse-coupled oscillators, while Mahlow et al.²⁶ proposed to utilize k -nearest neighbor regressor to predict the emergence of environment-induced spontaneous quantum synchronization in an open system setting.

Despite the increasing interest in utilizing these methods to study dynamical systems, there remains a gap in understanding what features are crucial for these models to make predictions. We aim to bridge this gap in our work, where we leverage feature-representation learning techniques like non-negative matrix factorization²⁷ and supervised matrix factorization^{28,29} (See Appendices C and D for details) to provide an interpretable framework for synchronization prediction.

2. Matrix Factorization Techniques and interpretable feature extraction

Matrix factorization techniques have proven to be powerful tools for describing various latent features of data of interest in terms of a ‘linear combination’ of atomic elements. This problem has been studied for many decades and has been used in various scientific fields^{30–36}. More generally, low-dimensional feature extraction techniques have been used extensively in the last few decades for complex tasks that involve studying the local interactions of elements in various fields. Highly accurate and precise reconstruction of billions of protein structures¹⁰; OTT media recommendation systems decoding users’ item-response patterns¹¹; the innovation in novel TPU architectures to allow efficient matrix factorization¹²; and dating back to the PageRank³⁷ search algorithm for ranking internet websites and web pages; these are all some of the most well-known and important applications of such techniques.

Despite enjoying fruitful applications in the aforementioned areas, the potential of these techniques has not been harnessed widely in the context of coupled oscillators. Recently, Luo³⁸ proposed to decompose network dynamics into a composite of weighted principal components, and subsequently learn the governing differential equations using sparse regression. In this work, we make use of matrix factorization-based approaches in conjunction with a Markov-chain Monte-Carlo subgraph sampling algorithm³⁹ to learn underlying features from the data and make predictions regarding the synchronization of oscillator dynamic systems.

II. STATEMENT OF THE PROBLEM

A graph $G = (V, E)$ consists of sets V and E , node set and edge set, respectively. Let Ω denote the *phase space* of

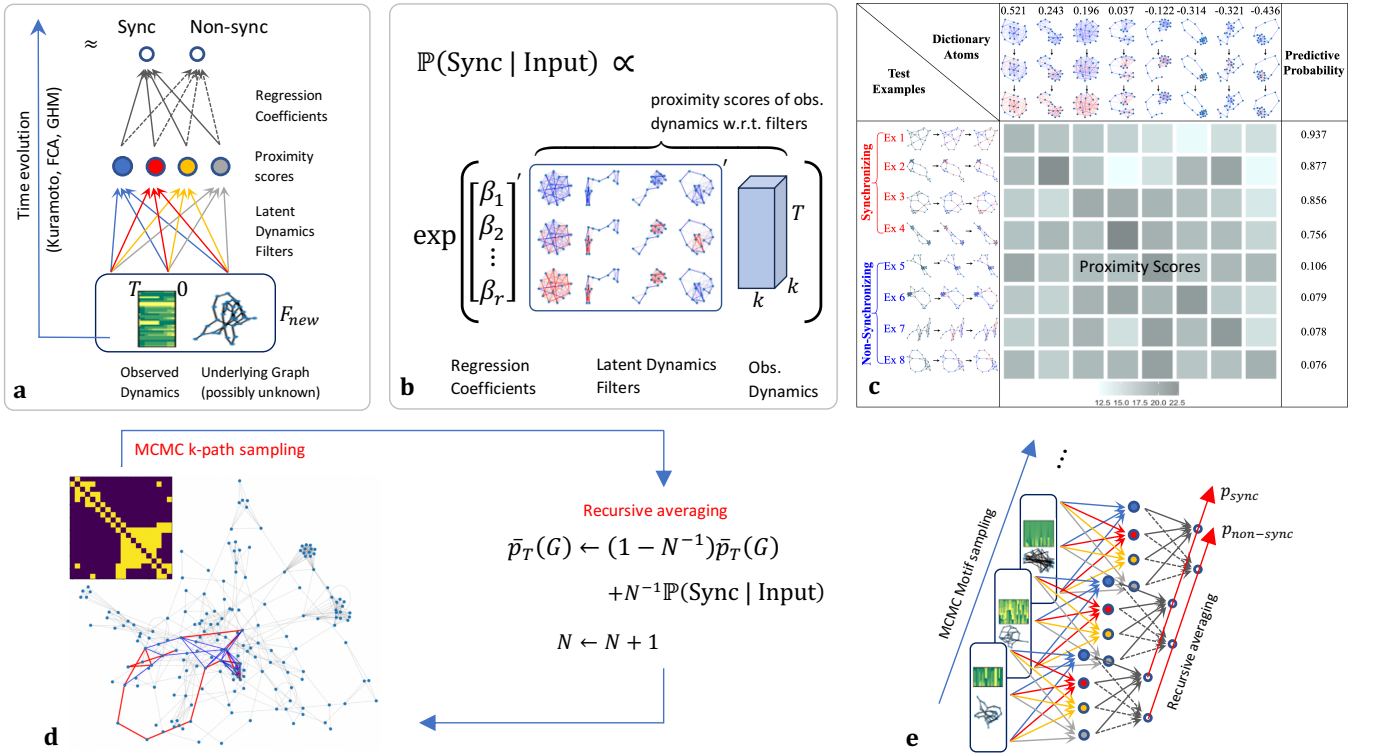


FIG. 1. (Panels **a-c**) Scheme of the Latent Dynamics Model (LDM) for synchronization prediction. Input dynamics on a k -node graph observed for T iterations is represented as a $k \times k \times T$ tensor. Taking convolution of the input dynamics with R (4 in the figure), latent dynamics filters (nonnegative tensors of shape $k \times k \times T$) give proximity scores for R patterns, which are combined into one scalar for a final score for the predictive probability of eventual synchronization. The predictive probability of synchronization is proportional to the exponential of the final score. (Panels **d-e**) A uniformly randomly sampled 20-path (with red edges) and additional edges on the induced subgraph (in blue) with the corresponding adjacency matrix on the top left. We recursively compute the predictive probabilities of synchronization using dynamics on such sampled subgraphs, and the average value converges to the predictive probability of synchronization of the whole graph, which is the expectation of the predictive probabilities over subgraphs induced on k -paths.

each node, which may be taken to be the unit circle $\mathbb{R}/2\pi\mathbb{Z}$ for continuous-state oscillators and the discrete circle $\mathbb{Z}/\kappa\mathbb{Z}$, $\kappa \in \mathbb{N}$ for finite-state oscillators. We call a map $X : V \rightarrow \Omega$ a *phase configuration*, and say it is *synchronized* if it takes a constant value across nodes (i.e., $X(v) = \text{Const.}$ for all $v \in V$). A *coupling* is a function $\mathcal{C}$ that maps each pair (G, X_0) of graph and initial configuration $X_0 : V \rightarrow \Omega$ deterministically to a *trajectory* $(X_t)_{t \geq 0}$ of phase configurations $X_t : V \rightarrow \Omega$. In this paper, we consider $\mathcal{C}$ to be the time evolution rule for Kuramoto model⁴⁰, Firefly Cellular Automata (FCA)^{23,41}, and Greenberg-Hastings Model²⁴. We use a discretization of Kuramoto model (see (A2) in the appendix) and an ‘iteration’ of Kuramoto dynamics refers to applying one step of the difference equation. See Appendix A for details on each of these coupled oscillator models.

The main problem we investigate in this work is to predict the synchronization of coupled oscillators on a large-scale graph G using subgraph-level information. This means that we observe some subgraphs of fixed size k (potentially much smaller than the number of nodes in G) and the dynamics in G are restricted on these subgraphs. In order to make this setting precise, we formulate the following sampling oracle:

Subgraph Sampling Oracle. Given a graph G and a fixed

integer $k \geq 0$, we can sample a k -node connected subgraph H of G and observe dynamics on G restricted on H up to a fixed number of iterations T_0 . However, we cannot observe dynamics on larger subgraphs and more than T_0 iterations.

We can now formulate the main problem we aim to address in this work: *Large-scale synchronization prediction by local dynamics decomposition*:

Problem II.1 Let $(X_t)_{t \geq 0}$ be a coupled oscillator dynamics on a (possibly large) connected graph G governed by a coupling $\mathcal{C}$. Suppose that we have a sampling oracle for subgraph size k and time horizon T_0 .

- (i) (Dynamics decomposition) Observed dynamics on subgraphs can be approximately decomposed into a linear combination of some key dynamics patterns.
- (ii) (Synchronization prediction) Using the decomposition in (i), one can predict the following indicator variable:

$$\mathbf{1}(X_t \text{ on } G \text{ is eventually synchronized}). \quad (1)$$

Our goal is to use SMF to learn low-rank latent factors that offer interpretable, data-reconstructive, and class-

discriminative features, addressing the challenges posed by high-dimensional data.

Our approach has two components. First, we will develop an interpretable model to predict the synchronization indicator at the k -node subgraph level. Second, we apply the trained subgraph-level model to a randomly sampled k -node along with the observed dynamics. The expectation of the predicted probability over random k -node subgraphs will be the predictive probability for the parent graph to eventually synchronize.

III. MODEL DESCRIPTION

At a high level, our model predicts the eventual synchronization on the whole graph G by averaging the predictions on many suitably chosen subgraphs of F . Below, we first describe how we make synchronization prediction using a single subgraph and then discuss how to efficiently combine subgraph-level prediction with our particular choice of the subgraph sampling oracle.

A. Latent linear dynamics model on a single subgraph

Suppose that we have a system of coupled oscillators on a connected graph $G = (V, E)$. The goal of LLDM is to model the predictive probability that the system will eventually synchronize based on the observation of dynamics up to T iterations restricted on a k -node subgraph of G , say F . (Here we assume $k \leq |V|$ and allow $F = G$.) Since observing dynamics on a subgraph during a fixed time period only gives partial information on the long-term dynamics on the whole graph G , we model the indicator variable that the dynamics on G will eventually synchronize given this partial information as a Bernoulli random variable with unknown success probability.

For a precise formulation, let $(X_t)_{0 \leq t < T}$, $X_t : V \rightarrow \mathbb{Z}/\kappa\mathbb{Z}$ denote the dynamics on G that is assumed to have evolved according to some coupled oscillator model (e.g., FCA, Kuramoto, or GHM). Let $X_t[F]$ denote the restriction of X_t on the node set of F . Our basic modeling assumption is the following:

$$\mathbf{1}(X_t \text{ synchronizes as } t \rightarrow \infty | F, (X_t[F])_{0 \leq t < T}) \sim \text{Bernoulli}(p_T(F)), \quad (2)$$

where $p_T(F)$ is the unknown probability of eventual synchronization of the dynamics on G given the partial observation.

Next, we introduce the key modeling assumption for LLDM that $p_T(F)$ above depends on a certain R -dimensional *proximity score vector* $\mathbf{h}$ that is computed as follows. First, we represent the pair $(F, (X_t[F])_{0 \leq t < T})$ of input data by a nonnegative tensor $\mathcal{X}$ of shape $k \times k \times T$, where each slice $\mathcal{X}[:, :, t]$ represents the graph topology F decorated by the phase configuration X_t defined as

$$\mathcal{X}[i, j, t] := A_{ij} \min\{(X_t(i) - X_t(j) \pmod{\kappa}), X_t(j) - X_t(i) \pmod{\kappa})\}, \quad (3)$$

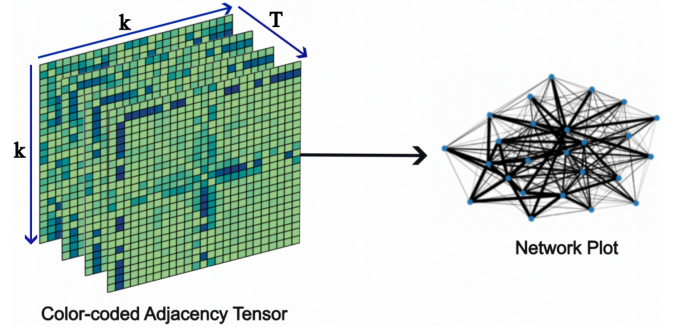


FIG. 2. Example of a $k \times k \times T$ colored adjacency tensor (CAT) and the top slice as a graph network plot

Here, $A = (A_{ij})$ is the adjacency matrix of F defined by $A_{ij} := \mathbf{1}(\{i, j\} \in E(F))$. We call the tensor $\mathcal{X}$ the *colored adjacency tensor* (CAT) for $(F, (X_t[F])_{1 \leq t < T})$ (see Figure 2). In Appendix E, we demonstrate that the use of various encoding methods of graph topology of F , as simple as the adjacency matrix or as advanced as modern graph embedding methods such as DeepWalk⁴², graph2vec⁴³, provides little to no improvement in prediction accuracy for the synchronization problem at hand.

Second, a key notion we introduce in this work is the ‘dictionary’ of *latent dynamics filters*, which is an R -tuple $\mathcal{D} = [\mathcal{F}_1, \dots, \mathcal{F}_R]$ of nonnegative tensors $\mathcal{F}_i \in \mathbb{R}_{\geq 0}^{k \times k \times T}$ of unit Frobenius norm. Here R is an integer parameter called the ‘size’ of the dictionary $\mathcal{D}$. Each filter $\mathcal{F}_i$ represents an elementary pattern of coupled oscillator dynamics on a general k -node graph for T iterations. We denote by $\langle \cdot, \cdot \rangle$ the inner product between two tensors of the same shape, which is the sum of the products of all corresponding entries. Since both the input tensor $\mathcal{X}$ and the filter $\mathcal{F}_i$ are assumed to be nonnegative, their inner product $\langle \mathcal{F}_i, \mathcal{X} \rangle$ can be interpreted as the *proximity score* of the input dynamics $\mathcal{X}$ with respect to the dynamics pattern represented by the filter $\mathcal{F}_i$. Since there are R filters, we can compress a single tensor data element, $\mathcal{X}$, into an R -dimensional vector of proximity scores

$$\mathbf{h} := \text{MAT}(\mathcal{D})^T \text{VEC}(\mathcal{X}) = (\langle \mathcal{F}_1, \mathcal{X} \rangle, \dots, \langle \mathcal{F}_R, \mathcal{X} \rangle)^T, \quad (4)$$

where $\text{MAT}(\cdot)$ and $\text{VEC}(\cdot)$ are the matricization and the vectorization operators—fixing the lexicographic ordering of entries. (Note that $\text{MAT}(\mathcal{D}) \in \mathbb{R}^{k^2 T \times r}$ and $\text{VEC}(\mathcal{X}) \in \mathbb{R}^{k^2 T}$ so that $\text{MAT}(\mathcal{D})^T \text{VEC}(\mathcal{X}) \in \mathbb{R}^r$.) The i th coordinate of $\mathbf{h}$ measures how similar the observed dynamics on the subgraph F , encoded in the tensor $\mathcal{X}$, to the particular dynamics and networks encoded in the i th latent dynamic filter $\mathcal{F}_i$.

