Sparse Network Optimization for Synchronization

Regina S. Burachik · Alexander C. Kalloniatis · C. Yalçın Kaya

Received: 22 June 2020 / Accepted: 19 August 2021 / Published online: 9 September 2021
© Crown 2021

Abstract
We propose new mathematical optimization models for generating sparse dynamical graphs, or networks, that can achieve synchronization. The synchronization phenomenon is studied using the Kuramoto model, defined in terms of the adjacency matrix of the graph and the coupling strength of the network, modelling the so-called coupled oscillators. Besides sparsity, we aim to obtain graphs which have good connectivity properties, resulting in small coupling strength for synchronization. We formulate three mathematical optimization models for this purpose. Our first model is a mixed integer optimization problem, subject to ODE constraints, reminiscent of an optimal control problem. As expected, this problem is computationally very challenging, if not impossible, to solve, not only because it involves binary variables but also some of its variables are functions. The second model is a continuous relaxation of the first one, and the third is a discretization of the second, which is computationally tractable by employing standard optimization software. We design dynamical graphs that synchronize, by solving the relaxed problem and applying a practical algorithm for various graph sizes, with randomly generated intrinsic natural frequencies and initial phase variables. We test robustness of these graphs by carrying out numerical simulations with random data and constructing the expected value of the network’s order parameter and its variance under this random data, as a guide for assessment.

Keywords Optimization · Sparse graphs · Sparse networks · Synchronization · Kuramoto model · Optimal control · Discretization
1 Introduction

Spontaneous synchronization happens frequently in nature: common examples are fireflies flashing, crickets chirping, planets orbiting and neurons firing. All these phenomena consist of a set of agents that exhibit a cyclic behaviour: these agents are called oscillators [15]. Two or more oscillators are said to be coupled if they influence each other by some physical or chemical process. In our setting, each oscillator is associated with the node of a graph, and the whole graph represents a network. The result of such interactions is often synchrony, which means that all entities end up with the same frequency after a short period of time. The Kuramoto model [25] was originally motivated whereby a system of coupled oscillating nodes for sufficient coupling lock on to a common frequency, in this case the mean of the ensemble of natural frequencies of the oscillators, despite the variance in those frequencies. For a recent literature review of the model see [34].

In the following, we summarize salient aspects of the literature relevant to network optimization. The original model, as in [25], was posed for all-to-all coupling in a complete graph. Since then the pattern of synchronization has been explored for a variety of regular and irregular graphs. At one end, sparse regular graphs are the poorest in synchronization, by which we mean they require high coupling strength to achieve synchronization. For example, synchronization on regular structures such as trees [10] and rings [31,35] admits a degree of solubility for critical coupling, or bounds thereon. These are in some respects balanced structures: sparse, with uniform degree distributions. However, as they scale in size, they require increasingly higher values of coupling to achieve synchrony. Irregular or complex graphs, such as Erdös–Rényi [19], small world [17], and the scale-free Barabasi–Alberts [32] networks synchronize for significantly smaller coupling, with small world generally the worst of these [9]. However, their complexity leads to very uneven distribution of degrees which may incur some cost, for example imbalances in work or communication loads depending on the physical setting of the synchronization process. In our case, our interest in such graphs is for the purpose of synchronized decision-making in organizations [20] where span of control and cognitive overload require sparseness and balanced degree distribution.

In terms of approaches that explicitly use a form of optimization, [4] employed a stochastic hill-climbing approach based on random network link rewirings and acceptance or rejection on the criterion of improving the Kuramoto order parameter \( r \), to be defined below. Contemporaneously, [36] applied gradient descent to the order parameter to be able to more efficiently compute optimal weighted networks. The authors of [24] follow an evolution process similar to that of [4], but, to avoid hubs in the network, the hill-climbing trajectory only accepts a rewiring that preserves the initial degree distribution. In the same work, the authors propose the use of an additional energy-like function in the selection process.
A Markov Monte Carlo replica exchange approach was developed by [40] to optimize network synchronization against noise, but the method becomes computationally demanding beyond $N = 15$ size graphs. In a second-order generalization of the Kuramoto model, relevant to electrical power systems, [14] used convex optimization to determine optimal placements of frequencies or link weights within an existing network structure. The examples considered in that work were of similarly small size.

A number of optimization approaches have been applied to a related, but simpler, coupled oscillator model in which the graph Laplacian figures explicitly in the interaction function, and to which the Kuramoto model approximates under linearization. Using the master stability function of [33], the authors of [3] observed that graphs with lower Laplacian spectral ratio exhibited improved synchronization (see also [27] for stability of twisted states). In this spirit, a stochastic rewiring approach based on improving the spectral ratio was explored by [11,12] yielding heterogeneous and typically dense graphs. Complementary to the spectral ratio approach is that of expander graphs [18] which have the property that every partition of the nodes into two subsets has a number of boundary edges between them that scales with the size of the smallest subset. While quite powerful analytical results linking stability and expansion properties can be established for coupled dynamical systems with ‘diffusive coupling’, such as [28], for Kuramoto type interactions things are more challenging. Here [13] conducted computational search over instances of graphs with such properties, with results that have excellent expansion and synchronization properties but that increase in degree as the graphs are scaled in size.

To conclude this summary, we mention that [37] have provided a construction for the Kuramoto model that yields modified trees—which are therefore sparse and well-balanced—that are provably expanders, with enhanced synchronization. In this respect, the starting point (or template) of the tree hierarchy was key. They also showed that a computational search over random variations of the links on the leaf nodes of this construction (maintaining the degree distribution) to find instances with lowest possible spectral ratio did not necessarily give graphs of better synchronization.

For more details on similar techniques, which progressively search for a network so as to promote its fitness for synchronization, see [34, Section 8].

In most of these approaches, however, optimization takes the form of a hill-climbing evolution, where, according to certain rules and merit functions, a sequence of “mutations” of an initial network, is constructed. After a prescribed number of steps (usually stipulated a priori), the last mutation accepted will exhibit some “optimal” properties, in the sense that the merit functions used in the process have the best values among all the previous mutations.

In the present paper, we propose, implement and solve a new optimization model for designing graphs that (i) are sparse, (ii) can achieve synchronization in the context of the Kuramoto model [34] and (iii) have a relatively narrow range of small degrees. Conditions (i) and (iii) are helpful for ease of implementation of a network, as well as for promoting balanced sharing of communication- or work-loads.

We mathematically incorporate these three features into our optimization model. Indeed, our optimization variables are the set of all weighted (connected) undirected networks (expressed as $N \times N$ symmetric matrices with nonnegative coefficients), and a set of $N$ functions, namely the phases $\{\theta_i\}$ for $i = 1, \ldots, N$ in the Kuramoto
system. Our objective function involves a penalty term ensuring sparsity (given by the \( \ell_1 \) norm of the matrix) and two other terms promoting the small variance of the degrees (see the second and third terms in the objective function in (6)).

To the authors' knowledge, there is no available approach that proposes a mathematical optimization model that performs a "universal search" among all undirected networks which are able to synchronize by means of the Kuramoto model.

The initial version of the optimization model we construct is an infinite-dimensional mixed-integer programming problem, which is in the form of an optimal control problem. By relaxing the integer variables, we obtain a continuous-variable model, which is nonsmooth and still infinite-dimensional, and therefore not tractable yet. We then discretize the problem, so that optimization software can be employed, as it is done for optimal control problems. The discretization produces a large-scale problem, which, in general, prohibits any use of nonsmooth solvers. However, in our particular model, non-smoothness appears in a peculiar form that can be transformed into a smooth, i.e. differentiable, form. We use tricks and techniques from optimization to convert the problem into a smooth one so that powerful differentiable solvers can be employed to get a solution, even though the problem is still a large-scale and non-convex one.

