Network community detection via iterative edge removal in a flocking-like system

Filipe Alves Neto Verri, Roberto Alves Gueleri, Qiusheng Zheng, Junbao Zhang, and Liang Zhao

1 Computer Science Division, Aeronautics Institute of Technology, São José dos Campos, Brazil
2 Institute of Mathematics and Computer Science, University of São Paulo, São Carlos, Brazil
3 School of Computer Science, Zhongyuan University of Technology, Zhengzhou, China
4 Faculty of Philosophy, Science and Letters at Ribeirão Preto, University of São Paulo, Ribeirão Preto, Brazil

Abstract We present a network community-detection technique based on properties that emerge from a nature-inspired flocking system. Our algorithm comprises two alternating mechanisms: first, we control the particles alignment in higher dimensional space and, second, we present an iterative process of edge removal. These mechanisms together can potentially reduce accidental alignment among particles from different communities and, consequently, the model can generate robust community-detection results. In the proposed model, a random-direction unit vector is assigned to each vertex initially. A nonlinear dynamic law is established, so that neighboring vertices try to become aligned with each other. After some time, the system stops and edges that connect the least-aligned pairs of vertices are removed. Then, the evolution starts over without the removed edges, and after enough number of removal rounds, each community becomes a connected component. The proposed approach is evaluated using widely accepted benchmarks and real-world networks. Experimental results reveal that the method is robust and excels on a wide variety of networks. For large sparse networks, the edge-removal process runs in quasilinear time, which enables application in large-scale networks. Moreover, the distributed nature of the process eases the parallel implementation of the model.

1 Introduction

The study of complex networks attracts many studies from different areas. Networks are graphs that represent the relationships among individuals in many real-world complex systems. Each vertex is an object of study, and an edge exists if its endpoints interact somehow [22, 27].

A community structure is commonly found in many real networks, such as social networks [6, 18], oil–water flow structure [9], human mobility networks [26], spatial structure of urban movement in large cities [30], corporate elite networks in the fields of politics and economy [11], and many more. Formally, communities are groups of densely connected vertices [7, 23, 24], while connections between different communities are sparser.

The problem of community detection is related to the graph partition problem in graph theory. Finding the optimal partition is an NP-hard problem in most cases [8], thus making room for many researches to find out sub-optimal solutions in feasible time.

Newman and Girvan [20] have proposed a metric called modularity, whose purpose is to quantify how a network is likely to display community structure [15]. It does not make any assumption on a-priori knowledge of the network, e.g., vertex labels. One of the algorithms employed in this paper for comparison is called “Cluster Fast Greed”, or just CFG, and it is based on a greedy optimization of modularity [7, 14]. It performs fast, in time $O(m \log n)$, where $d$ is the depth of the dendrogram describing the network’s hierarchical community structure returned by the algorithm, $n$ is the number of vertices, and $m$ is the number of edges in the network. In cases where $m \sim n$ and $d \sim \log n$, it runs in quasilinear time: $O(n \log^2 n)$. Another algorithm employed here is the so-called “Louvain”, which is based on the optimization of modularity too [5, 14]. The authors advocate in favour of the computation time, which makes the algorithm applicable on large networks.

The relationship between modularity and synchronization is an interesting research topic [2, 4, 21]. However, a few community-detection algorithms have exploited this relationship [24]. Synchronization dynamics are relevant to the one proposed in this paper, where the flocking alignment is somewhat similar to synchronization state. In our proposed model, particles evolve in a $D$-dimensional space, where $D > 1$ is flexible. Thus, particles align in a space with higher dimensionality than other synchronization dynamics. Such a feature potentially reduces accidental alignment among parti-
particles from different communities. Consequently, the technique is robust and good results have been obtained.