Now we suppose the probability $p_T(F)$ of eventual synchronization given data $\mathcal{X} \in \mathbb{R}^{k \times k \times T}$ is modeled as follows:

$$p_T(F) = \frac{\exp(\beta^T \mathbf{h})}{1 + \exp(\beta^T \mathbf{h})} \text{ or } \text{logit}(p_T(F)) = \beta^T \mathbf{h}, \quad (5)$$

where $\beta = (\beta_1, \dots, \beta_R)^T \in \mathbb{R}^r$ is a vector of regression coefficients and $\text{logit}(p) := \frac{p}{1-p}$. Thus, LLDM for fixed subgraph

size k is parameterized by $(\mathcal{D}, \beta)$. From the above representation, we see that LLDM is a linear model on the latent space of features measured by the proximity score matrix.

Our general scheme of predicting synchronization of coupled oscillators using LLDM is depicted in Figure 1. Figure 1c depicts eight observed FCA dynamics on subgraphs of NWS (see Table I) in the rows (“Test Examples”) and eight latent dynamics filters with their corresponding regression coefficients in the columns (“Dictionary Atoms”). Each of them is $k \times k \times T$ tensors for $k = 20$ and $T = 50$, and three snapshots at times 0, 25, 50 are shown with arrows indicating time evolution. The proximity scores are shown in the heat map (in grayscale) and the corresponding predictive probabilities (see (5)) are shown in the last column. “Ex1” in Figure 1c is a synchronizing test example, which has the largest proximity score with the first latent dynamic filter (with regression coefficient 0.521), and has a large predictive probability of 0.917 for eventual synchronization. Also, “Ex7” is a non-synchronizing test example that has large proximity scores with the fifth (−0.122) and the seventh (−0.321) latent dynamic filter and has a small predictive probability of 0.078 for eventual non-synchronization.

B. Choosing the sampling oracle: k -path motif sampling

In the previous section, we introduced LLDM with parameters $\mathcal{D}$ and β , without any assumption on the subgraph F on which we observe the dynamics on G . In order for this model to be effective, especially when G is large and sparse, we may need to restrict the class of ‘appropriate’ k -node subgraphs in G sampled by our sampling oracle. For instance, if we consider all induced subgraphs obtained by sampling k nodes uniformly at random from G , then when G is sparse, most of such subgraphs will be disconnected and have a few edges, so the dynamics observed on such subgraphs will not be informative of the dynamics on the whole graph G . Furthermore, LLDM assumes that the observed subgraphs come with prescribed node ordering so that their adjacency matrix, and in turn their CAT. Thus it is computationally beneficial to restrict ourselves to consider k -node connected subgraphs that have canonical node ordering.

We propose to consider k -node connected subgraphs that are obtained by first uniformly randomly sampling a ‘ k -path’ in the graph G and then taking the induced subgraph on the sampled paths (i.e., including all edges between the sampled nodes in G). See Figure 1d for an illustration. Here a k -walk is a sequence $\mathbf{x} = (x_1, \dots, x_k)$ of k nodes (which may or may not be distinct), such that x_j and x_{j+1} are adjacent for all $j \in \{1, \dots, k-1\}$. A k -walk is a k -path if all nodes in the walk are distinct. This sampling method has two notable advantages. First, it guarantees that the sampled k -node induced subgraph is connected with the minimum number of imposed edges. Second, it induces a canonical node ordering of the sampled subgraphs. In order for efficient sampling of a large number of k -paths approximately uniformly at random, we use the k -walk motif-sampling algorithm in Lyu et al.³⁹ (which is a Markov chain Monte Carlo (MCMC) algorithm) in

conjunction with rejection sampling. This subgraph sampling oracle has been recently used in Lyu et al.⁴⁴ for mesoscale network reconstruction.

C. Averaging subgraph-level predictions over many subgraphs

Using the k -path motif sampling method introduced in the previous section, we propose the following simple procedure to improve LLDM by averaging over many subgraphs. Namely, instead of using a single k -node subgraph F and the dynamics observed on it to predict the synchronization indicator on G , we use many such subgraphs F and average the corresponding predictive probabilities. This effectively combines subgraph-level predictions over many subgraphs. Accordingly, we define

$$\bar{p}_T(G) := \mathbb{E}_F [p_T(F)], \quad (6)$$

where F is a random k -node connected subgraph in G induced on a uniformly random k -path in G , and $p_T(H)$ is the predictive probability (using LLDM, see (5)) of G being eventually synchronized given the information on F . The quantity $\bar{p}_T(G)$ above is the averaged predictive probability that G will eventually synchronize.

In order to effectively compute the expectation in (6), we use the Monte Carlo approximation along an MCMC trajectory of k -paths. That is, our sampling oracle generates a sequence of k -paths $(\mathbf{x}_s)_{s \geq 0}$ in G that forms an irreducible Markov chain. By Lyu et al.³⁹ (Thm. 2.3), we have that almost surely,

$$\bar{p}_T(G) = \lim_{N \rightarrow \infty} \left(p_{T;N}(G) := \frac{1}{N} \sum_{s=1}^N p_T(F[\mathbf{x}_s]) \right) \quad (7)$$

where $F[\mathbf{x}_s]$ denotes the subgraph of G induced on the nodes in the s th k -path $\mathbf{x}_s$. The above sample average in the right-hand side of (7) can be computed recursively without storing all past samples. This gives us the following recursive algorithm for computing the approximate predictive probability $p_{T;s}(G)$ for all $s \geq 1$:

$$\begin{cases} \mathcal{X}_s & \leftarrow \text{CAT on subgraph } F[\mathbf{x}_s] \\ p^{(s)} & \leftarrow \sigma(\beta^T \text{MAT}(\mathcal{D})^T \text{VEC}(\mathcal{X}_s)) \\ p_{T;s}(G) & \leftarrow \left(1 - \frac{1}{s}\right) p_{T;s-1}(G) + \frac{1}{s} p^{(s)}, \end{cases} \quad (8)$$

where $\sigma(x) = \frac{\exp(x)}{1 + \exp(x)}$ and $(\mathcal{D}, \beta)$ is a given hyperparameter for LLDM. The recursion (8) can be executed over arbitrarily many MCMC samples efficiently.

Furthermore, by Lyu et al.³⁹ (Thm. 2.23), the recursive averaging (8) is guaranteed to converge to the population mean (6) exponentially fast in N . That is, for each $\delta > 0$ and for all $N \geq 1$,

$$\mathbb{P}(|\bar{p}_T(G) - p_{T;N}(G)| \geq \delta) < 2 \exp\left(\frac{-2\delta^2 N}{9\tau_{\text{mix}}}\right), \quad (9)$$

where τ_{mix} is the mixing time of the standard lazy simple symmetric random walk on G , which depends on the size and topology of G . In practice, we observe that $p_{T:N}(G)$ converges quickly to $\bar{p}_T(G)$ in many problem instances, see Figure 4a.

IV. LEARNING HYPERPARAMETERS

In this section, we discuss how to learn the model hyperparameters, the regression coefficient vector β , and the dictionary of latent dynamics filters $\mathcal{D}$.

A. Generating the training data set

Suppose we have $2m$ observed coupled oscillator dynamics (Kuramoto, FCA, or GHM) $(X_t^{(j)})_{0 \leq t < T}$ for $j = 1, \dots, 2m$ on the whole graph G , and let $y_j \in \{0, 1\}$ denote the indicator that $X_t^{(j)}$ synchronizes as $T \rightarrow \infty$. We assume half the dynamics are synchronizing ($y_1, \dots, y_m = 1$) and the other half are non-synchronizing ($y_{m+1}, \dots, y_{2m} = 0$). Sample k -node subgraphs $F_1, \dots, F_{N_0}$ in G (we use $k \in \{10, 15, 20, 25, 30\}$) are sampled through our subgraph sampling oracle described in Section III B. We then restrict the first dynamics $(X_t^{(1)})_{0 \leq t < T}$ on the subgraphs $F_1, \dots, F_{N_0}$. This gives N_0 training examples $(\mathcal{X}_1, y_1), \dots, (\mathcal{X}_{N_0}, y_1)$, where each $\mathcal{X}_k$ is the CATs of shape $k \times k \times T$ encoding $(X_t^{(1)})_{0 \leq t < T}$ restricted on F_1 . Next, we restrict the second dynamics $(X_t^{(2)})_{0 \leq t < T}$ on the subgraphs and obtain training examples $(\mathcal{X}_{N_0+1}, y_2), \dots, (\mathcal{X}_{2N_0}, y_2)$, and so on. In total, this creates a training data set consisting of pairs $(\mathcal{X}_j, y_j)$, $j = 1, \dots, N$ ($N = mN_0$), where each $\mathcal{X}_i$ is the $k \times k \times T$ CAT of an observed dynamics on subgraph F_i and y_i is the corresponding synchronization indicator of the dynamics on the whole graph G .

We also propose an alternative way to generate a training data set when the whole graph G is not available. Suppose we have the same k -node subgraphs $F_1, \dots, F_{N_0}$ in G sampled through our subgraph sampling oracle described in Section III B, but assume that generating many instances of the coupled oscillator dynamics on the whole graph G is computationally prohibitive. In this case, we can simply run the $2m$ dynamics on the sampled subgraphs F_i and record whether the subgraph dynamics synchronize or not with the indicator variable $\tilde{y}_{2m(i-1)+j}$. Denoting by $\tilde{\mathcal{X}}_{2m(i-1)+j}$ the $k \times k \times T$ CAT of the j th dynamics solely run on F_i , this gives us the training examples $(\tilde{\mathcal{X}}_\ell, \tilde{y}_\ell)$ for $\ell = 1, \dots, N (= mN_0)$.

B. How to learn β given $\mathcal{D}$

Once we have $\mathcal{D}$, we can estimate the regression coefficients in β from a set of training examples by solving the standard logistic regression optimization problem. Namely, using the latent dynamics filters in $\mathcal{D}$, we can form the $N \times r$ proximity score matrix $\mathbf{H}$, whose (i, j) coefficient is given by

$$\mathbf{H}[i, j] := \langle \mathcal{X}_i, \mathcal{F}_j \rangle. \quad (10)$$

Note that the i th column of $\mathbf{H}$ gives the proximity score vector for the i th observation X_i . Then joint log-likelihood of observing $(y_1, \dots, y_N)$ under LLDM is

$$\log L(y_1, \dots, y_N | \mathcal{X}_1, \dots, \mathcal{X}_N) = \sum_{i=1}^N \log \mathbb{P}(Y = y_i | \mathbf{H}) \quad (11)$$

$$= \sum_{i=1}^N \{y_i \log \pi_i + (1 - y_i) \log(1 - \pi_i)\}, \quad (12)$$

where π_i is the predictive probability for $\mathcal{X}_i$ under the regression parameter β and the known latent dynamics filters in $\mathcal{D}$. We can then estimate the corresponding regression coefficients, $\hat{\beta}$, from the above joint log-likelihood function with maximum likelihood estimation (MLE) as

$$\hat{\beta} \in \arg \max_{\beta \in \mathbb{R}^r} \sum_{i=1}^N y_i \log \pi_i + (1 - y_i) \log(1 - \pi_i). \quad (13)$$

The above is an unconstrained convex optimization problem, which can be solved by standard first-order optimization algorithms such as gradient descent⁴⁵. When the two classes (i.e., synchronizing and non-synchronizing) are sufficiently balanced in the training data set, then one can employ faster second-order methods such as Newton-Raphson with numerical stability⁴⁶.

As in generalized linear model theory, we have asymptotic normality of the MLE $\hat{\beta}$ estimated from independent random samples $(\mathbf{h}_i, Y_i)$ for $i = 1, \dots, N$ as $N \rightarrow \infty$. Namely, we assume that for a true model parameter β^* ,

$$\text{logit}(\mathbb{E}[Y_i | \mathbf{h}_i]) = \langle \beta^*, \mathbf{h}_i \rangle + \varepsilon_i, \quad (14)$$

where $\mathbf{h}_i$ denotes the i th row of $\mathbf{H}$ and ε_i s are i.i.d. normal random variables with a constant variance $\sigma^2 > 0$ ⁴⁷.

The MLE $\hat{\beta}$ is known to converge to β^* with asymptotically normal fluctuation as the sample size tends to infinity. More precisely,

$$\frac{1}{\sigma} \sqrt{n}(\hat{\beta} - \beta^*) \xrightarrow{D} N(\mathbf{0}, \Sigma^{-1}), \quad (15)$$

given that the sample covariance matrix $\frac{1}{N} \mathbf{H}^T \mathbf{H}$ converges to a limiting covariance matrix Σ as $N \rightarrow \infty$ ⁴⁸. This provides a statistically powerful mechanism for determining important covariates, or features, that affect the response according to the LLDM, which we demonstrate in Section V E. In the following sections, we propose how to compute the dictionary $\mathcal{D}$ of latent dynamics filters.

A detailed section about the training data can be found in Appendix B.

C. How to learn $\mathcal{D}$ from observed dynamics $\mathcal{X}_1, \dots, \mathcal{X}_N$

We now know how to learn β given $\mathcal{D}$. In this subsection, we propose some methodologies to learn the dictionary $\mathcal{D} = [\mathcal{F}_1, \dots, \mathcal{F}_R]$ of latent dynamics filters from the observed dynamics in the form of CATs $\mathcal{X}_1, \dots, \mathcal{X}_N$.

A naïve choice of $\mathcal{D}$ is the set of all observed CATs $\mathcal{X}_i$ for $i = 1, \dots, N$. This means that we regard every single observed dynamic becomes a latent dynamics filter. This choice of $\mathcal{D}$ is undesirable since it is computationally expensive to use a very large number N of filters in our model and also it does not provide any reduced-dimensional representation of the observed dynamics, which hinders the interpretability of our method. Instead, we employ a matrix factorization-based approach (See Appendix C) to learn a small set of bases filters of size R , where $R \ll N$ (typically $R \in \{2, 8, 25, 100\}$). Principal component analysis (PCA) is a popular tool for extracting key features and reducing the dimensionality of the data set, but it is not suitable for our purpose since we desire non-negative basis elements for the CATs so that we can interpret the basis elements (filters) as representing latent dynamics on latent subgraphs. Hence, we employ *non-negative matrix factorization* (NMF) (with vectorizing the tensor input) to extract nonnegative latent $k \times k \times T$ latent dynamic filters $\mathcal{F}_1, \dots, \mathcal{F}_R$. See Section C for background and details on NMF.

(i) Feature extraction from observed dynamics: Use *non-negative matrix factorization* (NMF) to extract R nonnegative basis tensors $\mathcal{F}_1, \dots, \mathcal{F}_R$ from $\mathcal{X}_1, \dots, \mathcal{X}_N$.