One should note that a solution to our discrete and smooth optimization model does not represent a graph immediately. A key procedure we propose, Algorithm 1, uses solutions of our discrete and smooth optimization model so as to construct increasingly sparse graphs while keeping the spectral ratio of the Laplacian relatively small. In other words, Algorithm 1 helps us

(i) Find a sparse adjacency matrix of the graph such that the system defined by the equations in (1) below is synchronized, and
(ii) Ensure that the resulting graph exhibits a small spectral ratio (defined in (5) below) and
(iii) Has a possibly small and narrow range of node degrees.

We present extensive numerical experiments by using this algorithm for graphs with \( N = 20 \) and \( N = 127 \). We illustrate the fact that when adding a moderate number of edges, a great improvement in synchronization properties of the graph can be achieved. We also make comparisons with certain graphs from the literature, in particular from [37], where graphs are obtained by appending at the bottom level of a hierarchical tree, additional matchings linking the leaf nodes. The graphs found in this way in [37] enhance synchronization compared to the original tree. We emphasize the key result: that we can generate graphs with optimal sparseness, balanced load, and good synchronization properties starting from a random frequency and initial phase input. Though expander approaches are extremely powerful when certain regularities are strictly imposed on the graph, or linearization approximations imposed, the optimization approach we present here allows for trade-offs to be explored between regularity, sparsity and enhancement of synchronization independent of strong coupling assumptions. Because the graphs we generate show low degree heterogeneity, the resultant synchronization is not so sensitive to the input frequency choices.

The paper is organized as follows. In Sect. 2, we recall classical concepts and properties of the Kuramoto model and the spectral ratio of a graph. In Section 3, we introduce our mathematical model that promotes synchronization as well as pose its
relaxation. In Section 4, we discuss discretization of the time-dependent functions in the model and smoothing. In Sect. 5, we present our numerical experiments and provide a detailed interpretation of the numerical results. In Sect. 6, we provide concluding remarks and comment on future work.

2 Preliminaries

In this section we briefly recall the mathematical concepts we will use in our models, including the tree-expander construction approach presented in [37].

2.1 The Kuramoto Model

Let $A \in \{0, 1\}^{N \times N}$ (i.e. $A$ is an $N \times N$ matrix with all its entries equal to 0 or 1) and $\omega = (\omega_1, \ldots, \omega_N) \in \mathbb{R}^N$. The Kuramoto model over an $N$-node network can be written as a system of ODEs:

$$\dot{\theta}_i(t) = \omega_i - \sigma \sum_{j=1}^{N} A_{ij} \sin(\theta_i(t) - \theta_j(t)), \quad \theta_i(0) = \theta_{i,0}, \quad i = 1, \ldots, N,$$

where $\theta_i : [0, T] \to \mathbb{R}$ represents the phase of the $i$th node, with $\dot{\theta}_i := d\theta_i/dt$, and $\theta(t) := (\theta_1(t), \ldots, \theta_N(t))$ is the phase vector in $\mathbb{R}^N$. Moreover, $\omega$ is the vector of intrinsic natural frequencies, $\sigma$ is the coupling strength of the network, $\theta_0 := (\theta_{i,1}, \ldots, \theta_{i,N})$ is the initial phase vector, and $A$ is the adjacency matrix of a graph with $N$ nodes.

It is important to distinguish two types of synchronization. Phase synchronization refers to asymptotic equality of the phases, namely the case when $(\dot{\theta}_i(t) - \dot{\theta}_j(t)) \to 0$ as $t \to \infty$, $\forall i, j \in \{1, \ldots, N\}$. This cannot hold for non-identical oscillators, where $\omega_i \neq \omega_j$. Frequency synchronization, on the other hand, refers to asymptotic equality of the frequencies, namely the case when $(\dot{\theta}_i(t) - \dot{\theta}_j(t)) \to 0$ as $t \to \infty$, $\forall i, j \in \{1, \ldots, N\}$.

Except in certain regular graphs, when coupling is sufficient for frequency synchronization to set in, the corresponding phases tend to be clustered close to each other, namely that $(\theta_i(t) - \theta_j(t)) \leq \varepsilon$ for all $t > t_\ell$ and all $i, j \in \{1, \ldots, N\}$, with some $\varepsilon > 0$ and sufficiently large $t_\ell > 0$, which is referred to as (full) phase locking. The higher the coupling is, the smaller the discrepancies between the oscillator phases will be. In the computational setting that we have, we consider the finite time horizon $[0, T]$ with $T \geq t_\ell$. This synchronization, however, usually requires dense matrices $A$. Viewing links as costly (in resource, in communications, in load), we seek to keep the topology of the graph as simple as possible. Therefore, it is relevant to find sparse matrices $A$ for which synchronization is achieved.
2.2 The Order Parameter

The so-called order parameter $r(t)$ is a function of time $t$ defined as:

$$ r(t) = \frac{1}{N} \left| \sum_{j=1}^{N} e^{i \theta_j(t)} \right|, $$

where $| \cdot |$ denotes the magnitude of a complex number, with $i = \sqrt{-1}$. It is not difficult to derive, from (2), that

$$ r^2(t) = \frac{1}{N^2} \left[ N + 2 \sum_{i<j}^{N} \cos(\theta_i(t) - \theta_j(t)) \right], $$

where the usage of $i$ as an index this time should be clear from the context. As a measure of synchronicity, usually the following long-term average $\bar{r}$ of $r$ is used:

$$ \bar{r}(T) = \frac{1}{T-T_{th}} \int_{T_{th}}^{T} r(t) \, dt, $$

where $T > 0$ is a fixed large number and $T_{th}$ is some fraction of the time-series describing the transient behaviour.

2.3 Spectral Ratio

Given a graph $G$ with $N$ nodes, call $d_i$ the degree of node $i$. Collecting all these coordinates in a single vector $d = (d_1, \ldots, d_N) \in \mathbb{N}^N$ produces the degree vector $d$ of $G$. If $A$ is the adjacency matrix of $G$, then the Laplacian $L$ of $G$ is defined as:

$$ L := \text{diag}(d) - A, $$

where $\text{diag}(d)$ is the diagonal matrix with $d_i$ its $i$th diagonal element. Let $\lambda_2$ and $\lambda_{\text{max}}$ be the second smallest and the largest eigenvalues of $L$, respectively. Recall that $\lambda_2 \geq 0$. Moreover, if the graph is connected, then $\lambda_2 > 0$. For a connected graph, we denote by

$$ Q_L := \frac{\lambda_{\text{max}}}{\lambda_2}, $$

the spectral ratio of the graph $G$. As alluded in the introduction, for a somewhat different class of dynamical systems, which the Kuramoto model approximates when linearized, graphs with small $Q_L$ seem to result in highly synchronized graphs, in that the graph is connected and $\sigma$ is small [3]. For the Kuramoto model, such an effect has been observed in [37].
2.4 Tree-Expander Construction

In [37], Taylor, Kalloniatis and Hoek propose a construction procedure for graphs which appends at the bottom level of a hierarchical tree additional matchings linking the leaf nodes. Recall that if the number of levels in a hierarchical tree is $m$, then the total number of nodes in the tree is $p = 2^m - 1$. Following the procedure given in [37], we have constructed a graph for $m = 7$, i.e. $p = 127$. We show in Section 5.2.2 a 158-edge graph we have obtained as well as a 158-edge that was obtained in [37], and report their $Q_L$ values. Intuitively, an expander graph is a graph in which every subset of the vertices that is not “too large” has a “large” boundary. Intuitively, this property favours connectivity. The authors show in [37] that the graph they construct is an expander graph. The resulting graphs have significantly improved synchronization properties, and are sparse, of nearly uniform degree distribution and small $Q_L$ (although, for other random leaf node matchings of the tree, the expander construction does not give the smallest possible $Q_L$). They therefore provide a reasonable benchmark to compare the results of our optimization. In Sect. 5.2.2, we compare the properties of these expander graphs with those obtained through our optimization approach described in Algorithm 1.