The first alternating step of our model is a nonlinear collective complex system that takes inspiration from the flocking formation in nature [19]. In the second stage, we measure the misalignment of each pair of vertices that are directly connected and remove a fraction of edges that result the highest misalignments. Flocks are groups of individuals that move in a coordinated fashion. This coordinated motion emerges even in the absence of any leader, what makes it a self-organizing phenomenon. In our model, each vertex is an aligning particle. Therefore, each vertex carries a velocity vector, pointing to a random direction at the beginning and, as the process evolves, progressively turns itself toward the same direction of its neighboring vertices. As a simplification of the process, the vertices actually do not move, so the term “velocity” is just an analogy to the direction of motion in flocking systems.

The dynamical process is suspended after a certain number of iterations, and then, the second stage takes place. The edge that connects the least-aligned pair of vertices is supposed to link distinct communities. After enough number of removal cycles, most of the inter-community edges are expected to be removed, and thus, the network becomes partitioned into disconnected components.

Our model is evaluated using widely accepted benchmarks and real-world networks. It not only excels on many different scenarios but also has very low computational cost. As a result, the research shows potential for a broad range of applications, including big data.

The rest of this paper is organized as follows. Sections 2 and 3 describe the proposed model and present analytical results of the system. In Sect. 4, computer simulations illustrate the process and assess its performance. Finally, Sect. 5 discusses and concludes this paper.

2 Model description

Here, we present the iterative edge-removal approach in detail. As mentioned before, the process is divided in two alternating stages: the particle-alignment dynamical process and the edge-removal stage itself.

2.1 The particle-alignment dynamical model

Given an undirected network, free of self-loops, and multiple edges, we let each vertex \( i \in \{1, 2, \ldots, n\} \) be a particle in the collective dynamical system. Therefore, each vertex carries a velocity vector \( \mathbf{v}_i(t) \in \mathbb{R}^D \), which points to some direction in a multi-dimensional space. The vertices actually do not move at all, so the term “velocity” is just an analogy to the direction of motion in flocking systems, whose moving particles try to align themselves to their neighbors, i.e., try to take the same direction of motion.

Edges of the network define the neighborhood of each particle \( i \), i.e., those particles to which \( i \) can directly interact. We denote \( A_{ij} = 1 \) if vertices \( i \) and \( j \) interact, and \( A_{ij} = 0 \) otherwise. The neighborhood is unchanged throughout the dynamical process, and thus, the interaction network is the same all the time.

Initially, each particle \( i \) is assigned a random initial velocity \( \mathbf{v}_i(0) \), which is a unit vector pointing to a random direction. A three-dimensional space \((D = 3)\) is employed for all the experiments presented in this paper. However, how dimensionality impacts on the final results has to be investigated.

The nonlinear dynamical system is governed by the expression

\[
\mathbf{v}_i(t+1) = \frac{\mathbf{v}_i(t)}{\|\mathbf{v}_i(t)\|} + \alpha \frac{1}{k_i} \sum_{j=1}^{n} A_{ij} \left( \frac{\mathbf{v}_j(t)}{\|\mathbf{v}_j(t)\|} - \frac{\mathbf{v}_i(t)}{\|\mathbf{v}_i(t)\|} \right),
\]

where \( \|\cdot\| \) denotes the Euclidean norm. The nonlinearity of the dynamical system is introduced by the velocity normalization. Variable \( t \) is the iteration index (time), which starts from zero. Parameter \( \alpha > 0 \) defines how fast the velocities are updated. The value \( k_i \) is the degree of vertex \( i \), i.e., the number of neighbors it has

\[
k_i = \sum_{j=1}^{n} A_{ij}.
\]

Although the particles do not move—there is even no position defined to them—using the unit vector, we enforce something analogous to the “constant-speed motion” that each particle would perform. In addition, it enforces that at any given time-step \( t \), all the particles would move at the same speed. Constant-speed motion is a property commonly modelled in studies of self-propelled particles [19,28]. It is also responsible for the rotational symmetry breaking that makes the set of particles agree on the same velocity direction. By not enforcing such normalization, all velocities would vanish to zero—or close to zero—due to their random initial distribution, making opposite vectors neutralize each other.