This approach is especially useful when we may not have much prior knowledge about the underlying graph and dynamical system to meaningfully distill out parts of the training data.

There are a number of sufficient conditions on coupled oscillator systems that are guaranteed to lead to global synchronization or non-synchronization (see Sec. VB). In scenarios like ours, where one has some existing knowledge about the dynamical system at hand and might want to emphasize higher and finer-detailed interpretability of the learned filters, one can ‘distill out’ a certain portion of training data before we apply NMF for interpretable feature extraction. This suggests the following variant of the previous method (i) of learning $\mathcal{D}$ from the observed dynamics:

(ii) Feature extraction from observed dynamics after distillation: Subsample the ‘most relevant’ observed dynamics $\mathcal{X}_{i_1}, \dots, \mathcal{X}_{i_m}$ using knowledge from coupled oscillator systems, for e.g., dense/sparse underlying graphs and concentrated initial configuration (see Section VB) and then apply NMF as in (i).

Note that (ii) is an example of utilizing the tight interplay between knowledge of these coupled oscillators that originates from the literature along with machine learning-based knowledge.

D. How to learn $\mathcal{D}$ and β jointly

Combining the procedures in the previous two sections, we can first learn a dictionary of learn latent dynamics filters $\mathcal{D}$ from the observed dynamics $\mathcal{X}_1, \dots, \mathcal{X}_N$ without using the synchronization indicators $y_1, \dots, y_N$ and then learn the regression coefficients in β using the learned $\mathcal{D}$ with synchronization indicators $y_1, \dots, y_N$. However, it is also possible to

learn $(\mathcal{D}, \beta)$ jointly from the training examples $(\mathcal{X}_i, y_i)$ for $i = 1, \dots, N$, and we argue that such joint learning is beneficial for our goal of addressing Problem II.1. Recall that the latent dynamics filters $\mathcal{F}_1, \dots, \mathcal{F}_R$ in $\mathcal{D}$ should ultimately satisfy the following two objectives: (1) (*data reconstruction*) They represent patterns in dynamics on subgraphs that are rich enough so that any observed dynamics on a subgraph can be approximately reconstructed by a (nonnegative) linear combination of them; and (2) (*class-discrimination*) they represent patterns in dynamics on subgraphs that are the most effective in discriminating eventual synchronization and non-synchronization on G . If we learn $\mathcal{D}$ only from the observed dynamics as in Section IV C, it may satisfy (1), but not necessarily (2).

From this perspective, we also propose to use *supervised matrix factorization* (SMF)^{29,49} to learn low-rank latent dynamic factors that offer interpretable, data-reconstructive, and class-discriminative features, addressing the challenge of satisfying the two objectives (1) and (2) simultaneously. SMF is similar to NMF in that it extracts a set number of latent features from the observed data set for interpretable dimension reduction, but the matrix factorization process is supervised by the class labels so that the latent features can also be class-discriminative. For this, we formulate a non-convex constrained optimization problem to jointly learn β and $\mathcal{D}$ and solve that problem iteratively via block-coordinate descent type methods (see Appendices B and D).

Note that this approach entails a setup for LLDM that is similar to that of a two-layer neural network^{50,51} with one input, one hidden, and one output layer (See Figure 1a). Hence, it is possible to use backpropagation instead of an SMF-based approach to jointly learn the parameters that represent our filters $\mathcal{F}_1, \dots, \mathcal{F}_R$ and regression coefficients β by using $\mathcal{X}_1, \dots, \mathcal{X}_N$ as our input layer elements which aim to predict the synchronization indicators $y_1, \dots, y_N$. However, using traditional backpropagation for training a feedforward neural network would result in filters with significantly degraded interpretability, due to the loss of the nonnegativity constraint. Bassi et al.¹⁶ showed that complicated training regimes involving deep feedforward neural networks (FFNN) and long-term recurrent convolutional networks (LRCNs)⁵² perform quite similarly to other simple algorithms like random forest, gradient boost, and logistic regression on the task of predicting synchronization. We further demonstrate that in Section V, where the simple and interpretable architecture of LLDM manages to match or outperform the performance of a multi-layer neural network architecture that is much more complex in comparison, while also maintaining the interpretability of the learned filters.

V. RESULTS

We now report on the performance of LLDM for synchronization prediction tasks. We consider two cases, where (1) one seeks to predict the synchronization indicator on small subgraphs using the dynamics on the subgraphs observed during a short time period; and (2) one seeks to predict the synchronization indicator on the whole (parent) graph using the

dynamics restricted on several subgraphs observed during a short time period. We also discuss the goodness-of-fit of our model, as well as the interpretability of the learned filters. Experimental details, hyperparameter choices, and model architectures can be found in Appendix B.

A. Networks

In our experiments, we take the large-scale graph G to be one of the following three types of networks, (1) UCLA, (2) CALTECH, and (3) networks generated from the Newmann-Watts-Strogatz (NWS) model. UCLA and CALTECH networks are part of the FACEBOOK100 dataset⁵³, where the nodes represent users in the respective Facebook networks, and the edges encode Facebook ‘friendships’ between these accounts. Furthermore, for our third large network, we generate a single connected graph using the Newman–Watts–Strogatz (NWS)⁵⁴ small-world network model with $n = 20000$ nodes, $k = 1000$ nearest neighbors in the circulant initial graph, and each non-adjacent pair of nodes gets a new edge independently with probability $p = 0.5$. Lastly, we also generate 500 instances of NWS networks with $n = 300$ nodes, $k = 12$ nearest neighbors, and shortcut edge probability $p = 0.4$. The basic summary statistics of all these three networks can be found in Table I.

TABLE I. Basic Graph Statistics of the Large-scale Graph Networks.

Networks	UCLA	CALTECH	NWS	NWS'
# of graphs	1	1	1	500
# of nodes	20467	769	20000	300
# of edges	747613	16656	1.49e+7	$2.52e+3 \pm 19.4$
Edge density	0.0036	0.0564	0.0750	0.0562

B. Sufficient conditions for synchronization

Some properties of the underlying graphs or dynamics themselves are well known to be critical for the synchronization behavior of a system of coupled oscillators. We state some of these well-known properties below, which come from the traditional literature on coupled oscillator systems. These conditions will be the basis of our baseline predictor, whose performance gives a sense of how easy or difficult a given synchronization prediction problem is.

It is well-known that coupled oscillator systems on dense graphs are relatively easier to synchronize compared to systems on sparse graphs. For example, Kassabov, Strogatz, and Townsend recently showed that Kuramoto oscillators with identical natural frequency on a connected graph where each node is connected to at least $3/4^{\text{th}}$ of all nodes are globally synchronizing for almost all initial configurations⁵⁵. Moreover, a more intricate analysis of Kuramoto oscillators on networks shows that it is possible to generate dense circulant networks that are just able to prevent global synchronization, and sparse circulant networks which tend to globally synchronize⁵⁶.

For FCA, it is known that the dynamics synchronize on a path for a κ -coloring configuration if $\kappa \geq 3$ and further that for finite trees with maximum degree Δ , FCA dynamics do not synchronize if $\Delta \geq \kappa$ and synchronize if $\Delta < \kappa \leq 6$ ^{23,57}. Also, GHM tends to not synchronize on complete or highly dense graphs.¹⁶ also showed that GHM synchronizes on paths.

The next sufficient condition is the *concentration principle*, which is a fundamental observation in coupled oscillators and has been widely used in the clock synchronization literature^{7,17,18} as well as multi-agent consensus problems^{58,59}. This principle, stated below, follows from the fact that the phase difference between any two nodes, when isolated from the rest, monotonically decreases to zero, assuming an identical oscillation frequency.

Concentration Principle *Consider an arbitrarily connected graph G . For Firefly Cellular Automata (FCA), as well as the Kuramoto Model (KM) with identical intrinsic frequency, the given dynamics on G synchronize if all phases at any given time are confined in an open half-circle in the phase space Ω . Furthermore, if all states used in the configuration X_t are confined in an open half-circle for any $t \geq 1$, then the trajectory on G will eventually synchronize.*

An open half-circle refers to any arc of length $< \pi$ for the continuous phase space $\Omega = \mathbb{R}/2\pi\mathbb{Z}$ and any interval of $< \kappa/2$ consecutive integers (mod κ) for the discrete phase space $\Omega = \mathbb{Z}/\kappa\mathbb{Z}$. This confinement in an open half-circle is what we define a phase configuration as being ‘concentrated’. Further, since the concentration principle does not hold for the Greenberg-Hastings model, we define a phase configuration for GHM to be ‘concentrated’ if it is synchronized. The baseline predictor we use for our experimental validation in Section V is based on the concentration principle: *Predict synchronization is the phase configuration at any time during the observed dynamics is concentrated; otherwise flip a fair coin.* See Appendix B for details.

We take advantage of some of these sufficient conditions when we formulate a theory-informed data distillation approach in Section IV C to learn latent dynamic filters for LLDM.

C. Model validation I: Subgraph level

Here we discuss the performance of the latent linear dynamics model (LLDM) at the subgraph level for synchronization prediction. We discuss the results of synchronization prediction using LLDM based on joint-optimization using SMF (labeled ‘LLDM’ in Table II) as well as the theory-informed data distillation approach (see Sec. IV C (ii)) (labeled ‘LLDM-T’ in Table II). We consider the case of 10-, 20-, and 30-node subgraphs sampled using the subgraph sampling oracle in Sec. III B from the 20000-node NWS parent graph (in Table I). We generated training and testing data sets by running Kuramoto, FCA, and the GHM dynamics on the sampled subgraphs. See Appendix B for more details on generating data sets for the experiments.

TABLE II. Prediction accuracy of various methods for FCA and Kuramoto dynamics on subgraphs with k number of nodes where $k \in \{10, 20, 30\}$ sampled from a large-connected NWS parent graph. All accuracy values are an average of 5 seeds. The highest accuracy for each setting is indicated in **bold** font.

	FCA			Kuramoto		
	$k = 10$	$k = 20$	$k = 30$	$k = 10$	$k = 20$	$k = 30$
Baseline	80.2	69.6	69.4	56.5	66.1	64.3
LogReg	92.7	94.8	76.4	83.5	80.1	80.2
FFNN	92.2	95.4	82.8	84.2	83.4	79.2
LLDM-T (R=2)	82.7	81.3	75.9	74.3	76.0	80.1
LLDM-T (R=8)	86.5	81.9	75.7	75.2	77.1	80.3
LLDM (R=2)	91.5	93.2	77.2	81.4	77.5	78.3
LLDM (R=8)	93.0	94.2	84.8	82.1	78.4	78.4

We used extremely low-rank LLDM with $R \in \{2, 8\}$ and compared the prediction accuracy with that of the baseline predictor, logistic regression, and feedforward neural network. A subset of our experimental results is reported in Table II. For a detailed set of results on different subgraph sizes, parent networks, and dynamics models, see Appendix F.

Consider the performance of LLDM using SMF on FCA and Kuramoto dynamics in Table II. We observe that LLDM outperforms the baseline in all settings. We also observe that LLDM performs well across all three subgraph sizes for both the Kuramoto and FCA dynamics. In some cases, our method outperforms logistic regression as well as FFNN, while in other cases, LLDM is still quite close to the prediction accuracies of black-box methods.

Additionally, there seems to be a trade-off between choosing the number R of dictionary atoms to learn from our data. Generally, a higher rank parameter R leads to a higher accuracy as more atoms can effectively capture more fine-grained features of the data. Furthermore, it can be seen that for the problem instances we created, it is easier to predict the synchronization of the FCA dynamics than for Kuramoto dynamics. Furthermore, the prediction task becomes harder as the subgraph size increases, as perhaps the synchronization indicator on a larger subgraph depends on more complex patterns in dynamics. We also observe that the theory-informed LLDM-T in most settings performs worse compared to the SMF-based LLDM, but the accuracies are in general still comparable. This shows the potential of harnessing a theory-informed approach if we have appropriate knowledge about the dynamical systems, especially in settings where we have a large number of noisy observations.

D. Model validation II: Global level

Next, we use LLDM to predict synchronization on a large graph using information only at the subgraph level. Here, we take G to be one of the 500 instances of the 300-node graphs in the dataset NWS' in Table I. We run the FCA and Kuramoto dynamics on G for $T' = 50$ and 100 iterations, respectively, and let y_G be the indicator of whether the system on G is globally synchronized at time T' . Half of them globally syn-

chronize at time T' (so $y_G = 1$) and the other half do not (so $y_G = 0$), which are split into 80% training set and 20% testing set. From each G , we sample a single trajectory of 50 iterations of the MCMC k -path ($k \in \{10, 20, 30\}$) motif sampling algorithm (see Section III B), which gives a sequence of k -node subgraphs $F_1, \dots, F_{50}$. We then restrict the dynamics on G on the sampled subgraphs, thereby creating pairs $(\mathcal{X}_1, y_G), \dots, (\mathcal{X}_{50}, y_G)$ of $k \times k \times T$ CATs and (global) synchronization indicator, where T is varied between 10 and 100. In this way, we create a total of 500×50 pairs of observed subgraph dynamics and synchronization indicators.

We use a block-minimization algorithm for SMF²⁹ on the training set to jointly learn the dictionary of latent dynamic filters $\mathcal{D}$ and the vector of regression coefficients β , where we used three rank parameters $R \in \{4, 25, 100\}$. We can then use the trained LLDM on the testing data set and approximately compute the predictive probability $\bar{p}_T(G)$ that the dynamics on G will eventually synchronize with the sample average in (7) and (8). For more details of the experimental setup, see Appendix B. The experimental results for predicting the synchronization on G from the subgraph dynamics are shown in Figure 3.

It is important to note that our method of aggregating subgraph-level predictions to form a single global-level prediction, as we proposed in Section III C, is a general procedure that can be applied to any subgraph-level predictor. That is, if we have a model that computes a predictive probability $p_T(F)$ of global synchronization on G based on a T -iterations of dynamics observed on a subgraph F , then we can compute the predictive probability for many subgraphs F and take their mean to be the predictive probability of global synchronization as in (6). We compare the performance of LLDM for rank $R \in \{4, 25, 100\}$ against the baseline predictor, logistic regression, and FFNN, where we use the same local-global aggregation (6) for all methods for a fair comparison.

We first observe that all three methods, logistic regression, FFNN, and LLDM outperform the baseline for both dynamics models on almost all subgraph sizes. Furthermore, it holds consistent from the subgraph-level experiments that prediction of synchronization on FCA is in general easier than that of Kuramoto, as we see that most methods can perfectly predict the emergent properties of the parent graph when shown all 50 iterations of the evolution of dynamics, while for Kuramoto the accuracy peaks at around 90% when the model is shown 100 iterations of dynamics evolution.

Second, the global-level prediction task here becomes easier if we increase the subgraph size, contrary to the subgraph-level prediction task in Section V C, as more information on the global dynamics is revealed by observing dynamics on larger subgraphs.