We proceed now to briefly explain the construction, following closely the explanation provided in section 5 of [37]. Recall that for a network with an even number of nodes, a matching is a set of edges that include all the nodes of the network, and where no two edges have a common node. Thus, matchings correspond to a non-overlapping pairing of the node set. The idea for the construction in [37] is to add such a matching at the bottom of a hierarchical tree. (Note that the number of nodes at a lower layer of a hierarchical tree is a power of 2, and hence an even number.) The construction has two steps. In the first step one constructs a matching for the nodes at the lower layer of the tree. Recall that a hierarchical tree is a graph of $2^m - 1$ nodes, with exactly $m$ layers (1 node in layer 1, 2 nodes in layer 2, $2^2 = 4$ nodes in layer 3,..., $2^{m-1}$ nodes in the last layer, layer $m$). So the first step of the construction is to produce a matching for a graph of $2^{m-1}$ nodes. Once this graph is produced, then in Step 2, the graph produced in Step 1 is used as the lower layer of the hierarchical tree. Note that the hierarchical tree already has the “upward-pointing” edges at the lower layer. In Step 2, we need to add the “horizontal edges” connecting the nodes at the bottom. This is done by adding the matchings between the $2^{m-1}$ nodes obtained in Step 1. For more details of this construction, the reader is referred to [37, Section 5].

3 Optimization Models

As mentioned in Sect. 1, our aims are (i) to find a sparse $A$, and to achieve (ii) synchronization and (iii) a narrow distribution of degrees. These aims, when realized, are expected to yield a relatively small $Q_L$. To obtain such a matrix $A$, we propose optimization models with decision variables $(A, \theta)$, where $A \in \mathbb{R}^{N \times N}$ and $\theta$ is the vector function defined in (1). When the graph is assumed to be undirected, the matrix $A$ is symmetric. Similar models can be posed for the directed case. We fix a random vector $\omega$, and a time $t_c$ after which we impose synchronization among $\theta_i$’s.
3.1 A Mixed Integer Optimization Problem

The coefficients of the adjacency matrix are binary, i.e. $A_{ij} \in \{0, 1\}$. Let $d$ be the vector of degrees as defined in Section 2.3. Also define
\[ d_{\min} := \min_{1 \leq i \leq N} d_i \quad \text{and} \quad d_{\max} := \max_{1 \leq i \leq N} d_i. \]

Fix positive parameters $c_1, c_2, c_3, t_c$, as well as a small $\varepsilon > 0$. We propose the following optimization model for the problem that we have.

\[
\begin{align*}
\text{objective :} & \quad \min_{A, \theta} c_1 \sum_{i,j} |A_{ij}| + c_2 \frac{d_{\max}}{d_{\min}} + c_3 \left( \frac{d_{\max}}{d_{\min}} - 1 \right) \\
\text{subject to the following constraints.} & \\
\text{integrality :} & \quad A_{ij} \in \{0, 1\} \quad \forall i, j, \\
\text{symmetricity :} & \quad A_{ij} = A_{ji} \quad \forall i, j, \\
\text{oscillator dynamics :} & \quad \dot{\theta}_i(t) = \omega_i + \sigma \sum_{j=1}^{N} A_{ij} \sin(\theta_j(t) - \theta_i(t)), \quad \forall t \in [0, T], \quad \theta_i(0) = \theta_{i,0}, \quad \forall i, \\
\text{synchrony :} & \quad \max_{t \geq t_c} \left\{ 0, \left( \sum_{ij} |\theta_i(t) - \theta_j(t)| \right) - \varepsilon \right\} = 0, \quad \forall i, j.
\end{align*}
\]

The following comments and observations can be made about this model.

- The first term in the objective function is the $\ell_1$-norm of the elements of the matrix $A$, which, when minimized, is well known to promote sparsity of $A$. The two subsequent terms represent the overall discrepancy between the degrees of the nodes. It turns out that addition of these terms accelerates computation by 10 to 100 times.
- We have tried specific software for mixed integer optimization, but due to the large number of variables, the above model has so far been intractable.
- We use a given value of $\sigma$ in the ODE constraints. It is possible to consider $\sigma$ as another decision variable, if smaller values of $\sigma$ are desired.

3.2 A Model with Relaxation

Due to the computational complexity, or intractability, of the model in (6), we can consider a relaxation of the model in the following way. First, combine the coefficients.

[Note: The rest of the text is not visible in the image provided.]
of the matrix and the coupling strength $\sigma$ into a single aggregated variable

$$\tilde{A}_{ij} := \sigma A_{ij},$$

(7)

and allow $\tilde{A}_{ij}$ to take any nonnegative real value, i.e. $\tilde{A}_{ij} \geq 0$. Next, we also relax the (integer-valued) degree vector by means of the generalized degree vector $\tilde{d} = (\tilde{d}_1, \ldots, \tilde{d}_N) \in \mathbb{R}^N$ such that

$$\tilde{d}_i := \sum_{j=1}^N \tilde{A}_{ij},$$

(8)

for $i = 1, \ldots, N$. Subsequently, the maximum and minimum degree variables we used in the previous model are modified accordingly as

$$\tilde{d}_{\min} := \min_{1 \leq i \leq N} \tilde{d}_i \quad \text{and} \quad \tilde{d}_{\max} := \max_{1 \leq i \leq N} \tilde{d}_i.$$  

(9)

The relaxation (7) allows to consider the ODE (1) with no $\sigma$ appearing on the right-hand side, and, more importantly, to replace the integrality constraints used in the $\ell_1$-model in (6) by non-negativity constraints. The use of these aggregated variables allows us to propose a model which can be more easily implemented. So, fix positive parameters $c_1, c_2$ and $c_3$, as before, and consider the following optimization problem:

\[
\begin{cases}
\text{objective :} & \min_{\tilde{A}, \theta} \ c_1 \sum_{i,j} |\tilde{A}_{ij}| + c_2 \ \tilde{d}_{\max} + c_3 \left( \frac{\tilde{d}_{\max}}{\tilde{d}_{\min}} - 1 \right) \\
\text{subject to the constraints.} & \\
\text{symmetry :} & \tilde{A}_{ij} = A_{ji} \ \forall i, j, \\
\text{nonnegativity :} & \tilde{A}_{ij} \geq 0, \ \forall i, j, \\
\text{oscillator dynamics :} & \dot{\theta}_i(t) = \omega_i + \sum_{j=1}^N \tilde{A}_{ij} \sin(\theta_j(t) - \theta_i(t)), \ \forall t \in [0, T], \ \theta_i(0) = \theta_{i,0}, \ \forall i, \\
\text{synchrony :} & \max_{t \geq t_0} \left\{ 0, \left( \sum_{ij} |\theta_i(t) - \theta_j(t)| \right) - \varepsilon \right\} = 0, \ \forall i, j.
\end{cases}
\]

(10)

Since every term in (8) is multiplied by $\sigma$, we have that

$$\frac{\tilde{d}_{\max}}{\tilde{d}_{\min}} = \frac{d_{\max}}{d_{\min}}.$$
Therefore, the terms involving \((d_{\text{max}}/d_{\text{min}})\) in the model (6) remain unchanged by the relaxation, in that \((\tilde{d}_{\text{max}}/\tilde{d}_{\text{min}})\) still truly means the ratio of the maximum and minimum degrees in the graph.

4 Discretization and Smoothing

As it stands in Sect. 3.2, the \(\ell_1\)-model is nondifferentiable. On the other hand, it is well known that a smooth re-formulation of the \(\ell_1\)-model can be obtained by using standard nonlinear programming techniques—see, for example, [30,38]. This allows us to use smooth optimization solvers for solving the discretized \(\ell_1\)-model.

With the elements \(\tilde{A}_{ij}\)s of the matrix \(\tilde{A}\) interpreted as constant control functions and \(\theta_i\)s interpreted as the state variables, the model in (10) is an optimal control problem. We have discretized the \(\ell_1\)-model, using the trapezoidal rule, in a fashion similar to the way optimal control problems are discretized; see, for example, [21,22]. Challenging optimal control problems have successfully been solved using direct discretization previously [1,2,6,23].