Third, observe that LLDM with ranks $R = 25$ and $R = 100$ consistently matches or outperforms all other methods. LLDM with $R = 4$ seems to lag behind LLDM with higher ranks and other methods, but still provides a competitive prediction accuracy, indicating that even with just 4 latent dynamics filters extracted from the vast amount of data, we can predict reasonably and in some cases even match LLDM with much higher ranks and other methods.

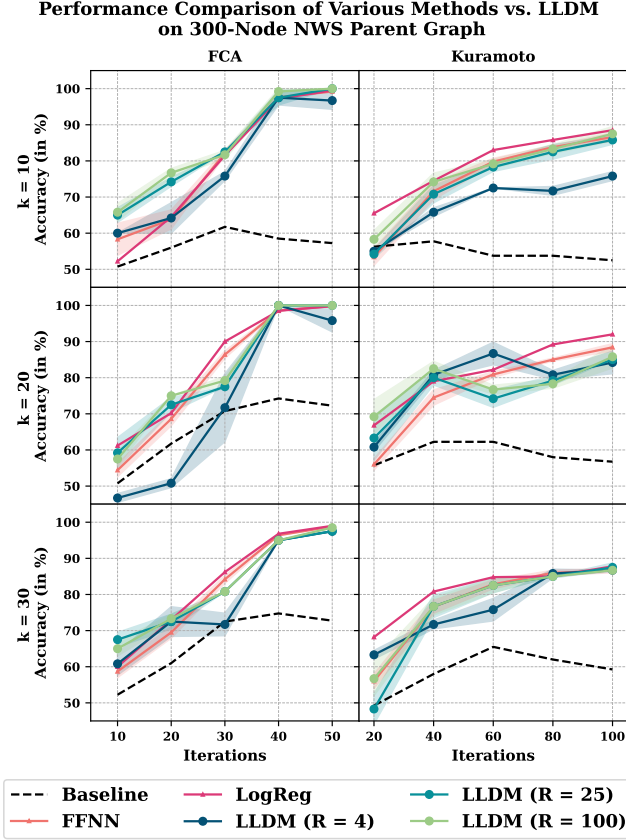


FIG. 3. Prediction accuracy of various methods on large 300-node NWS graphs by using subgraphs of sizes 10, 20, and 30. The prediction accuracies and the standard deviation error shades are shown for varying levels of iterations of dynamics data shown in these models.

Fourth, the error bars in Figure 3 represent one standard deviation of the prediction accuracy computed over ten runs of learning $(\mathcal{D}, \beta)$ by SMF and the prediction of LLDM, where the randomness comes from the random initialization of SMF algorithm. From these error bars, we observe that LLDM with rank $R = 4$ has a higher fluctuation of prediction accuracy compared to its higher-rank counterparts. This suggests a potential trade-off that comes with selecting the appropriate rank for such large-scale synchronization prediction. While low-rank LLDM proves to be computationally efficient due to being light-weight by learning fewer atoms in the dictionary, this comes at the cost of a higher uncertainty in the prediction due to not a large amount of variation of the data being explained by just 4 atoms.

Lastly, we investigate how the overall prediction accuracy is affected by the number N of subgraphs we average the predictive probabilities over (by using (8)) to get an estimate of the population average (6). Figure 4a verifies that the recursively averaged predictive probabilities $p_{T:N}$ in (8) indeed converge as we increase the sample size N . Figure 4b shows that the overall prediction accuracy on the test set increases linearly as N increases, saturating around $N = 25$ samples for FCA and $N = 35$ samples for Kuramoto dynamics.

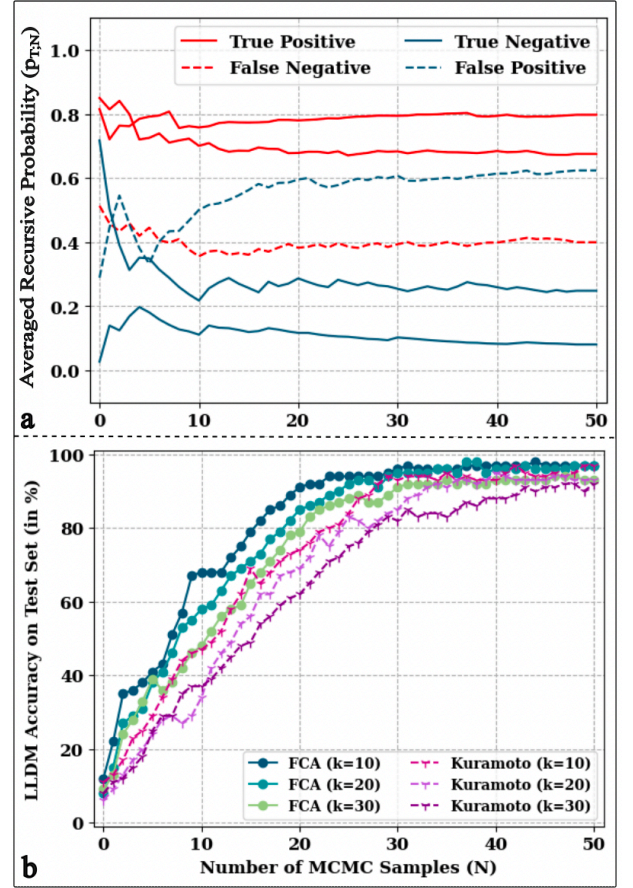


FIG. 4. (Panel a) Examples of recursively averaged probability for Kuramoto dynamics on 20-node subgraphs. We observe that the predicted probability tends to converge in the first 20-25 MCMC samples. (Panel b) Classification accuracy of a pre-trained rank-4 LLDM dictionary on the test set.

Overall, these results strongly suggest that our approach of utilizing subgraph-level predictions to extrapolate to the parent graph level helps to tackle the prediction of complex dynamics on large graphs very effectively and in a compute-efficient manner with a simple framework.

E. Model Validation III: Goodness of Fit – Linearizing Nonlinear Dynamics

Recall that LLDM seeks to model the synchronization indicator of nonlinear dynamical systems via a linear representation of some latent dynamical patterns. The successful experiments in the previous sections demonstrate that the synchronization indicator may indeed be modeled as a linear function of some latent features observed in nonlinear dynamics. In this section, we provide further evidence for this claim by utilizing statistical analysis for generalized linear models such as goodness-of-fit and deviance residual plots.

We first discuss a visual heuristic for goodness-of-fit in linear regression. Let Y denote a univariate response variable and

$X = [X_1, \dots, X_R]$ denote a vector of covariates. We assume the conditional expectation $y := \mathbb{E}[Y|X]$ of Y given X as a linear function $\hat{y} := \beta^T X$, where $\beta \in \mathbb{R}^R$ is a vector of regression coefficients. However, there could be a higher order dependence between y and the covariates. To see whether a linear model has a goodness-of-fit, we investigate the residual $\hat{r}$ below as a function of the covariates:

$$\hat{r} := y - \hat{y} = O(X^p), \quad (16)$$

where $O(X^p)$ denotes a higher-order, polynomial relationship among the covariates in the data. We can plot the residual $\hat{r}$ by the estimate of the fitted values in regression, $\hat{y}$, and in practice, it is often the case that we will see a nonlinear pattern before any model tuning or data transformation is performed in linear regression.

The importance of observing the relationship between residuals by their fitted values is that it serves as a heuristic measure for the goodness-of-fit of a specified linear model. If any nonlinearities were present substantially from this visual heuristic, it would suggest to a practitioner that higher-order interactions between covariates must be incorporated. For example, a cubic residual plot would suggest incorporating cubic order terms between covariates in a more complex model to capture the cubic nonlinearities that may be missed in a simpler linear model of purely first-order covariates.

In our setting for logistic regression, we utilize ‘deviance residuals’⁶⁰, which is the appropriate choice for modeling residuals in logistic regression models that are as follows:

$$d_i := \text{sign}(e_i) [-2(y_i \log \hat{p}_i + (1 - y_i) \log(1 - \hat{p}_i))]^{1/2},$$

for data point i , with $e_i = y_i - \hat{p}_i$, where y_i is the corresponding label for observation i , and $\hat{p}_i$ is the predicted response after fitting a logistic regression model as in

$$\hat{p}_i := \frac{\exp(\beta^T \mathbf{X}_{i,:})}{1 + \exp(\beta^T \mathbf{X}_{i,:})}, \quad (17)$$

where $\mathbf{X}_{i,:}$ is the i th row of $\mathbf{X}$, corresponding to the i th observation.

We form a deviance residual plot with pairs of data points between fitted values and deviance residuals, $(\hat{p}_i, d_i)$, and of a smoothing spline approximating a residual curve of $(\hat{p}_i, d_i)$. (This residual curve can be produced from various options of smoothing spline algorithms.) Since we are interested in modeling the residual data after fitting our LLDM, we utilize a simple univariate spline algorithm⁶¹. The resulting smoothing spline represents the distribution of the deviance residuals away from $y = 0$ across possible fitted values such that a more sporadic nonzero curve—especially at the tail ends—would indicate poor model fit, and a flatter, more zero-valued curve would mean sufficient linear fit as the deviance residual appears to be approximately zero across all possible fitted values.

Figure 5 shows multiple goodness-of-fit plots of deviance residuals by fitted values where we have smoothing mostly flat spline curves. In particular, there is a good linear fit for LLDM utilizing ranks 2, 4, 8, and 16 for FCA, Kuramoto, and

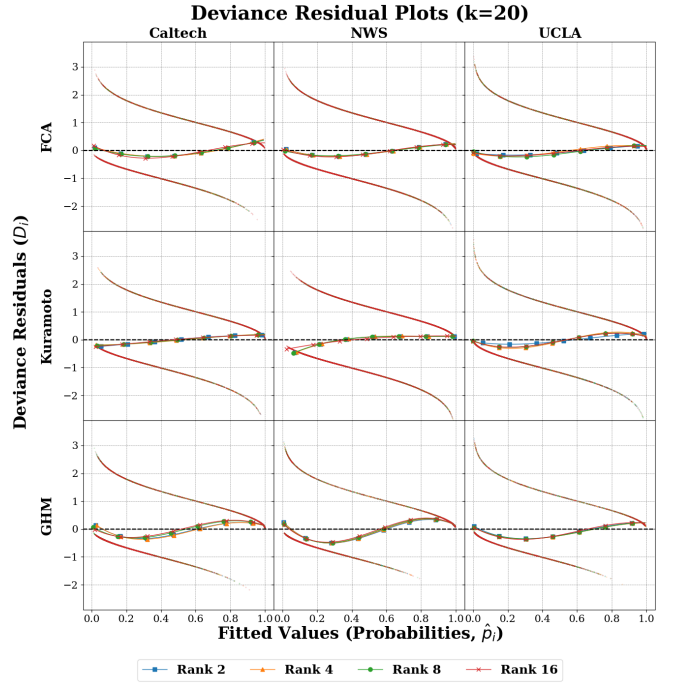


FIG. 5. Goodness-of-fit with deviance residuals for LLDM on subgraphs with $k = 30$ nodes sampled from CALTECH, UCLA, and NWS graphs with FCA, Kuramoto, and GHM models on rank $R \in \{2, 4, 8, 16\}$. We generally observe that across all ranks R , our model for 20-node subgraphs has a very good linear fit. The univariate smoothing spline for each rank smooths and interpolates the $(\hat{p}_i, d_i)$ points into a polynomial curve, where we can choose the degree of the polynomial, k , as well as a smoothing factor, s . We choose $k = 3$ and $s = R - \sqrt{2R}$ as a common heuristic choice.

GHM on all three networks, UCLA, CALTECH and NWS for subgraphs of 30 nodes. More plots are given in the Appendix H in Figure 14 for subgraphs of various sizes $k = 10, 15, 25$, and 30 (we see that there is goodness-of-fit for these models as well). Therefore, LLDM not only performs well in predicting the synchronization of coupled oscillator systems in terms of accuracy but also in representing the synchronization indicator in a latent linear form.

An important observation in Figure 5 is that for higher filter matrix ranks there is less linear LLDM fit. An intuitive explanation for this behavior is that the LLDM does not benefit from greater linear model complexity and that additional covariates contribute noise to the model. We see cases where a rank 2 LLDM is more than sufficient to linearly capture the synchronization indicator as a function of the proximity scores of latent dynamic patterns.

F. Interpretability of the learned latent dynamics filters

Now that we have validated the performance of LLDM at both the subgraph and global level using accuracy as well as the goodness-of-fit for the coefficients, we now proceed to discuss the *interpretability* of latent dynamics filters (‘filters’ hereafter), as seen in Figure 6, Figure 7, and in Appendix G.

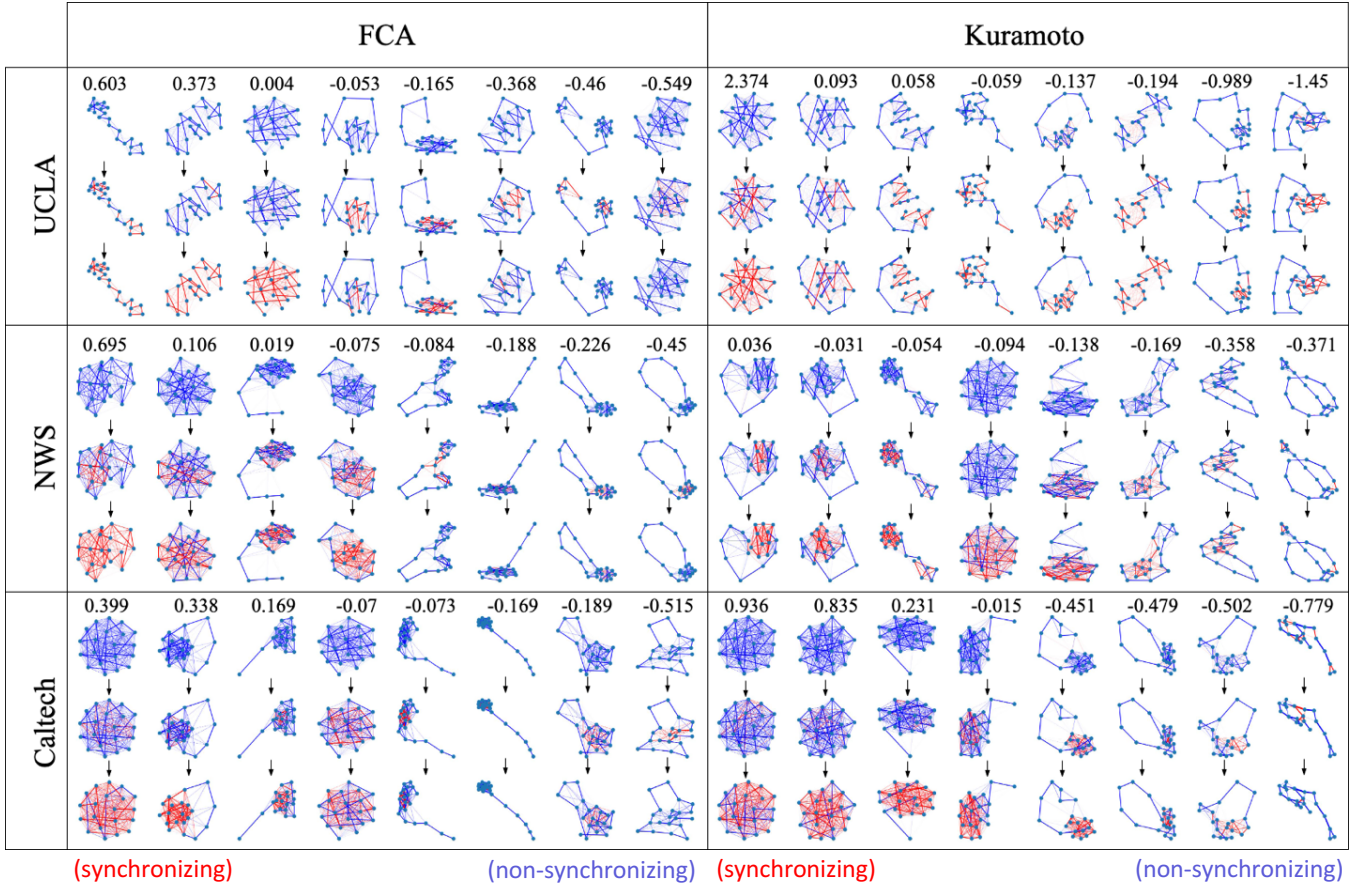


FIG. 6. The latent dynamics filters learned by rank-8 SMF for FCA and Kuramoto dynamics on 20-node CALTECH, UCLA, and NWS networks. Logistic regression coefficients associated with the latent dynamic filters are shown on top of each filter (+ = synchronization and - = non-synchronization). Each latent dynamic filter is a tensor of shape $k \times k \times T$ with $k = 20$, and we only show three temporal snapshots of such tensor in each column. The colors on the edges range from blue (indicating a large phase difference) to red (indicating a small phase difference). For instance, the leftmost latent dynamic filter for CALTECH subgraphs and FCA dynamics (bottom left) starts with densely connected blue edges ending up with all red edges, indicating a dynamic pattern on densely connected subgraphs with large mutual phase differences leading to synchrony.

Here, we consider rank $R = 8$ filters for LLDLM learned by SMF algorithm from the Kuramoto, FCA, and GHM dynamics restricted on the 20-node subgraphs sampled from parent networks of CALTECH, UCLA, and NWS in Table I. Similar plots for subgraphs of size $k \in \{10, 15, 25, 30\}$ can be found in Appendix G.

For each of the tile in both Figures 6 and 7, the eight filters are represented by eight columns. Recall that each filter is a CAT of size $k \times k \times T$, representing latent dynamics on k -node subgraphs. We only show three equally-spaced snapshots of such tensors as three $k \times k$ matrices in each column, where the time evolution is indicated by the arrows. The colors on the edges range from blue (indicating a large phase difference) to red (indicating a small phase difference). For instance, on the one hand, the leftmost filter for CALTECH subgraphs and FCA dynamics (bottom left) starts with densely connected blue edges ending up with all red edges, indicating a dynamic pattern on densely connected subgraphs with large mutual phase differences leading to synchrony. On the other hand, the rightmost filter for CALTECH and FCA dynamics

ends up with relatively sparse blue edges, indicating that the corresponding subgraph is sparsely connected and the phase differences along its edges are large at time T . The filters are in decreasing order for their corresponding logistic regression coefficients (shown on top), where positive (resp. negative) coefficients indicate positive association with eventual (resp. non-)synchronization. So the filters to the left are more representative of what kind of input dynamics and graphs are more likely to synchronize and as we move to the right of the plots, change to being more likely to not synchronize.

We now examine some specific details that can be observed from Figure 6, where we consider the case of the FCA and Kuramoto dynamics on CALTECH as the underlying graph on the last row of tiles in the figure. We see that for both dynamics, the leftmost filters indicating patterns for synchronization have the densest edges, and almost the entire graph tends to synchronize as time goes on as almost all edges become red. Moving along the columns to the right, we see that almost all filters that have positive or very small negative coefficients tend to have graphs that are either densely connected or are

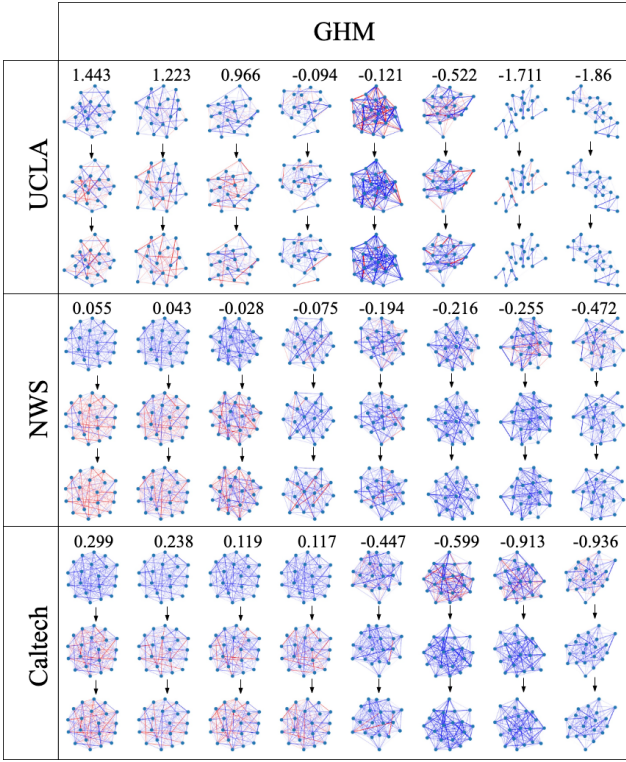


FIG. 7. The latent dynamics filters learned by rank-8 SMF for GHM dynamics on 20-node CALTECH, UCLA, and NWS networks.

mostly dense with short paths that go out of a community (a densely connected subset of nodes) structure. Moreover, for such cases, we again observe that the communities that we observe tend to synchronize by the end of the training iteration T but the nodes on the short extending paths do not. We further move right to observe filters that are more likely to not synchronize, observing most of the weakly non-synchronizing filters have a small community structure that weakly synchronizes (see partially communities consisting of red edges) at later times. However, there is a large extending path or cycle that is non-synchronizing.

Lastly, observe that the filters associated with the most negative coefficients show that the graph structures are very sparse and do not contain any substantial communities. Moreover, almost the entire graph remains not synchronized at the end of the training iteration T . Therefore, the latent dynamic patterns captured by these filters with the associated regression coefficients are consistent with the existing knowledge of sufficient conditions for synchronization and non-synchronization. A similar observation holds for the filters learned from the dynamics on subgraphs of NWS.

Next, we discuss filters learned from oscillator dynamics on subgraphs of UCLA. Recall that UCLA is an order of magnitude sparser than other networks CALTECH and NWS (see Table I), so the filters learned from the subgraphs of UCLA show sparser edge density than the ones learned from subgraphs of the other networks. As before, the regression coefficients for Kuramoto dynamics tend to be positively correlated with edge density in the filters, indicating that Kuramoto dynamics on dense sub-

graphs tend to be synchronizing. The filter with the largest regression coefficient is dense on the whole, and the ones that show weak synchronization are weakly dense yet seem to have a loose community structure and no long-extending paths. Similarly, for the non-synchronizing filters, we observe a very loose to no community structure along with long paths and cycles. However, we see different trends for the filters for the FCA dynamics. There, the regression coefficient seems to depend on the specific topology of the subgraphs and the dynamics on them, rather than just the edge density of the subgraphs we observe the dynamics on.

Finally, we move on to discuss Figure 7 where we look at the rank $R = 8$ filters learned for the GHM dynamics on subgraphs of the three networks. The filters, in this case, look quite similar to each other, unlike the Kuramoto and FCA dynamics in Figure 6, which suggests that for GHM dynamics, fewer than $R = 8$ filters would suffice to fit our LLDM. Indeed, we obtain similar prediction accuracy for GHM dynamics with $R = 2$ as with $R = 8$, see Table F in Appendix F.

Furthermore, we observe that most of the filters with a positive association with synchronization seem to be dense overall, and the edges in the filters tend to become red (synchronized) overall within the first half of the training iterations T . On the other hand, the filters that correspond to non-synchronization (rightmost columns in Fig. 7) seem to be relatively sparser or have a path-like structure. There we observe synchronized edges (in red) developing within the filters in time, but most edges remain non-synchronized (in blue).

VI. CONCLUSION

In this paper, we propose a latent linear model that effectively linearizes highly nonlinear dynamics resulting from coupled oscillators interacting on graphs. A fundamental concept we introduce is ‘latent dynamic filters’, which encode some key dynamical patterns associated with synchronization/non-synchronization of the system, which enable subgraph-level synchronization prediction by using a latent linear model. These filters are directly learned from the data by incorporating supervised matrix factorization techniques. The predictive probability of global synchronization is computed by averaging many subgraph-level predictions along an MCMC subgraph sampling trajectory.

Our framework has the benefit of being simple and lightweight, and we carried out an extensive study to show that it matches or outperforms traditional black-box methods on the prediction of synchronization of the coupled oscillator dynamics on large graphs. We provide an efficient recursive averaging algorithm to combine many subgraph-level predictions, whose convergence is both theoretically and experimentally justified. A statistical validation of our model was also provided. Finally, we provide a computational framework to extract key patterns responsible for synchronization/non-synchronization from many instances of observed coupled oscillator dynamics. This provides added interpretability of our method for a better understanding of rich nonlinear dynamics of coupled oscillators. We hope that our work will inspire

a new line of research harnessing the potential of our simple and interpretable computational framework to help better understand various complex dynamics systems beyond coupled oscillators.

VII. MATERIALS

The code for the algorithms and simulations used in this work is provided in <https://github.com/zwu363/Interpretable-ML-Sync>.

ACKNOWLEDGEMENTS

This work is supported by NSF Grants DMS-2010035 and DMS-2206296.

REFERENCES

- ¹S. H. Strogatz, “From kuramoto to crawford: exploring the onset of synchronization in populations of coupled oscillators,” *Physica D: Nonlinear Phenomena* **143**, 1–20 (2000).
- ²J. A. Acebrón, L. L. Bonilla, C. J. P. Vicente, F. Ritort, and R. Spigler, “The kuramoto model: A simple paradigm for synchronization phenomena,” *Reviews of modern physics* **77**, 137 (2005).
- ³S. Nair and N. E. Leonard, “Stable synchronization of rigid body networks,” *Networks and Heterogeneous Media* **2**, 597 (2007).
- ⁴R. Pagliari and A. Scaglione, “Scalable network synchronization with pulse-coupled oscillators,” *Mobile Computing, IEEE Transactions on* **10**, 392–405 (2011).
- ⁵F. Dörfler and F. Bullo, “Synchronization and transient stability in power networks and nonuniform kuramoto oscillators,” *SIAM Journal on Control and Optimization* **50**, 1616–1642 (2012).
- ⁶F. Dörfler and F. Bullo, “On the critical coupling strength for kuramoto oscillators,” in *Proceedings of the 2011 American Control Conference (IEEE, 2011)* pp. 3239–3244.
- ⁷J. Klinglmayr, C. Kirst, C. Bettstetter, and M. Timme, “Guaranteeing global synchronization in networks with stochastic interactions,” *New Journal of Physics* **14**, 073031 (2012).
- ⁸P. S. Skardal and A. Arenas, “Higher order interactions in complex networks of phase oscillators promote abrupt synchronization switching,” *Communications Physics* **3**, 218 (2020).
- ⁹R. M. D’Souza, J. Gómez-Gardenes, J. Nagler, and A. Arenas, “Explosive phenomena in complex networks,” *Advances in Physics* **68**, 123–223 (2019).
- ¹⁰J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Židek, A. Potapenko, *et al.*, “Highly accurate protein structure prediction with alphafold,” *Nature* **596**, 583–589 (2021).
- ¹¹Y. Koren, R. Bell, and C. Volinsky, “Matrix factorization techniques for recommender systems,” *Computer* **42**, 30–37 (2009).
- ¹²H. Mehta, S. Rendle, W. Krichene, and L. Zhang, “Alx: Large scale matrix factorization on tpus,” *arXiv preprint arXiv:2112.02194* (2021).
- ¹³H. Fan, L.-W. Kong, Y.-C. Lai, and X. Wang, “Anticipating synchronization with machine learning,” *Physical Review Research* **3**, 023237 (2021).
- ¹⁴S. Guth and T. P. Sapsis, “Machine learning predictors of extreme events occurring in complex dynamical systems,” *Entropy* **21**, 925 (2019).
- ¹⁵S. N. Chowdhury, A. Ray, A. Mishra, and D. Ghosh, “Extreme events in globally coupled chaotic maps,” *Journal of Physics: Complexity* **2**, 035021 (2021).
- ¹⁶H. Bassi, R. P. Yim, J. Vendrow, R. Koduluka, C. Zhu, and H. Lyu, “Learning to predict synchronization of coupled oscillators on randomly generated graphs,” *Scientific reports* **12**, 1–15 (2022).
- ¹⁷J. Nishimura and E. J. Friedman, “Robust convergence in pulse-coupled oscillators with delays,” *Physical review letters* **106**, 194101 (2011).
- ¹⁸H. Lyu, “Global synchronization of pulse-coupled oscillators on trees,” *SIAM Journal on Applied Dynamical Systems* **17**, 1521–1559 (2018).
- ¹⁹T. N. Thiem, M. Kooshkbaghi, T. Bertalan, C. R. Laing, and I. G. Kevrekidis, “Emergent spaces for coupled oscillators,” *Frontiers in computational neuroscience* **14**, 36 (2020).
- ²⁰C. M. Bishop, *Pattern recognition and machine learning* (springer, 2006).
- ²¹A. Hefny, C. Downey, and G. J. Gordon, “Supervised learning for dynamical system learning,” *Advances in neural information processing systems* **28** (2015).
- ²²K. Itabashi, Q. H. Tran, and Y. Hasegawa, “Evaluating the phase dynamics of coupled oscillators via time-variant topological features,” *Physical Review E* **103**, 032207 (2021).
- ²³H. Lyu, “Synchronization of finite-state pulse-coupled oscillators,” *Physica D: Nonlinear Phenomena* **303**, 28–38 (2015).
- ²⁴J. M. Greenberg and S. Hastings, “Spatial patterns for discrete models of diffusion in excitable media,” *SIAM Journal on Applied Mathematics* **34**, 515–523 (1978).
- ²⁵Z. Chen, T. Anglea, Y. Zhang, and Y. Wang, “Optimal synchronization in pulse-coupled oscillator networks using reinforcement learning,” *PNAS nexus* **2**, pgad102 (2023).
- ²⁶F. Mahlow, B. Çakmak, G. Karpat, İ. Yalçınkaya, and F. Fanchini, “Predicting the onset of quantum synchronization using machine learning,” *arXiv preprint arXiv:2308.15330* (2023).
- ²⁷D. D. Lee and H. S. Seung, “Learning the parts of objects by non-negative matrix factorization,” *Nature* **401**, 788–791 (1999).
- ²⁸J. Mairal, J. Ponce, G. Sapiro, A. Zisserman, and F. Bach, “Supervised dictionary learning,” *Advances in neural information processing systems* **21** (2008).
- ²⁹J. Lee, H. Lyu, and W. Yao, “Exponentially convergent algorithms for supervised matrix factorization,” To appear in *Neural Information Processing Systems* (2023).
- ³⁰M. W. Berry and M. Browne, “Email surveillance using non-negative matrix factorization,” *Computational & Mathematical Organization Theory* **11**, 249–264 (2005).
- ³¹R. Boutchko, D. Mitra, S. L. Baker, W. J. Jagust, and G. T. Gullberg, “Clustering-initiated factor analysis application for tissue classification in dynamic brain positron emission tomography,” *Journal of Cerebral Blood Flow & Metabolism* **35**, 1104–1111 (2015).
- ³²A. Sitek, G. T. Gullberg, and R. H. Huesman, “Correction for ambiguous solutions in factor analysis using a penalized least squares objective,” *IEEE transactions on medical imaging* **21**, 216–225 (2002).
- ³³M. W. Berry, M. Browne, A. N. Langville, V. P. Pauca, and R. J. Plemmons, “Algorithms and applications for approximate nonnegative matrix factorization,” *Computational statistics & data analysis* **52**, 155–173 (2007).
- ³⁴Y. Chen, X. Wang, C. Shi, E. K. Lua, X. Fu, B. Deng, and X. Li, “Phoenix: A weight-based network coordinate system using matrix factorization,” *IEEE Transactions on Network and Service Management* **8**, 334–347 (2011).
- ³⁵L. Taslaman and B. Nilsson, “A framework for regularized non-negative matrix factorization, with application to the analysis of gene expression data,” *PloS One* **7**, e46331 (2012).
- ³⁶B. Ren, L. Pueyo, G. B. Zhu, J. Debes, and G. Duchêne, “Non-negative matrix factorization: robust extraction of extended structures,” *The Astrophysical Journal* **852**, 104 (2018).
- ³⁷S. Brin and L. Page, “The anatomy of a large-scale hypertextual web search engine,” *Computer networks and ISDN systems* **30**, 107–117 (1998).
- ³⁸R. Luo, “Unraveling low-dimensional network dynamics: A fusion of sparse identification and proper orthogonal decomposition,” *arXiv preprint arXiv:2308.10458* (2023).
- ³⁹H. Lyu, F. Memoli, and D. Sivakoff, “Sampling random graph homomorphisms and applications to network data analysis,” *Journal of machine learning research* **24**, 1–79 (2023).
- ⁴⁰Y. Kuramoto, *Chemical oscillations, waves, and turbulence* (Courier Corporation, 2003).