Suppose that the ODE in Model (10) is defined over the time horizon \([0, T]\), with \(T = t_c\). Let

\[
f(\theta_1(t), \ldots, \theta_N(t), \tilde{A}) := \omega_i + \sum_{j=1}^{N} \tilde{A}_{ij} \sin(\theta_j(t) - \theta_i(t)), \quad \forall t \in [0, T].
\]

Consider a regular partition of \([0, T]\), such that \(0 = t_0 < t_1 < \ldots < t_M = T\), with \(t_{k+1} := t_k + h\) and \(h := T/M\). Let \(\theta_i^k\) be an approximation of \(\theta_i(t_k)\), \(k = 0, 1, \ldots, M - 1\). Then, the ODE can be discretized by applying the trapezoidal rule:

\[
\theta_i^{k+1} = \theta_i^k + \frac{h}{2} \left( f(\theta_1^k, \ldots, \theta_N^k, \tilde{A}) + f(\theta_1^{k+1}, \ldots, \theta_N^{k+1}, \tilde{A}) \right),
\]

\(k = 0, 1, \ldots, M - 1\). After this discretization, and the replacement of the ODE with this discretization, the optimization model in (10) becomes a finite-dimensional optimization problem. With the number of optimization variables being \(((N - 1)NM/2)\), the discretized problem is large scale even with a moderate graph size \(N\) and a moderate partition size \(M\). Moreover, the objective function of the problem is nondifferentiable, prohibiting efficient use of smooth optimization methods and software.

Techniques from the optimization literature can be employed to transform the non-smooth objective function into a smooth one as follows. Nonsmoothness of the first term (the \(\ell_1\)-norm) is caused by the moduli, which can be avoided by defining two new variables, \(B_{ij}^1\) and \(B_{ij}^2\) such that (see, for example, [38])

\[
\tilde{A}_{ij} := B_{ij}^1 - B_{ij}^2, \quad B_{ij}^1, B_{ij}^2 \geq 0.
\]

Then, one can show that

\[
|\tilde{A}_{ij}| = B_{ij}^1 + B_{ij}^2.
\]
The RHS of the ODEs in (11) then becomes

\[
f(\theta_1(t), \ldots, \theta_N(t), B_{1ij}, B_{2ij}) := \omega_i + \sum_{j=1}^{N} (B_{1ij} - B_{2ij}) \sin(\theta_j(t) - \theta_i(t)),
\]

\[\forall t \in [0, T]. \quad (15)\]

To tackle the second and third terms in the objective function, one can define the new optimization variables (see, for example, [30])

\[
\alpha := \tilde{d}_{\text{max}} \quad \text{and} \quad \beta := \tilde{d}_{\text{min}}.
\]

Then,

\[
\min \frac{\tilde{d}_{\text{max}}}{\tilde{d}_{\text{min}}} \equiv \min \frac{\alpha}{\beta}
\]

with

\[
\sum_{j=1}^{N} \tilde{A}_{ij} \leq \alpha, \quad \text{for} \quad i = 1, \ldots, N, \quad (16)
\]

\[
\sum_{j=1}^{N} \tilde{A}_{ij} \geq \beta, \quad \text{for} \quad i = 1, \ldots, N. \quad (17)
\]

The synchrony constraint in Model (10) is also nonsmooth, which can be replaced by a smooth version given by

\[
\sum_{i \neq j} \left( \theta_i^M - \theta_j^M \right)^2 \leq \varepsilon. \quad (18)
\]

Incorporation of (11)–(18) transforms Model (10) into the smooth optimization problem below.

\[
\begin{aligned}
\text{objective :} & \quad \min_{B_1, B_2, \theta, \alpha, \beta} c_1 \sum_{i,j}^N (B_{1ij} + B_{2ij}) + c_2 \frac{\alpha}{\beta} + c_3 \left( \frac{\alpha}{\beta} - 1 \right) \\
\text{subject to the following constraints.} \\
\text{symmetry :} & \quad B_{1ij} - B_{2ij} = B_{1ji} - B_{2ji}, \quad \forall i, j, \\
\text{nonnegativity :} & \quad B_{1ij} - B_{2ij} \geq 0, \quad \forall i, j, \\
\text{discretization :} & \quad \theta_i^{k+1} = \theta_i^k + \frac{\epsilon}{2} \left( f(\theta_1^k, \ldots, \theta_N^k, B_{1ij}^k, B_{2ij}^k) + f(\theta_1^{k+1}, \ldots, \theta_N^{k+1}, B_{1ij}^k, B_{2ij}^k) \right), \\
& \quad k = 0, 1, \ldots, M - 1, \quad \theta_i^0 = \theta_{i0}, \quad \forall i, \\
\text{synchrony :} & \quad \sum_{i \neq j} \left( \theta_i^M - \theta_j^M \right)^2 \leq \varepsilon, \\
\text{auxiliary :} & \quad B_{1ij}, B_{2ij} \geq 0, \quad \forall i, j, \\
& \quad \sum_{j=1}^{N} (B_{1ij} - B_{2ij}) \leq \alpha \quad \text{and} \quad \sum_{j=1}^{N} (B_{1ij} - B_{2ij}) \geq \beta, \quad \forall i.
\end{aligned}
\]

(19)
For the above problem, a standard differentiable optimization software can now be used to find a solution.

5 Numerical Implementation and Experiments

5.1 Numerical Implementation

We consider a numerical implementation of the smooth and finite-dimensional model (19) derived in Sect. 4. We choose the initial values for the vector of intrinsic frequencies $\omega_i$ from the normal distribution with mean 2 and variance 1/2, namely $\omega_i \sim \mathcal{N}(2, 1/2)$ and the initial conditions $\theta_i(0), i = 1, \ldots, N$, are drawn from the uniform distribution over the interval $[-\pi/2, \pi/2], i = 1, \ldots, N$.

In solving the discretized and re-formulated optimization model (19), we have paired up the optimization modelling language AMPL [16] and the finite-dimensional optimization software Ipopt [39]. The AMPL–Ipopt suite was run on a 13-inch 2018 model MacBook Pro, with the operating system macOS Mojave (version 10.14.6), the processor 2.7 GHz Intel Core i7 and the memory 16 GB 2133 MHz LPDDR3. The Ipopt options $\text{tol}=1e-8$, $\text{acceptable\_tol}=1e-8$ and $\text{linear\_solver}=ma57$ were provided within an AMPL code.

We have used the following choice of the penalty parameters in model (19): $c_1 = 1, c_2 = c_3 = 100$. As mentioned before, the use of the terms with $c_2$ and $c_3$ accelerated convergence by 10–100 times, compared with the case when $c_2 = c_3 = 0$ and the Euclidean distance $\sum_{ij}(\tilde{d}(i) - \tilde{d}(j))^2$ is penalized instead.

A numerical solution of (19) yields nonnegative real values for $\tilde{A}_{ij} = B_{ij}^1 - B_{ij}^2$. It should be noted that $\tilde{A}_{ij}/\sigma$ for some $\sigma > 0$ does not necessarily result in $A_{ij} \in \{0, 1\}$. Therefore, we implement the following practical prototype algorithm in constructing a sparse binary adjacency matrix $A$ from the real matrix $\tilde{A}$.

Prototype Algorithm

\textbf{Step 1 (Initialization)} Using some $\omega_i \in \mathcal{N}(2, 1/2), \tilde{\theta}_i(0) \in \mathcal{U}[-\pi/2, \pi/2], i = 1, \ldots, N$, and integer $M > 0$, find a solution $\tilde{A}_{ij}$ to the relaxed model (19).

Set $\varepsilon > 0$ small, $\eta_0 = 0$, $T = 2000$ and $t_c = 1500$. Set the coupling strength $\sigma > 0$ large. Set $k = 0$.

\textbf{Step 2 (Generate adjacency matrix $A^{(k)}$)} Let $A_{ij}^{(k)} := \text{sgn}(\max\{0, \tilde{A}_{ij} - \eta_k\})$, for all $i, j = 1, \ldots, N$.

\textbf{Step 3 (Solve Kuramoto dynamics)} Solve (1) over $t \in [0, T]$, with $\sigma$, $\omega_i$ and $\theta_i(0)$ from Step 1 and $A^{(k)}$ from Step 2.

\textbf{Step 4 (Test synchrony to potentially increase sparsity of $A^{(k)}$)} If $\max_{t_c \leq t \leq T} \sum_{ij} |\dot{\theta}_i(t) - \dot{\theta}_j(t)| \leq \varepsilon$, then choose $\eta_{k+1} > \eta_k$, set $k := k + 1$ and go to Step 2.

\textbf{Step 5 (Stopping)} If $k \geq 1$, then declare “Algorithm successful,” return the adjacency matrix $A = A^{(k-1)}$. Otherwise, declare “Algorithm unsuccessful.” Stop.

In Step 2 of the algorithm above, we define the entries $A_{ij}^{(k)} := \text{sgn}(\max\{0, \tilde{A}_{ij} - \eta_k\})$ of $A^{(k)}$ in each iteration $k$. Note that with $\eta_k > 0$, some of the entries $\tilde{A}_{ij} - \eta_k$ may become negative, but after applying the max operator one gets $A_{ij}^{(k)} \in \{0, 1\}$. In
Step 4 of the algorithm, first, synchrony is tested with $\eta_0 = 0$, and then $\eta_k$ is increased in each iteration $k$ (i.e. $\eta_{k+1} > \eta_k$) so as to increase the number of 0-entries in $A^{(k+1)}$, i.e. to obtain a sparser $A^{(k+1)}$.

Note that the prototype algorithm above is deemed unsuccessful (in Step 5) only in the case when $k = 0$ and the (binary) adjacency matrix $A^{(0)}$ that is generated by Step 2 (with $\eta_0 = 0$) does not produce synchrony. In the numerical experiments, we have performed (those reported in this paper and those not reported), the prototype algorithm (as well as Algorithm 1 given below) was always successful.

Step 3 and the synchrony verification in Step 4 of the prototype algorithm may be replaced by the verification of a connectivity property, as connected graphs do synchronize. One such property is the strict positivity of the second smallest eigenvalue of the Laplacian of the graph. Alternatively, one may also replace the same parts of the prototype algorithm by the requirement of a “suitably small” value of $Q_L$ in Step 4, so as to have a “more desirable” connectivity property of the graph. Implementation of the latter criterion leads to a slightly different algorithm from the prototype as described below.

Algorithm 1

**Step 1** *(Initialization)* Using some $\omega_i \in \mathcal{N}(2, 1/2)$, $\theta_i(0) \in U[-\pi/2, \pi/2]$, $i = 1, \ldots, N$, and integer $M > 0$, find a solution $\tilde{A}_{ij}$ to the relaxed model (19). Set $\eta_0 > 0$ small. Set $k = 0$.

**Step 2** *(Generate adjacency matrix $A^{(k)})*$ Same as in Prototype Algorithm.

**Step 3** *(Potentially increase sparsity of $A^{(k)}$ while keeping spectral ratio small)*

While $Q_L$ is “small enough,” choose $\eta_{k+1} > \eta_k$, set $k := k + 1$ and go to Step 2.

**Step 4** *(Stopping)* Same as Step 5 of Prototype Algorithm.

Although $Q_L$ has not been included in the optimization model directly, it is computed in Step 3 of Algorithm 1 (exogenously) for each new graph generated in Step 2. We have observed in the numerical experiments that when $k = 0$, $Q_L$ is in general small enough; however, the matrix is not sufficiently sparse. As the sparsity of $A$ is increased, $Q_L$ is in general also increased. Then, if $Q_L$ is not “small enough” any longer, we stop and take the latest graph for which $Q_L$ is acceptable. In other words, we increase $k$ in Algorithm 1 until sparsity and the value of $Q_L$ are both acceptable.

Note that a solution of the relaxed model (19) depends on the random selection of the vectors $\omega$ and $\theta(0)$. Moreover, even if we fix the choice for $\omega$ and $\theta(0)$, each run of the AMPL-Ipopt suite with the same $\omega$ and $\theta(0)$ is observed to generate a different (presumably locally optimal) solution. This is something expected because of the combinatorial nature of the underlying model (6). So, the solution obtained with Algorithm 1, or Prototype Algorithm, will be different with each $\tilde{A}$ found by solving model (19). Prototype Algorithm is designed to promote sparsity of the graph, and Algorithm 1 produces graphs with a relatively small spectral ratio. In the numerical experiments we describe in the next section, we have always used Algorithm 1. This is because we wanted to make sure our resulting graphs had a relatively small spectral ratio, and the matrices produced by it had a high level of sparsity.
Remark 1 In applications, there might be a need to find out a small set of the so-called vital nodes [26] that play a critical role, for example, in propagating information and/or maintaining network connectivity. With this in mind, define a set of links in a connected network to be critical if their deletion produces a network which is either disconnected, or it has unacceptable connectivity properties such as a large $Q_L$. In Step 3 of Algorithm 1, we discard an adjacency matrix if it produces a disconnected graph, or if $Q_L$ is too large. Therefore, Step 3 in Algorithm 1 can ultimately be also used to identify a set of critical links. However, this potential use of Algorithm 1 is not investigated here as it is outside the scope of the current paper.

5.2 Numerical Experiments

In Sects. 5.2.1 and 5.2.2, Problems with 20 and 127 nodes, respectively, are solved. For simplicity in the computational modelling, we set $t_c := T = 20$ in each of these two problems, where $[0, T]$ is the domain, or time horizon, of $\theta$ as in (1). This “terminal” choice of $t_c$ indeed produces the desired synchronization, way earlier. We also use a moderate grid size, $M = 10$, for the time horizon so as to make computations faster.

The reason we have chosen $N = 127$ is that this number is of the form $N = 2^m - 1$, with $m = 7$, and therefore the 127-node graphs can be compared with those obtained in [37]. In the latter reference, graphs with good connectivity properties are constructed by combining a binary tree with an expander graph, as also briefly explained in Sect. 2.4. Our aim in choosing $N = 127$ is simply to be able to compare the connectivity properties of constructions such as those in [37] with those we obtain through Algorithm 1.

For each of the 20- and 127-node problems, we proceed as follows. First we obtain an acceptable (binary) adjacency matrix $A$ as an output of Algorithm 1. We randomly generate 30 sets of the vectors $\omega$ and $\theta(0)$, whose components are taken from the distributions $\mathcal{N}(2, 1/2)$ and $\mathcal{U}(-\pi/2, \pi/2)$, respectively. For each of the 30 randomly generated sets of $\omega$ and $\theta(0)$, we solve the Kuramoto system for a sequence of (fixed) values of $\sigma$. For this purpose we have used MATLAB, Release 2019b. In order to solve (1), we implemented the solver $\text{ode15s}$, effective for stiff ODEs, with the absolute and relative tolerances of $10^{-6}$.

Now, with the discrete solution of the phase vector $\theta(t)$ for a fixed value of $\sigma$ at hand, Equations (3)–(4) are used to obtain the corresponding $\bar{r}$. For numerical integration we have used Simpson’s rule. Then, using the 30 different sequences of $\bar{r}$ obtained for each of the 30 random data, the expected value $\mathbb{E}[\bar{r}]$ and the variance $\text{Var}[\bar{r}]$ of $\bar{r}$ are computed and plotted against the coupling strength $\sigma$. We also provide for each graph a histogram of the degrees of the nodes.

The variance plot can be used as a measure of robustness of the connectivity of the graph for $\omega$ and $\theta(0)$ coming from distributions that were also chosen in Algorithm 1. The plot provides a quick picture of what coupling strength values a network will be resilient, i.e. will be insensitive, to possible changes in $\omega$ and $\theta(0)$.