- ⁴¹H. Lyu, “Time complexity of synchronization of discrete pulse-coupled oscillators on trees,” arXiv preprint arXiv:1610.00837 (2023).
- ⁴²B. Perozzi, R. Al-Rfou, and S. Skiena, “DeepWalk: Online learning of social representations,” in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2014) pp. 701–710.
- ⁴³A. Narayanan, M. Chandramohan, R. Venkatesan, L. Chen, Y. Liu, and S. Jaiswal, “graph2vec: Learning distributed representations of graphs,” arXiv preprint arXiv:1707.05005 (2017).
- ⁴⁴H. Lyu, Y. H. Kureh, J. Vendrow, and M. A. Porter, “Learning low-rank latent mesoscale structures in networks,” arXiv preprint arXiv:2102.06984 (2021).
- ⁴⁵S. P. Boyd and L. Vandenberghe, *Convex optimization* (Cambridge university press, 2004).
- ⁴⁶R. A. Fisher, “On the mathematical foundations of theoretical statistics,” *Philosophical transactions of the Royal Society of London. Series A, containing papers of a mathematical or physical character* **222**, 309–368 (1922).
- ⁴⁷J. A. Nelder and R. W. Wedderburn, “Generalized linear models,” *Journal of the Royal Statistical Society Series A: Statistics in Society* **135**, 370–384 (1972).
- ⁴⁸P. Billingsley, *Convergence of probability measures* (John Wiley & Sons, 2013).
- ⁴⁹J. Lee, H. Lyu, and W. Yao, “Supervised dictionary learning with auxiliary covariates,” arXiv preprint arXiv:2206.06774 (2022).
- ⁵⁰F. Rosenblatt, “Principles of neurodynamics. perceptrons and the theory of brain mechanisms,” Tech. Rep. (Cornell Aeronautical Lab Inc Buffalo NY, 1961).
- ⁵¹D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning internal representations by error propagation,” Tech. Rep. (California Univ San Diego La Jolla Inst for Cognitive Science, 1985).
- ⁵²J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, “Long-term recurrent convolutional networks for visual recognition and description,” in *Proceedings of the IEEE conference on computer vision and pattern recognition* (2015) pp. 2625–2634.
- ⁵³A. L. Traud, P. J. Mucha, and M. A. Porter, “Social structure of facebook networks,” *Physica A: Statistical Mechanics and its Applications* **391**, 4165–4180 (2012).
- ⁵⁴M. E. Newman and D. J. Watts, “Renormalization group analysis of the small-world network model,” *Physics Letters A* **263**, 341–346 (1999).
- ⁵⁵M. Kassabov, S. H. Strogatz, and A. Townsend, “Sufficiently dense kuramoto networks are globally synchronizing,” *Chaos: An Interdisciplinary Journal of Nonlinear Science* **31**, 073135 (2021).
- ⁵⁶A. Townsend, M. Stillman, and S. H. Strogatz, “Dense networks that do not synchronize and sparse ones that do? a3b2 show [editpick]?>,” *Chaos: An Interdisciplinary Journal of Nonlinear Science* **30**, 083142 (2020).
- ⁵⁷H. Lyu and D. Sivakoff, “Synchronization of finite-state pulse-coupled oscillators on $\mathbb{Z}$,” arXiv.org:1701.00319 (2017).
- ⁵⁸L. Moreau, “Stability of multiagent systems with time-dependent communication links,” *Automatic Control, IEEE Transactions on* **50**, 169–182 (2005).
- ⁵⁹B. Chazelle, “The total s-energy of a multiagent system,” *SIAM Journal on Control and Optimization* **49**, 1680–1706 (2011).
- ⁶⁰P. Mcclagh and J. Nelder, “Generalized linear models 2nd ed,” Chapman and Hall: New York (1989).
- ⁶¹P. Dierckx, “An algorithm for smoothing, differentiation and integration of experimental data using spline functions,” *Journal of Computational and Applied Mathematics* **1**, 165–184 (1975).
- ⁶²H. Lyu and Y. Li, “Block majorization-minimization with diminishing radius for constrained nonconvex optimization,” (2023), [arXiv:2012.03503 \[math.OC\]](#).
- ⁶³A. F. Agarap, “Deep learning using rectified linear units (relu),” arXiv preprint arXiv:1803.08375 (2018).
- ⁶⁴N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting,” *The journal of machine learning research* **15**, 1929–1958 (2014).
- ⁶⁵S. Wold, K. Esbensen, and P. Geladi, “Principal component analysis,” *Chemometrics and intelligent laboratory systems* **2**, 37–52 (1987).
- ⁶⁶A. Y. Ng, M. I. Jordan, and Y. Weiss, “On spectral clustering: Analysis and an algorithm,” in *Advances in neural information processing systems* (2002) pp. 849–856.

Appendix A: Models of Coupled Oscillators

Systems of coupled oscillators have been studied over multiple decades^{1–6}. In this work, we consider three popular models of coupled oscillators to study their synchronization behavior, described below.

1. Kuramoto model

The Kuramoto model^{1,2} of coupled oscillators is perhaps one of the most well-studied models in the dynamical system community. Consider a graph $G = (V, E)$ and a continuous phase space $\Omega = \mathbb{R}/2\pi\mathbb{Z}$. The evolution of the phase dynamics of the initial phase configuration $X_0 : V \rightarrow \Omega$ governed by the Kuramoto model of coupled oscillators is determined by the following system of ordinary differential equations in (A1)

$$\frac{d}{dt}X_t(v) = \omega_v + K \sum_{u \in \mathcal{N}(v)} \sin(X_t(u) - X_t(v)) \quad \forall v \in V, \quad (\text{A1})$$

where $\mathcal{N}(v)$ represents the set of neighboring nodes of v in G , ω_v denotes the *intrinsic frequency* of node v , and K denotes the *coupling strength* of the model. We discretize the ordinary differential equation in (A1) so that each ‘step’ in Kuramoto dynamics is given by the following difference equation:

$$X_{t+h}(v) - X_t(v) = h \left(\sum_{u \in \mathcal{N}(v)} K \sin(X_t(u) - X_t(v)) \right), \quad (\text{A2})$$

where we choose a step size of $h = 0.05$ and $K = 1$.

In this paper, we assume the intrinsic frequency of all the nodes in G are identical, or equivalently zero, by using a rotating frame of dynamics without loss of generality. We further assume the coupling strength to be unity. Note that synchronization is an absorbing state, in the sense that if X_τ is constant (e.g., synchronized), then X_t is constant for all $t \geq \tau$. See Figure 8 for an example of the Kuramoto dynamics evolving on an 8×8 grid.

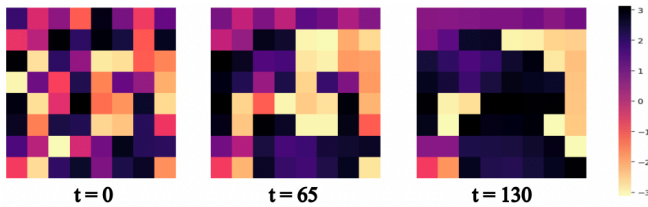


FIG. 8. **Evolution of Kuramoto dynamics.** Representation of the evolution of the Kuramoto dynamics on an 8×8 2D-grid graph with snapshots at time iterations $t = 0$, $t = 65$, and $t = 130$.

2. The Firefly Cellular Automata

The Firefly Cellular Automata (FCA)^{23,57} is a model to study discrete pulse-coupled oscillators. Consider a graph

$G = (V, E)$ and $\kappa \geq 3$ to define $\Omega = \mathbb{Z}/\kappa\mathbb{Z}$ with an ordering $0 < 1 < \dots < \kappa - 1$. The evolution of the phase dynamics of the initial phase configuration is governed by $X_0 : V \rightarrow \Omega$ and we further define $b(\kappa) = \lfloor \frac{\kappa-1}{2} \rfloor$ to be the *blinking color* of the configuration. The time evolution of this κ -colored FCA dynamics is dictated by the update mapping $X \rightarrow X'$ in (A3)

$$X'(v) = \begin{cases} X(v) & \text{if } X(v) > b(\kappa) \text{ and } v \in \mathcal{N}(v') \text{ such that} \\ & v' \in V \text{ has blinking state } b(\kappa) \\ X(v) + 1 & \text{otherwise} \end{cases} \quad (\text{A3})$$

For all experiments in this work, we use the FCA model with $\kappa = 5$. See Figure 9 for an example of 5-color FCA dynamics evolving on an 8×8 grid.

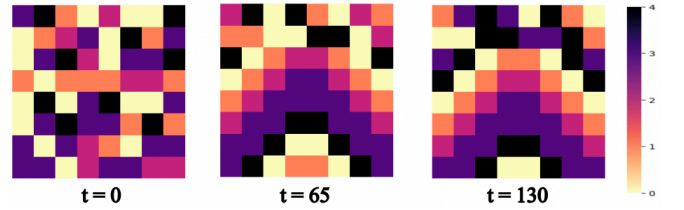


FIG. 9. **Evolution of FCA dynamics.** Representation of the evolution of the FCA dynamics on an 8×8 2D-grid graph with snapshots at time iterations $t = 0$, $t = 65$, and $t = 130$.

3. The Greenberg-Hastings model

The Greenberg-Hastings model (GHM)²⁴ is a popular model for studying discrete patterns of diffusion in excitable media. Consider a graph $G = (V, E)$ and define $\Omega = \mathbb{Z}/\kappa\mathbb{Z}$ with an ordering $0 < 1 < \dots < \kappa - 1$. The evolution of the phase dynamics of the initial phase configuration with GHM is governed by $X_0 : V \rightarrow \Omega$. The time evolution of this κ -colored GHM dynamics is dictated by the mapping $X \rightarrow X'$ in (A4)

$$X'(v) = \begin{cases} 0 & \text{if } X(v) = 0 \text{ and } X(v') \neq 1 \quad \forall v' \in \mathcal{N}(v) \\ 1 & \text{if } X(v) = 0 \text{ and } \exists v' \in \mathcal{N}(v) \text{ s.t.} \\ & X(v') = 1 \\ X(v) + 1 & \text{otherwise} \end{cases} \quad (\text{A4})$$

For all experiments in this work, we use the GHM model with $\kappa = 6$. See Figure 10 for an example of 6-color GHM dynamics evolving on an 8×8 grid.

Appendix B: Training Data, Models, Hyperparameters, and Experimental Details

Baseline Predictor The baseline predictor is based on the concentration principle defined in Section V B, which predicts eventual synchronization of the evolving dynamics if the phase configuration is concentrated at any time during the observed dynamics, and flips an independent fair coin for prediction otherwise.

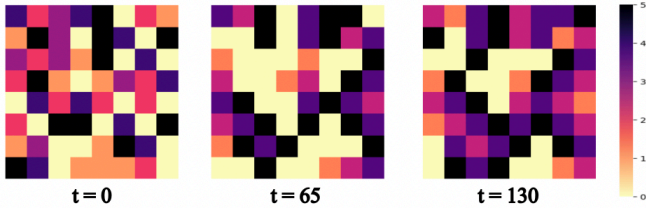


FIG. 10. **Evolution of GHM dynamics.** Representation of the evolution of the GHM dynamics on an 8×8 2D-grid graph with snapshots at time iterations $t = 0$, $t = 65$, and $t = 130$.

Details for the subgraph-level experiments in Section V C.