The adjacency matrices of the graphs mentioned in what follows can be accessed at the URL in [5].
**Remark 2** We recall that in Algorithm 1 first an edge-weighted graph with a weighted adjacency matrix $\tilde{A}$ is found as an approximate solution of the optimization problem (10). Then, by deleting edges which have smaller weights, a sparse synchronizing graph (of a balanced spread of degrees) with an adjacency matrix $A$ is obtained via Algorithm 1. With randomized choices of $\theta(0)$ and $\omega$, Algorithm 1 practically yields a new solution each time as there exist a large number of locally optimal solutions owing to the combinatorial nature of the problem.

**Remark 3** We obtain a synchronized network by satisfying the synchrony constraint in the optimization problem/model in (19), which is posed at the terminal time $T = 20$; in other words, we enforce synchrony (approximately) at the relatively short time of 20 units, instead of infinity. The accuracy of the transients up to the terminal time $T$ is not relevant for synchrony in this case; therefore, a fine discretization is not required as long as the system of ODEs representing the Kuramoto dynamics is stable and that the discretization, i.e. the numerical solution method applied to the ODE system, is stable.

We recall that for the Dahlquist test problem, the trapezoidal rule is an $A$-stable method, in that no matter how large or small the step size is the trapezoidal method is stable and so, as time tends to infinity, the solution point at infinity tends to the equilibrium of the Dahlquist problem—see, for example, [7,8]. In our case, the Kuramoto ODE system is obviously far more complicated than the Dahlquist equation and a similar conclusion for the stability of the discretized ODE system cannot possibly be derived analytically. However, in view of the known stability properties of the Kuramoto system, we use the trapezoidal method for discretization. Extensive numerical experiments indicate that a large step size, e.g. $h = T/M = 2$ can indeed be used in solving Problem (19). Moreover, we point out that smaller step sizes, i.e. a finer time grid, not only make the size of the optimization problem prohibitive but also do not necessarily yield different graphs, if the initial guesses for the optimization variables as well as the choice of the intrinsic frequencies are identical. We would like to point that the designed graphs are tested by solving the Kuramoto ODE system with a high numerical accuracy and ultimately synchronization of the graphs is verified.

**Remark 4** We have posed an optimization problem requiring achievement of three competing criteria: (i) sparsity, (ii) a narrow degree distribution and (iii) a good/robust synchronization. Criteria (i) and (ii) are combined to form the objective function of the optimization problem, and Criterion (iii) is enforced as a synchrony constraint in which synchrony is required to be satisfied within a small tolerance $\varepsilon = 10^{-4}$. In other words, the objective is to simultaneously maximize sparsity and minimize the degree and its range across the nodes of the network, subject to synchrony of the network practically in a short period of time rather than at infinity. The relaxed model (19) yields a weighted adjacency matrix $\tilde{A}$. Then, via Algorithm 1 edges/links which have smaller weights are deleted to obtain a number of adjacency matrices $A$ of sparse networks which are relatively synchronizable. One can in fact go in the other direction and first obtain the sparsest synchronizable network and then add remaining largest weight links one by one to obtain graphs with better synchrony properties (i.e. graphs synchronizable with a smaller coupling strength). One must note, however,
that a random addition of links to the sparsest synchronizable network obtained by our method will not in general furnish graphs with better synchrony properties.

**Remark 5** In the synchrony constraint of the optimization problem, practically we enforce that the trapezoidal discretization of the Kuramoto oscillators achieves synchrony after a relatively short duration of time (of 20 units). This requirement is sufficient to furnish synchronization robustly for relatively small values of the coupling strength $\sigma$, as illustrated by the averaged order parameter curves and their variances in Figs. 1 and 3 for the 20- and 127-node graphs.

### 5.2.1 20-Node Networks

We used Algorithm 1 for $N = 20$. We have chosen some representative instances that illustrate the effectiveness of our approach. Figure 1 corresponds to the graphs, denoted $G_i$, $i = 1, \ldots, 4$, that we obtained by using Algorithm 1.

In Fig. 1a, we display the plots of the expected values $E[\bar{r}]$ of $\bar{r}$ vs the coupling strength $\sigma$, shown by solid curves which are colour-coded for each $G_i$, $i = 1, \ldots, 4$. The dashed curves, also colour-coded, represent the standard deviation from each of the $E[\bar{r}]$ curves. Figure 1b, on the other hand, depicts the variance $\text{Var}[\bar{r}]$ of $\bar{r}$, colour-coded in the same way as in Fig. 1a. These plots indicate for which values of the coupling strength one can achieve connectivity, and thus synchrony, with a degree of robustness. Figure 2 displays the degree histogram of each graph.

Figure 1 suggests that the graph $G_1$, with 25 edges and $Q_L = 16.5$, would likely achieve a robust synchrony for $\sigma \geq 1.2$, since, for these values, $E[\bar{r}]$ is near 1.0 and $\text{Var}[\bar{r}]$ is relatively small. The graph $G_2$ has just one more edge than $G_1$; with 26 edges, it has a better value of $Q_L$, namely $Q_L = 13$. Figure 1a shows that its $E[\bar{r}]$ curve
Fig. 2 $N = 20$: degree distributions of the graphs $G_1$–$G_4$ obtained by Algorithm 1

comes near the value 1.0 more rapidly than that of $G_1$. Moreover, one can observe in Fig. 1b that its Var[$\tilde{r}$] curve decays relatively more quickly. As can be seen in Fig. 2, the degrees of the nodes of both graphs $G_1$ and $G_2$ range between 2 and 4, but a great majority of the nodes of $G_2$ are of degree 3 or 4.

Figure 2 also depicts that $G_3$ is a cubic graph, i.e. each of its nodes has degree 3. Cubic graphs are desirable in many practical applications because of their balanced workload sharing nature. We note that $G_3$ has 30 edges but $Q_L = 11.3$ is significantly smaller than those for $G_1$ and $G_2$. In the plots in Fig. 1, we observe that $E[\tilde{r}]$ approaches 1.0 and Var[$\tilde{r}$] (after making the usual peak) dies down much more rapidly than the those curves for $G_1$ and $G_2$, pointing to far more robustness in its synchronization.

The graph $G_4$ has 31 edges, and an even better value of $Q_L = 9.3$. The node degrees in this case range from 3 to 4, which can also be regarded as sharing workloads more evenly compared to $G_1$ and $G_2$. The “performance indicators” of the graph given in Fig. 1 are even more impressive than before, in terms of the criteria discussed for the previous graphs. Robustness of its synchronization is more evident with its Var[$\tilde{r}$] being much smaller than those for $G_1$, $G_2$ and $G_3$ with larger (synchronizing) values of $\sigma$.

5.2.2 127-Node Networks

To understand the influence of the penalizing terms in the relaxed and discretized Problem (19), we implemented Algorithm 1 by solving first Problem (19) using only the $\ell_1$-objective and ignoring all other terms which penalize a wide distribution of the degrees. This produced a graph with 178 edges, $Q_L \approx 300$, and degrees ranging from 1 to 13. This degree distribution is obviously not desirable.

Including the terms in (19) penalizing wider degree distributions across the graph nodes, i.e. setting $c_2 = c_3 = 100$, Algorithm 1 has produced more desirable graphs, e.g. the graphs $G_6$ and $G_7$, the features $E[\tilde{r}]$, Var[$\tilde{r}$] and the degree histograms of
Fig. 3 \( N = 127 \): mean and variance of the averaged order parameter curves of the graphs \( G_1 \)–\( G_4 \) obtained by Algorithm 1. In (a), the dashed curves represent the standard deviation from each of the solid \( E[\bar{r}] \) versus \( \sigma \) curves, corresponding to the variance in (b).

Fig. 4 \( N = 127 \): Degree distributions of the graphs \( G_5 \)–\( G_{10} \) obtained by Algorithm 1 which are depicted in Figs. 3 and 4. The generated graphs are still sparse and have much better spectral ratios \( Q_L \).
As mentioned in Sect. 2.4, we have also constructed a 127-node graph, namely $G_5$, using the tree-expander approach introduced and described in [37]. All of $G_5$, $G_6$ and $G_7$ show somewhat similar/close characteristics in terms of the $E[\bar{r}]$ and $\text{Var}[\bar{r}]$ curves, although the graph featured in Fig. 4 clearly has a much better degree distribution; its nodes have either degree 2 or degree 3. A closer examination of the $E[\bar{r}]$ and $\text{Var}[\bar{r}]$ curves suggests that $G_7$ will synchronize more robustly than $G_5$ and $G_6$, since its $E[\bar{r}]$ curve is above those of $G_5$ and $G_6$ and its $\text{Var}[\bar{r}]$ curve is below those of $G_5$ and $G_6$, for $\sigma \geq 1.4$.

The tree-expander construction of a graph that is introduced and described in [37] allows/requires choices amongst many options; so it is possible to get quite different graphs by this construction. A different graph constructed using an alternative sequence of choices is reported in [37], the features of which can also be seen in Fig. 3, labelled $G_8$. This graph has better $E[\bar{r}]$ and $\text{Var}[\bar{r}]$ curves, in that the $E[\bar{r}]$ curve approaches 1.0 relatively more quickly and the peak of $\text{Var}[\bar{r}]$ is smaller as well as it dying down more quickly, compared with the previous three graphs discussed, namely $G_5$, $G_6$ and $G_7$.

A desirable/useful characteristic of a graph would presumably be the following: (i) $E[\bar{r}]$ is almost “S-shaped” and approaches 1.0 rapidly and (ii) $\text{Var}[\bar{r}]$ peaks early and dies down quickly. Such a characteristic seems to be attainable by adding more edges, via an investigation using our optimization approach in Algorithm 1. In what follows, we consider solutions which have more edges, namely the graphs with 180 and 240 edges, respectively.

One might then argue that by increasing the number of edges the sparsity might be compromised. This would really depend on a particular application and what (number of edges) is practically meant by a sparse graph. Define the sparsity of an undirected graph as $1 - E/(N(N - 1)/2)$. In the previously discussed 127-node graphs, the percentage sparsity is 98.03% ($G_5$ and $G_8$, 158 edges), 97.93% ($G_6$, 166 edges), 97.91% ($G_7$, 167 edges). In the next two example graphs, the sparsity is 97.75% (180 edges) and 97.00% (240 edges), respectively. In all of the 127-node examples presented in this paper, the sparsity might be regarded to be quite close to one another (depending on a particular application, of course), which in this case is around 97–98%.

Figure 3 features the optimized graph $G_9$ found by using Algorithm 1, which has 180 edges, instead of the 158–167 edges the previous graphs have. Although $G_9$ has more edges than the previous graphs, this may not necessarily be regarded to be compromising sparsity (as discussed above). Now, both of the $E[\bar{r}]$ and $\text{Var}[\bar{r}]$ curves have markedly better behaviour: the peak of $\text{Var}[\bar{r}]$ is shifted to the left and $E[\bar{r}]$ approaches 1.0 markedly more rapidly. One might point to the rather wider range of degrees compared to the previous graphs; however, again depending on the application, this “slightly” wider range might be acceptable.

Figure 3 also features the optimized graph $G_{10}$, again found by using Algorithm 1, which has 240 edges. Although 240 edges sounds to be much bigger than 160 edges, the sparsity of $G_{10}$ is worse only by 1 percentage point, which might again be acceptable in some practical situations. The behaviour of the graph shown by the $E[\bar{f}]$ and $\text{Var}[\bar{r}]$ curves is rather impressive: the curve of $E[\bar{f}]$ is almost S-shaped, settling very near 1.0 quite early. The peak of $\text{Var}[\bar{f}]$ has been pushed to the far left and $\text{Var}[\bar{r}]$ dies down to
Table 1

| N   | Minimum CPU time (s) | Average CPU time (s) | Maximum CPU time (s) | Success rate   |
|-----|----------------------|----------------------|----------------------|----------------|
| 20  | 3                    | 5                    | 8                    | 17/30 (57%)    |
| 30  | 10                   | 18                   | 54                   | 23/30 (77%)    |
| 50  | 64                   | 139                  | 365                  | 14/30 (47%)    |
| 127 | 2038                 | 4419                 | 7828                 | 23/30 (77%)    |

almost zero around $\sigma = 1.0$ where the graph synchronize very well. These points to a desirable robust synchrony of the graph.

It is in the comparison of $G_5, G_8$ (the two expanders) with $G_9$ and $G_{10}$ that we see the trade-off opportunity of the optimization, as mentioned in the Introduction. The former two graphs meet the strict criterion of narrow degree distribution, with a certain impact on the level of synchronization that can be achieved—and are provably expander graphs. The latter two relax the degree narrowness and with a small increase in the number of links provide for enhanced synchronization. Depending on the tolerance for irregularity in the degree distribution (according to the application of the graph), $G_{10}$ may be an acceptable compromise between sparsity, regularity and synchrony.

5.2.3 Computational Effort

We have run AMPL–Ipopt suite to solve the optimization model (19), with randomly generated data, 30 times, with the graph sizes $N = 20, 30, 50, \text{ and } 127$, on the computer the specifications of which we provide in Sect. 5.1. As can be seen from Table 1, the AMPL–Ipopt suite was successful in about slightly more than half the time it was run. As can also be seen, the elapsed computational time grew exponentially with the graph size, as expected. Despite the exponential computational complexity, (locally) optimized graphs can be obtained in a reasonable amount of time. For example, a locally optimal 127-node graph can be designed in about one to two hours’ time.

Once $\bar{A}$ is found by solving model (19) and an adjacency matrix $A$ is constructed by Algorithm 1, the expected value $E[\bar{r}]$ and the variance $\text{Var}[\bar{r}]$ of $\bar{r}$ are computed and plotted against the coupling strength $\sigma$. This computation also incurs a comparable CPU time, in addition to the CPU times reported in Table 1. As we mention in Sect. 5.2, we use the MATLAB solver $\text{ode15s}$, effective for stiff ODEs, to solve (1), which can take a long time depending on the randomly generated data. In the case of $N = 20$, the CPU times for the graphs $G_1$ and $G_4$ are about 6.2 and 4.3 minutes, respectively. Given the fact that $G_4$ is more robust (although less sparse) than $G_1$, it is fair to say that these computations are expected to take shorter time for $G_1$ than those for $G_4$. We observe a similar situation for the $N = 127$ case: the CPU times for the graphs $G_5$ and $G_{10}$ are about 180 and 70 minutes, respectively.

6 Conclusion and Future Work

We have proposed an optimization algorithm (Algorithm 1) for designing sparse undirected networks, i.e. graphs, which synchronize well. The algorithms involve solution
of a discretized relaxed mathematical model which maximize sparsity and minimize the spread of degrees (in some sense) at the same time. Synchrony of the network is posed as a constraint by means of the well-known Kuramoto system of coupled oscillators. By means of carefully chosen examples, from small to large scale, we illustrate the working of our algorithm and optimization model. The outcome is a method to generate sparse well-balanced graphs with good synchronization from a random set of frequencies and initial phases. These graphs retain these properties when other frequency choices are used. In other words, in the resultant graphs there is not a strong correlation between the frequencies and degrees of the nodes (with a cubic generated in one case) underlying this insensitivity.

We have illustrated that our algorithm is successful in designing sparse synchronizing graphs with a relatively narrow distribution of degrees and small Laplacian eigenvalue ratio $Q_L$. By allowing more edges in the graph, but without sacrificing the sparsity much, we managed to obtain impressive synchronizing behaviour in terms of the expected order parameter curve $E[\bar{r}]$ and its variance $\text{Var}[\bar{r}]$. These graphs synchronize in a robust manner in that for relatively small synchronizing values of $\bar{r}$ the variance of $\bar{r}$ under randomized data is small. In particular, we have demonstrated explicitly the types of trade-off considerations that may be applied in choosing an optimal graph.

As pointed out in Remark 1, Algorithm 1 might be used as a valuable tool in identifying the so-called vital nodes and the critical set of links of the optimized sparse networks.