The data set we use for the subgraph-level experiments in Section V C is generated as follows. From one of the three networks described in Table I, we sample k -node subgraphs $F_1, \dots, F_N$ with $N = 10,000$ using the subgraph sampling oracle in Section III B, where $k \in \{10, 15, 20, 25, 30\}$. On the i th subgraph F_i , we randomly initialize and run T' iterations of Kuramoto, FCA, and GHM dynamics, where $T' = 200, 100$, and 100 , respectively. The data set consists of pairs $(\mathcal{X}_i, y_i)$, $i = 1, \dots, N$, where y_i is the indicator that the system is synchronized at time T' and $\mathcal{X}_i$ is the $k \times k \times T$ CAT that encodes the dynamics on F_i for the first $T < T'$ iterations, where $T = 100, 50$, and 8 iterations for the Kuramoto, FCA and GHM dynamics, respectively. These 10,000 data points are then uniformly randomly divided into training and testing sets, consisting of 80% and 20% of the examples, respectively.

For LLDm-T in Table II, we distill the generated data based on three observations from the literature as described in V B, namely graph density and initial half-circle concentration. In particular, we sample a set of subgraphs from our parent graphs and then distill the top 10% densest and top 10% sparsest, based on their edge density. In addition to these, we also select certain configurations of dynamics-network pairs where the dynamics follow the half-circle concentration, such that the total number of data points is 10,000 which we again split into train and test sets (80% and 20%).

For LLDm and LLDm-T in Table II, we used block minimization-type iterative algorithms for supervised matrix factorization²⁹ and nonnegative matrix factorization algorithms⁶² for 250 iterations, respectively.

Details for the global-level experiments in Section V D.

For the global-level experiments, we take the parent graph G to be one of the 500 instances of the 300-node graphs in the dataset NWS' in Table I. These graphs are generated by the NWS model with the circulant graph with $k = 12$ nearest-neighbors and $p = 0.4$ probability of adding a new edge independently between each non-adjacent pair of nodes. On each G , we simulate the Kuramoto and the 5-color FCA dynamics with random initial configuration for $T' = 50$ and 100 iterations of FCA and Kuramoto dynamics, respectively. This creates 500 pairs of dynamics on G and indicator y_G of whether the system on G is globally synchronized at time T' . We choose the initial configurations randomly so that half of them globally synchronize at time

T' (so $y_G = 1$) and the other half do not (so $y_G = 0$). These pairs are split into 80% training set and 20% testing set. From each G , we sample a single trajectory of 50 iterations of the MCMC k -path ($k \in \{10, 20, 30\}$) motif sampling algorithm (see Section III B), which gives a sequence of k -node subgraphs $F_1, \dots, F_{50}$. We then restrict the dynamics on G on the sampled subgraphs, thereby creating pairs $(\mathcal{X}_1, y_G), \dots, (\mathcal{X}_{50}, y_G)$ of $k \times k \times T$ CATs and (global) synchronization indicator, where T is varied between 10 and 100. In this way, we create a total of 500*50 pairs of observed subgraph dynamics and synchronization indicators. The block-coordinate descent algorithm for SMF⁴⁹ is run for 250 iterations to jointly learn the dictionary $\mathcal{D}$ (for ranks $R \in \{4, 25, 100\}$) of latent dynamic filters and vector of regression coefficients β from the training data set.

FFNN architecture. The FFNN architecture we use for the experiments is one with four fully connected layers, where each intermediate layer has 100 hidden nodes, batch normalization, and uses the ReLU⁶³ activation function. Further, we use a dropout⁶⁴ with $p = 0.25$ on each of our layers to prevent model overfitting.

Supervision tuning parameter. For our Supervised Matrix Factorization (SMF) based experiments (See Section V and Appendix D), we report the best results on doing a grid search on hyperparameter ξ (see (D3)) with choices $\xi \in [0.1, 0.5, 1.0]$. A higher value of ξ indicates that the model will be penalized more for learning filters that do not reconstruct the original data well, but not so much for wrong label predictions, and vice-versa.

Appendix C: Background on Feature extraction by NMF

Suppose we are given with n labeled signals $(\mathbf{x}_i, y_i)$ for $i = 1, \dots, n$, where $\mathbf{x}_i \in \mathbb{R}^p$ is a p -dimensional signal and $y_i \in \{0, 1\}$ is its binary label. Suppose p is large (e.g., high-dimensional signals) and there is a small number R of latent feature vectors $\mathbf{w}_1, \dots, \mathbf{w}_R$, forming a 'dictionary' matrix $\mathbf{W} \in \mathbb{R}^{p \times R}$ such that each high-dimensional signal $\mathbf{x}_i$ can be approximated by some linear combination $\mathbf{W}\mathbf{h}_i$ for some $\mathbf{h}_i \in \mathbb{R}^R$. In this way, the high-dimensional signal $\mathbf{x}_i$ is compressed into a low-dimensional signal $\mathbf{h}_i$. The problem of finding suitable factor matrices $\mathbf{W}$ and $\mathbf{H} = [\mathbf{h}_1, \dots, \mathbf{h}_n]$ can be formulated by a *matrix factorization* problem $\mathbf{X} \approx \mathbf{W}\mathbf{H}$. That is, each column of the data matrix is approximated by the linear combination of the columns of the *dictionary matrix* $\mathbf{W}$ with coefficients given by the corresponding column of the *code matrix* $\mathbf{H}$ (see Figure 11). Variants of matrix factorization problems have been investigated under many names over the decades, each with different assumptions and constraints: dictionary learning, factor analysis, topic modeling, and component analysis. It has applications in text analysis, image reconstruction, medical imaging, bioinformatics, network dictionary learning, and many other scientific fields more generally^{30–36,44}.

Along with Principal Component Analysis (PCA)⁶⁵, non-negative Matrix Factorization (NMF)²⁷ is a classical matrix

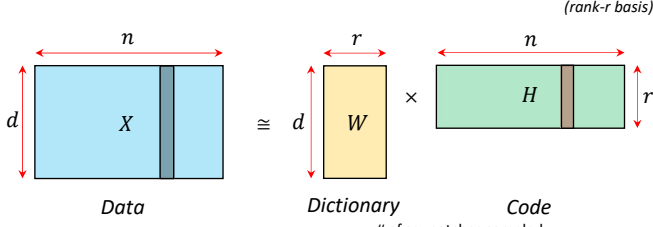


FIG. 14. Illustration of matrix factorization. Each column of the data matrix is approximated by the linear combination of the columns of the dictionary matrix with coefficients given by the corresponding column of the code matrix. Sample sq. patches $\rightarrow$ NMF

$$\inf_{\substack{\mathbf{W} \in \mathbb{R}^{d \times r}_{\geq 0}, \mathbf{H} \in \mathbb{R}^{r \times n}_{\geq 0}}} \|\mathbf{X} - \mathbf{WH}\|_F^2, \quad (C1)$$

where $\|\mathbf{A}\|_F = \sqrt{\sum_{i,j} A_{ij}^2}$ denotes the matrix Frobenius norm.

A consequence of the nonnegativity constraint on the code matrix is that one must represent the columns of the data matrix only using nonnegative linear combinations of the columns of the dictionary matrix $\mathbf{W}$. Since the columns of $\mathbf{W}$ are also constrained to be nonnegative, every feature captured in columns of $\mathbf{W}$ can only additively (rather than subtractively) be combined to explain the data points (columns in $\mathbf{X}$). This allows one to interpret the columns of $\mathbf{W}$ as ‘parts’ of the data and the columns of $\mathbf{H}$ as their ‘contribution’ in composing the columns of the data.

Appendix D: Background on Supervised Matrix Factorization (SMF)

Note that the NMF formulation in (C1) does not incorporate the labels $y_1, \dots, y_n$. This means the dictionary matrix $\mathbf{W}$ is for the best possible reconstruction of the data matrix $\mathbf{X}$, but the best reconstructive dictionary $\mathbf{W}$ may not be very effective for the classification tasks.

In the *supervised matrix factorization* (SMF) literature^{28,29}, one desires a dictionary that is reconstructive as well as *discriminative* in that such a compressed representation of signals is adapted to predicting the class labels y_i . In order to learn a dictionary matrix $\mathbf{W}$ that is both data-reconstructive and label-discriminative, we jointly model the pairs $(\mathbf{x}_i, y_i)$ of high-dimensional signal $\mathbf{x}_i$ and binary label y_i as

$$\mathbf{x}_i \approx \mathbf{W}\mathbf{h}_i \quad \text{and} \quad y_i | \mathbf{x}_i \sim \text{Bernoulli}(p_i), \quad (D1)$$

where $\mathbf{W} \in \mathbb{R}^{p \times R}_{\geq 0}$ is some unknown *nonnegative* dictionary matrix and the p_i is the predictive probability given by

$$p_i := \frac{\exp(\langle \beta, \mathbf{W}^T \mathbf{x}_i \rangle)}{1 + \exp(\langle \beta, \mathbf{W}^T \mathbf{x}_i \rangle)} \quad (D2)$$

for $i = 1, \dots, n$. Here we can interpret the product $\mathbf{W}^T \mathbf{x}_i$ as performing a convolution on the p -dimensional signal $\mathbf{x}_i$ using the R columns in $\mathbf{W}$. Since $\mathbf{W}$ is nonnegative, such convolution computes the proximity score of the pattern in each column of $\mathbf{W}$ observed in observed signal $\mathbf{x}_i$. In this sense, we view the columns of $\mathbf{W}$ as ‘filters’ that encode R particular patterns we seek to detect in $\mathbf{x}_i$. The vector $\mathbf{W}^T \mathbf{x}_i \in \mathbb{R}^R$ of proximity scores is then used as input to the logistic classifier with regression coefficients in β . If the j th coordinate of β is positive (resp., negative), then the proximity score $\langle \mathbf{w}_j, \mathbf{x}_i \rangle$ of the signal $\mathbf{x}_i$ with j th filter $\mathbf{w}_j$ is positively (resp., negatively) associated with y_i being one (resp., zero).

We formulate the following joint optimization problem for fitting *nonnegative* SMF model to the data-label pairs:

$$\min_{\mathbf{W} \geq 0, \mathbf{H} \geq 0, \beta} \left(\sum_{i=1}^n \ell(y_i, \langle \beta, \mathbf{W}^T \mathbf{x}_i \rangle) + \xi \|\mathbf{X} - \mathbf{WH}\|_F^2 \right), \quad (D3)$$

where $\mathbf{W} \in \mathbb{R}^{p \times R}_{\geq 0}$, $\mathbf{H} \in \mathbb{R}^{R \times n}_{\geq 0}$, $\beta \in \mathbb{R}^R$, and the negative log-likelihood function ℓ is defined by

$$\ell(y_i, \langle \beta, \mathbf{W}^T \mathbf{x}_i \rangle) := -y_i \log p_i - (1 - y_i) \log(1 - p_i) \quad (D4)$$

with p_i as in (D2). Here, the *tuning parameter* $\xi \geq 0$ controls the trade-off between the two objectives of classification and matrix factorization.

The objective function in (D3), say $F(\mathbf{W}, \mathbf{H}, \beta)$, is convex in each of the three variables $\mathbf{W}, \mathbf{H}$, and β while the other two variables are held fixed. Hence we can employ cyclic block coordinate descent (BCD) algorithms⁶². In particular, the simplest cyclic block minimization algorithm for solving (D3) reads as

$$\begin{cases} \mathbf{W}_{k+1} & \leftarrow \arg \min_{\mathbf{W} \geq 0} F(\mathbf{W}, \mathbf{H}_k, \beta_k) \\ \mathbf{H}_{k+1} & \leftarrow \arg \min_{\mathbf{H} \geq 0} F(\mathbf{W}_{k+1}, \mathbf{H}, \beta_k) \\ \beta_{k+1} & \leftarrow \arg \min_{\beta} F(\mathbf{W}_{k+1}, \mathbf{H}_{k+1}, \beta). \end{cases} \quad (D5)$$

Each sub-problem in (D5) is a convex optimization problem, which can be solved by using standard algorithms such as projected gradient descent⁴⁵. See Lee, Lyu, and Yao⁴⁹ for a more detailed implementation of BCD for nonnegative SMF (D3) and convergence guarantees.

Appendix E: Performance Comparison for Various Graph Embedding Methods

In this section, we provide additional experiments in order to demonstrate that the way we encode the topological features of the underlying graph does not make a significant difference in the prediction accuracy for the synchronization prediction of coupled oscillators on these graphs.

Graph embedding techniques like DeepWalk⁴², graph2vec⁴³, and Spectral Embedding⁶⁶ have been used extensively in recent years to embed a graph into a low-dimensional vector space while preserving the structure of the graph. In this paper, we have used a rather simple colored adjacency tensor (CAT)² to encode T -iterations of dynamics

on a k -node subgraph into a $k \times k \times T$ nonnegative tensor. As the name suggests, it is a stack of the adjacency matrix of the underlying graph, with additional information on the phase difference along the edges at each time. One may wonder if one uses a more sophisticated graph embedding algorithm to encode a graph-dynamics pair, then one would get a potentially significant performance gain in synchronization prediction problems. However, we argue that this is not the case.

Spectral embedding is a classical graph embedding technique that uses top eigenvectors of the graph Laplacian matrix as low-dimensional vector representations of the nodes of a network. The objective of the DeepWalk is to learn a mapping of nodes into a low dimensional Euclidean space such that two nodes that co-appear frequently in random walk sequences on the network would be assigned with their vector representations with the large inner product; two nodes that do not co-appear frequently will be nearly orthogonal after the embedding. The objective of the graph2vec is to learn low dimensional graph embeddings in an unsupervised manner, primarily used for graph classification.

In Table E, we perform synchronization prediction on $k = 20$ -node subgraphs from UCLA network using a logistic classifier on data sets generated by using various different methods to encode the graph topology. The data generation setting is identical to that for the subgraph-level experiments in Section VC, which we explained in detail in Appendix B. In this table, ‘dynamics’ means the $k \times T$ matrix encoding of the T -iterations of dynamics on the graph. We append this matrix with four different encodings of the underlying graph: Adjacency matrix, spectral embedding, DeepWalk, and graph2vec.

TABLE III. Effect on Logistic Regression Prediction Accuracy of Adding Graph Embedding Features in Addition to Dynamics.

Embedding Technique	Kuramoto	FCA	GHM
Dynamics	96.4%	92.7%	90.6%
Dynamics + Adjacency Matrix	96.3%	92.9%	91.4%
Dynamics + Spectral Embedding	96.9%	92.7%	90.9%
Dynamics + DeepWalk	96.5%	92.1%	91.3%
Dynamics + graph2vec	96.8%	93.1%	91.8%

We observe that encoding the underlying graph using various methods does not seem to provide the model with much additional information. The highest accuracy gain even for modern embedding methods compared to providing the model with only dynamics and the adjacency matrix is in the case of Kuramoto, with a 0.6% gain, which does not lead to a significant difference. Moreover, this trend remains relevant for all three coupled oscillator models. Overall, changing the graph embedding technique seems to have little to no effect on model performance, which is why a simple adjacency-matrix-based canonical representation that is encoded by our CATs (See Figure 2) is already sufficient to provide the model with enough graph topology information.