We have not explicitly incorporated the spectral ratio $Q_L$ into our optimization model. However, the numerical experiments have demonstrated that the additional terms involving the minimum and maximum degrees in the objective function promote smaller $Q_L$. Having said this, a measure of $Q_L$ might still be directly/explicitly included in a future model by using the Rayleigh quotient and the representation properties of the Laplacian using quadratic functions, such as the one in [29, Equation (14)]. Similar expressions that can be investigated can be found in [14, Remark 4.2].

Another interesting feature of a future optimization model would be constraints imposed on the degrees which the nodes of a graph should have. This would result in more targeted design of networks with certain desired load distributions.

Acknowledgements The authors are indebted to the two anonymous reviewers for carefully reading the initial submission and making valuable suggestions and comments which improved the paper. They would also like to acknowledge valuable discussions with Subhra Dey at the initiation of this project and with Richard Taylor during its later stages. This research was a collaboration under the auspices of the Modelling Complex Warfighting initiative between the Commonwealth of Australia (represented by the Defence Science and Technology Group) and the University of South Australia through a Defence Science Partnerships agreement.

Data Availability The datasets generated and analysed during the current study are available in the ArXiv repository, https://arxiv.org/src/2006.00428v1/anc—also see Reference [5].
References

1. Alt, W., Kaya, C.Y., Schneider, C.: Dualization and discretization of linear-quadratic control problems with bang-bang solutions. EURO J. Comput. Optim. 4, 47–77 (2016)
2. Banihashemi, N., Kaya, C.Y.: Inexact restoration for Euler discretization of box-constrained optimal control problems. J. Optim. Theory Appl. 156, 726–760 (2013)
3. Barahona, M., Pecora, L.M.: Synchronization in small world systems. Phys. Rev. Lett. 89, 054101 (2002)
4. Brede, M.: Local versus global synchronization in networks of non-identical Kuramoto oscillators. Eur. Phys. J. B. 62, 87 (2008)
5. Burachik, R. S., Kalloniatis, A. C., Kaya, C. Y.: Ancillary files for the preprint arXiv:2006.00428v1 (2020)
6. Burachik, R.S., Kaya, C.Y., Majeed, S.N.: A duality approach for solving control-constrained linear-quadratic optimal control problems. SIAM J. Control Optim. 52, 1771–1782 (2014)
7. Butcher, J.C.: Numerical Methods for Ordinary Differential Equations, 3rd edn. Wiley, Chichester (2016)
8. Corless, R.M., Kaya, C.Y., Moir, R.H.C.: Optimal residuals and the Dahlquist test problem. Numer. Algorithms 81, 1253–1274 (2019)
9. Dekker, A.H.: Studying organisational topology with simple computational models. J. Artif. Soc. Simul. 10, 6 (2007)
10. Dekker, A.H., Taylor, R.: Synchronization properties of trees in the Kuramoto model. SIAM J. Appl. Dyn. Syst. 12, 596–617 (2013)
11. Donetti, L., Hurtado, P.I., Munoz, M.A.: Entangled networks, synchronization, and optimal network topology. Phys. Rev. Lett. 95, 188701 (2005)
12. Donetti, L., Neri, F., Munoz, M.A.: Optimal network topologies: expanders, cages, Ramanujan graphs, entangled networks and all that. J. Stat. Mech. P08007 (2006)
13. Estrada, E., Gago, S., Caporossi, G.: Design of highly synchronizable and robust networks. Automatica 46, 1835 (2010)
14. Fazlyab, M., Doerfler, F., Preciado, V.M.: Optimal network design for synchronization of coupled oscillators. Automatica 84, 181 (2017)
15. Forger, D.B.: Biological Clocks, Rhythms, and Oscillations. The MIT Press, Cambridge (2017)
16. Fourer, R., Gay, D.M., Kernighan, B.W.: AMPL: A Modeling Language for Mathematical Programming, 2nd ed. (Brooks/Cole Publishing Company/Cengage Learning, 2003)
17. Hong, H., Choi, M.Y., Kim, B.J.: Synchronization on small-world networks. Phys. Rev. E 65, 026139 (2002)
18. Hoory, S., Linial, N., Widgerson, A.: Expander graphs and their applications. Bull. Am. Math. Soc. 43(4), 439–561 (2006)
19. Ichinomiya, T.: Frequency synchronization in a random oscillator network. Phys. Rev. E 70(2), 026116 (2004)
20. Kalloniatis, A.C., McLennan-Smith, T.A., Roberts, D.O.: Modelling distributed decision-making in command and control using stochastic network synchronisation. Eur. J. Oper. Res. 284, 588–603 (2020)
21. Kaya, C.Y.: Inexact restoration for Runge–Kutta discretization of optimal control problems. SIAM J. Numer. Anal. 48(4), 1492–1517 (2010)
22. Kaya, C.Y., Martínez, J.M.: Euler discretization for inexact restoration and optimal control. J. Optim. Theory Appl. 134, 191–206 (2007)
23. Kaya, C.Y., Maurer, H.: A numerical method for nonconvex multi-objective optimal control problems. Comput. Optim. Appl. 57(3), 685–702 (2014)
24. Kelly, D., Gottwald, G.A.: On the topology of synchrony optimized networks of a Kuramoto-model with non-identical oscillators. Chaos 21, 025110 (2011)
25. Kuramoto, Y.: Chemical Oscillations, Waves, and Turbulence. Springer, Berlin (1984)
26. Lü, L., Chen, D., Ren, X.L., Zhang, Q.M., Zhang, Y.C., Zhou, T.: Vital nodes identification in complex networks. Phys. Rep. 650, 1–63 (2016)
27. Medvedev, G.S., Tang, X.: Stability of twisted states in the Kuramoto model on Cayley and Random graphs. J. Nonlinear Sci. 25, 1169–1208 (2015)
28. Medvedev, G.S., Tang, X.: Synchronization of coupled chaotic maps. Physica D 304–305, 42–51 (2015)
29. Mohar, B.: Some applications of Laplace eigenvalues of graphs. In: Hahn, G., Sabidussi, G. (eds.) Graph Symmetry: Algebraic Methods and Applications. NATO ASI Series C 497, pp. 225–275. Kluwer, Dordrecht (1997)
30. Nocedal, J., Wright, S.: Numerical Optimization. Springer, New York (2006)
31. Ochab, J., Gora, P.F.: Synchronization of coupled oscillators in a local one-dimensional Kuramoto model. Acta Phys. Polonica B Proc. Suppl. 3, 453–462 (2010)
32. Oh, E., Lee, D.-S., Kahng, B., Kim, D.: Synchronization transition of heterogeneously coupled oscillators on scale-free networks. Phys. Rev. E 75, 011104 (2007)
33. Pecora, L.M., Carroll, T.L.: Master stability functions for synchronized coupled systems. Phys. Rev. Lett. 80, 2109 (1998)
34. Rodrigues, F.A., Peron, T.K.D.M., Ji, P., Kurths, J.: The Kuramoto model in complex networks. Phys. Rep. 610, 1–98 (2016)
35. Rogge, J.A., Aeyels, D.: Stability of phase locking in a ring of unidirectionally coupled oscillators. J. Phys. A Math. Gen. 37, 11135–11148 (2004)
36. Tanaka, T., Aoyagi, T.: Optimal weighted networks of phase oscillators for synchronization. Phys. Rev. E. 78, 046210 (2008)
37. Taylor, R., Kalloniatis, A., Hoek, K.: Organisational hierarchy constructions with easy Kuramoto synchronisation. J. Phys. A Math. Theor. 53(8), 085701 (2020)
38. Vossen, G., Maurer, H.: On $L^1$-minimization in optimal control and applications to robotics. Opt. Control Appl. Methods 27, 301–321 (2006)
39. Wächter, A., Biegler, L.T.: On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. Math. Progr. 106, 25–57 (2006)
40. Yanagita, T., Mikhailov, A.S.: Design of oscillator networks with enhanced synchronization tolerance against noise. Phys. Rev. E. 85, 056206 (2012)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.