Appendix F: Extended subgraph-level prediction accuracies

In this section, we provide the full set of results of prediction accuracy of various methods versus LLDM on the subgraph level for k -node subgraphs where $k \in \{10, 15, 20, 25, 30\}$ with the FCA, Kuramoto, and GHM dynamics induced on them. Table F represents accuracies for subgraphs sampled from NWS, table V represents accuracies for subgraphs sampled from CALTECH, and table VI represents accuracies for subgraphs sampled from UCLA. Each accuracy metric reported is the mean accuracy from 5 seeds, based on a grid search across tuning parameter $\xi \in [0.1, 0.5, 1.0]$.

Appendix G: latent dynamics filters or Dictionary Plots for Various Settings

In this section, we show the rank 8 dictionary atoms learned by LLDM from the three parent networks and three dynamics networks for subgraph sizes $k \in \{10, 15, 25, 30\}$ with $k = 20$ in the main text Section VF. See Figures 12 and 13.

Appendix H: Goodness-of-fit Plots

In this section, we show the rank 2, 4, 8, and 16 goodness-of-fit plots in Figure 14 with deviance residuals for subgraph sizes $k \in \{10, 15, 25, 30\}$ with $k = 20$ in the main text Section VE.

TABLE IV. Prediction accuracy of various methods for FCA, Kuramoto, and GHM dynamics on subgraphs with k number of nodes where $k \in \{10, 15, 20, 25, 30\}$ sampled from a large-connected NWS parent graph. All accuracy values are an average of 5 seeds. The highest accuracy for each setting is indicated in **bold**. Whenever the average values of accuracy are equal for two methods, **both** are represented with **bold** font.

	FCA					Kuramoto					GHM				
	$k=10$	$k=15$	$k=20$	$k=25$	$k=30$	$k=10$	$k=15$	$k=20$	$k=25$	$k=30$	$k=10$	$k=15$	$k=20$	$k=25$	$k=30$
Baseline	80.2	78.2	69.6	71.0	69.4	56.5	63.2	66.1	69.9	64.3	89.6	78.7	76.2	69.6	65.6
LogReg	92.7	95.2	94.8	89.1	76.4	83.5	81.8	80.1	82.4	80.2	95.5	92.7	93.6	90.1	92.8
FFNN	92.2	94.9	95.4	90.2	82.8	84.2	82.5	83.4	83.8	79.2	94.4	95.8	91.8	91.4	94.1
LLDM-T (R=2)	82.7	89.4	81.3	85.0	75.9	74.3	76.6	76.0	77.1	80.1	85.5	90.2	89.0	88.0	89.7
LLDM-T (R=8)	86.5	91.3	81.9	86.0	75.7	75.2	77.5	77.1	77.4	80.3	81.6	90.4	81.7	78.1	83.0
LLDM (R=2)	91.5	92.2	93.2	87.0	77.2	81.4	76.1	77.5	77.2	78.3	95.9	93.3	91.0	89.4	89.5
LLDM (R=8)	93.0	92.8	94.2	88.1	84.8	82.1	81.6	78.4	84.4	78.4	96.0	96.2	92.3	89.0	94.1

TABLE V. Prediction accuracy of various methods for FCA, Kuramoto, and GHM dynamics on subgraphs with k number of nodes where $k \in \{10, 15, 20, 25, 30\}$ sampled from a large-connected CALTECH parent graph. All accuracy values are an average of 5 seeds. The highest accuracy for each setting is indicated in **bold**. Whenever the average values of accuracy are equal for two methods, **both** are represented with **bold** font.

	FCA					Kuramoto					GHM				
	$k=10$	$k=15$	$k=20$	$k=25$	$k=30$	$k=10$	$k=15$	$k=20$	$k=25$	$k=30$	$k=10$	$k=15$	$k=20$	$k=25$	$k=30$
Baseline	78.4	74.3	68.8	68.3	63.7	52.5	61.6	68.3	66.1	65.4	85.3	74.5	72.6	66.7	65.8
LogReg	91.8	92.2	90.5	91.5	89.4	82.6	80.2	81.2	85.6	81.7	96.5	93.2	92.9	88.3	91.5
FFNN	92.9	93.8	91.4	92.2	87.2	83.6	81.2	80.8	90.4	81.9	95.3	95.2	95.7	90.8	91.8
LLDM-T (R=2)	89.0	88.8	78.4	81.9	76.2	83.9	78.2	79.5	82.4	81.0	96.0	92.5	89.2	86.4	84.3
LLDM-T (R=8)	86.9	90.6	80.1	80.3	82.4	82.7	78.6	79.8	84.8	81.7	92.9	92.4	84.9	70.9	76.7
LLDM (R=2)	88.8	91.4	81.1	84.6	80.4	82.6	76.8	79.2	83.7	80.1	95.4	92.2	89.0	84.7	82.4
LLDM (R=8)	90.4	93.0	89.5	87.7	87.8	82.8	77.9	82.2	89.0	79.6	95.5	93.4	93.8	91.6	90.1

TABLE VI. Prediction accuracy of various methods for FCA, Kuramoto, and GHM dynamics on subgraphs with k number of nodes where $k \in \{10, 15, 20, 25, 30\}$ sampled from a large-connected UCLA parent graph. All accuracy values are an average of 5 seeds. The highest accuracy for each setting is indicated in **bold**. Whenever the average values of accuracy are equal for two methods, **both** are represented with **bold** font.

	FCA					Kuramoto					GHM				
	$k=10$	$k=15$	$k=20$	$k=25$	$k=30$	$k=10$	$k=15$	$k=20$	$k=25$	$k=30$	$k=10$	$k=15$	$k=20$	$k=25$	$k=30$
Baseline	80.6	77.4	70.2	71.2	68.3	56.4	65.3	69.2	68.4	64.9	86.5	76.3	71.4	67.2	65.2
LogReg	89.8	89.2	94.1	93.8	91.2	91.3	90.2	91.4	92.8	94.6	94.2	92.4	95.6	92.3	92.3
FFNN	90.4	91.1	95.6	93.2	92.5	92.8	91.3	92.4	93.6	95.2	93.8	92.2	94.8	93.4	92.9
LLDM-T (R=2)	77.9	79.7	80.4	82.3	80.1	79.8	84.1	87.1	83.2	87.2	92.3	92.2	89.4	92.5	91.4
LLDM-T (R=8)	78.3	80.9	84.7	87.3	84.1	82.6	85.8	87.6	86.5	90.2	94.6	91.1	87.9	89.5	88.4
LLDM (R=2)	86.8	86.5	90.5	90.8	86.4	79.1	86.2	85.9	85.2	93.5	93.2	91.6	90.7	91.1	91.2
LLDM (R=8)	91.1	88.6	92.6	92.7	90.0	89.3	89.3	90.0	94.2	94.7	94.6	91.6	93.3	93.1	95.9

NTWK Dynamics		# Nodes	10 Nodes								15 Nodes							
NWS	FCA		1.514	1.487	0.792	-0.491	-0.719	-1.164	-1.609	-1.66	0.558	0.481	0.257	0.068	0.001	-0.196	-0.792	-1.363
	Kuramoto		0.024	-0.111	-0.384	-0.47	-0.486	-0.507	-0.571	-0.605	0.031	-0.033	-0.22	-0.24	-0.27	-0.273	-0.397	-0.632
	GHM		0.217	-0.265	-0.294	-0.306	-0.501	-0.593	-0.617	-0.799	0.043	-0.143	-0.242	-0.426	-0.538	-0.635	-0.67	-0.799
Caltech	FCA		1.55	0.498	0.443	0.404	-0.399	-0.559	-0.819	-1.602	0.825	0.801	0.371	-0.19	-0.371	-0.637	-0.697	-1.043
	Kuramoto		0.218	-0.291	-0.345	-0.359	-0.45	-0.452	-0.466	-0.626	1.136	0.354	-0.112	-0.189	-0.308	-0.319	-0.691	-1.232
	GHM		0.194	-0.202	-0.324	-0.401	-0.584	-0.628	-0.683	-0.685	0.065	-0.171	0.443	0.404	-0.399	-0.559	-0.819	-1.602
UCLA	FCA		2.081	0.664	0.55	0.496	-0.691	-0.79	-1.875	-2.333	0.908	0.415	0.203	0.05	-0.078	-0.285	-0.727	-1.335
	Kuramoto		0.128	-0.119	-0.124	-0.331	-0.477	-0.507	-0.605	-0.625	1.531	1.239	1.129	0.492	-0.842	-0.873	-1.167	-1.874
	GHM		0.098	-0.146	-0.329	-0.441	-0.528	-0.656	-0.831	-1.119	0.02	-0.043	-0.055	-0.393	-0.58	-0.824	-0.865	-0.889

FIG. 12. The 8-element SMF latent dynamics filters of FCA and Kuramoto on Caltech, UCLA, NWS networks of 10 nodes and 15 nodes

NTWK		# Nodes	25 Nodes								30 Nodes							
		Dynamics																
NWS		FCA	0.209	0.208	0.085	0.031	0.008	0.007	-0.15	-0.41	0.296	0.274	-0.024	-0.084	-0.093	-0.106	-0.135	-0.196
		Kuramoto	0.29	0.249	0.157	-0.172	-0.219	-0.28	-0.337	-0.407	0.51	0.246	0.148	0.03	0.007	-0.076	-0.489	-0.606
		GHM	0.027	-0.038	-0.07	-0.496	-0.53	-0.713	-0.941	-0.968	0.515	0.248	0.082	0.047	-0.443	-0.545	-0.623	-1.482
Caltech		FCA	0.38	0.211	0.097	0.064	-0.028	-0.14	-0.256	-0.464	0.522	0.364	0.166	0.074	0.047	-0.234	-0.458	-0.712
		Kuramoto	1.011	0.452	-0.099	-0.207	-0.248	-0.313	-0.321	-0.437	0.823	0.262	0.256	-0.188	-0.214	-0.277	-0.291	-0.536
		GHM	0.505	0.479	0.081	-0.003	-0.061	-0.229	-1.017	-1.371	0.176	0.074	0.049	-0.105	-0.277	-0.525	-0.757	-0.884
UCLA		FCA	0.299	0.267	0.176	0.082	0.059	-0.155	-0.173	-1.113	0.326	0.273	0.206	-0.22	-0.235	-0.279	-0.286	-0.287
		Kuramoto	1.737	0.416	-0.148	-0.3	-0.37	-0.4	-0.402	-0.469	0.494	0.466	0.388	0.316	-0.241	-0.549	-0.607	-0.612
		GHM	0.774	0.673	-0.047	-0.21	-0.647	-0.74	-0.808	-0.923	1.015	0.979	0.738	0.247	-0.535	-0.815	-0.888	-1.289

FIG. 13. The 8-element SMF latent dynamics filters of FCA and Kuramoto on CALTECH, UCLA, NWS networks of 25 and 30 nodes

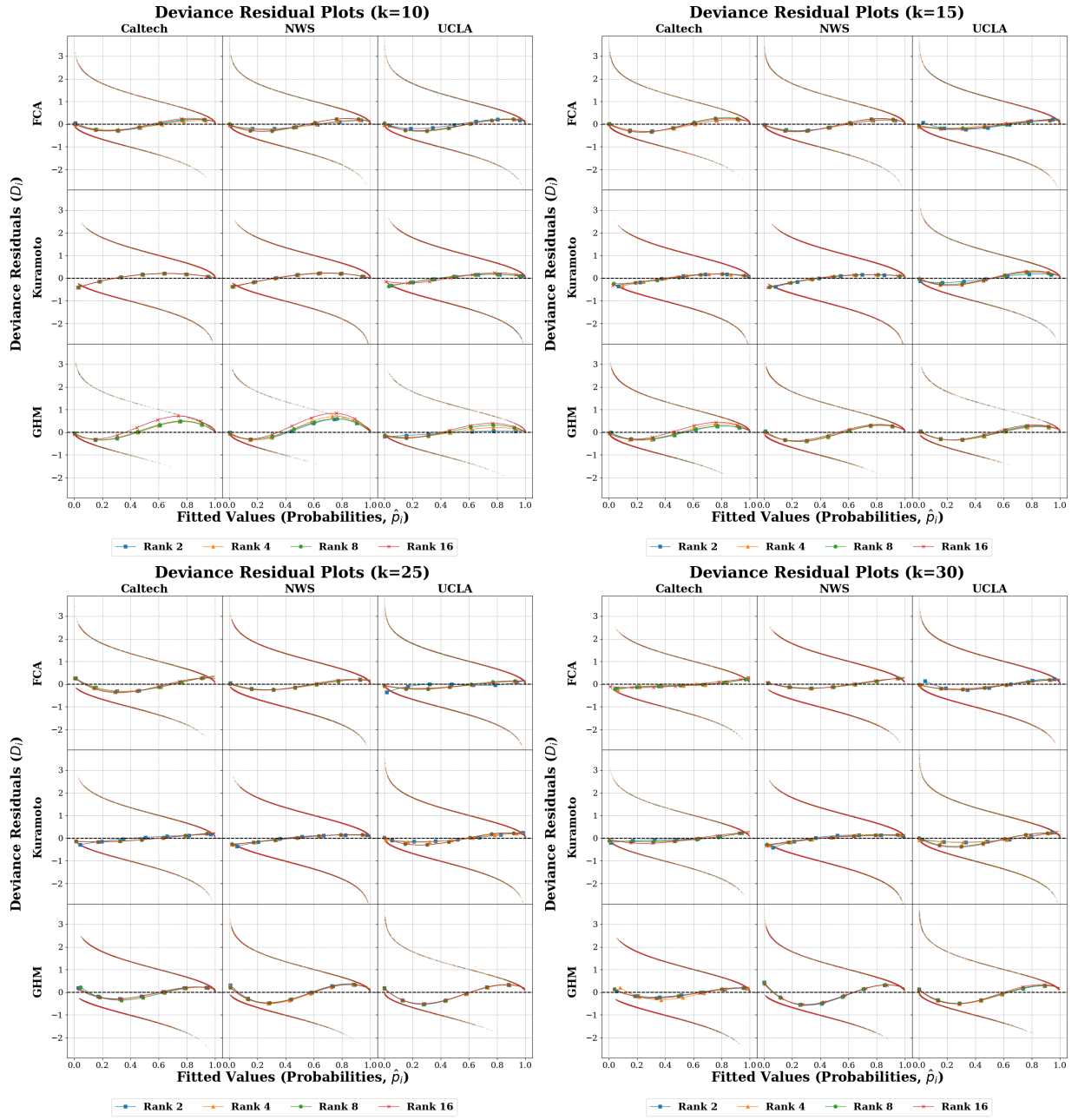


FIG. 14. Goodness-of-fit with deviance residuals for $k = 10, 15, 25$ and 30 on rank $2, 4, 8$ and 16 for CALTECH, UCLA, and NWS graphs with FCA, Kuramoto and GHM models