In a connected network, all velocity vectors asymptotically evolve towards the same value (consult Sect. 3.2). Throughout that process, however, vectors in the same community tend to follow a similar path (consult Sect. 4.1). After some time, the pattern displayed by the velocity vectors enables us to tell which ones are most likely to belong to different communities.

2.2 The iterative edge-removal process

We define the misalignment coefficient \( H_{ij}(t) \) as the level of disorder in terms of velocity-vectors’ misalignment between nodes \( i \) and \( j \). Such index is mathematically expressed by

\[
H_{ij}(t) = d_1 \left( \frac{\mathbf{v}_i(t)}{\|\mathbf{v}_i(t)\|} \cdot \frac{\mathbf{v}_j(t)}{\|\mathbf{v}_j(t)\|} \right),
\]

where \( d_1 \) is a normalization constant ensuring all misalignment coefficients are in the range \([0, 1] \).
where \( d_1(\cdot, \cdot) \) is the \( L_1 \) distance between vectors. In other words, \( H_{ij}(t) \) is just the city-block distance between the normalized velocity vectors of the vertices \( i \) and \( j \).

As we will see through the experiments presented in Sect. 4, misalignment coefficients of edges that connect distinct communities decrease slower than those connecting vertices inside the same community. It builds the basis of our iterative edge-removal approach: removing the edges with highest misalignment coefficients is likely to remove inter-community edges, thus making the distinction between different communities clearer and clearer over removal cycles. Therefore, the overall edge-removal process consists of the following steps:

1. After assigning random initial velocity vectors to every vertex, run the dynamical particle-aligning model, as described in the previous section, up to some number of time-steps. (Number of steps is discussed in Sect. 4.2.)
2. Once the dynamics is interrupted, collect the misalignment coefficient of each pair of interacting vertices. It is possible to run the dynamical model (step 1) multiple times, using different random assignments to the set of initial velocities. In this case, the final coefficient of each edge can be just the summation of individual coefficients collected after each run.
3. Remove the edges with the highest misalignment coefficients, then go to step 1 and run the dynamical model again, but this time using the new network without the removed edges.

Steps 1, 2, and 3, together, form what we call “cycle” or “round”. In this paper, we study the influence of running the dynamical model multiple times per cycle. We also study the influence of removing different numbers of edges per cycle.

### 3 Theoretical results on flocking alignment

We also present analytical and argumentative study of the dynamics given by Eq. (1).

#### 3.1 Domain of the velocity vectors

A state at time \( t + 1 \) in which \( \| \mathbf{v}_i(t) \| = 0 \) for any \( i \) is a singularity. To deal with this problem, we need to restrict the parameter \( \alpha \). Once \( \alpha > 0 \) by definition, we show that, if \( \alpha < 0.5 \) and \( \| \mathbf{v}_i(0) \| = 1 \) for all \( i \), then \( 0 < \| \mathbf{v}_i(t) \| \leq 1 \) for all \( i \) and \( t > 0 \). Thus, for any reasonable \( \alpha \) and the initial conditions proposed in this paper, the velocity vectors are nonzero vector with norm less than or equal to 1.

Given the restrictions of the parameters and initial conditions, the following lemmas prove that \( \| \mathbf{v}_i(t) \| \in (0,1] \) for all \( i \) at any time \( t \), guaranteeing the expected behavior of the model.

#### Lemma 1

Given that \( \| \mathbf{v}_i(0) \| \leq 1 \), for all particle \( i \), the norm of the velocity of every particle will not surpass 1.

**Proof**

For \( t = 0 \), \( \| \mathbf{v}_i(0) \| \leq 1 \) from the statement. Assume \( \| \mathbf{v}_i(t) \| \leq 1 \) for some \( t \). Then

\[
\| \mathbf{v}_i(t+1) \| \\
\text{Equation (1)} = \left\| \frac{\mathbf{v}_i(t)}{\| \mathbf{v}_i(t) \|} + \alpha \frac{1}{k_i} \sum_{j=1}^{n} A_{ij} \left( \frac{\mathbf{v}_i(t)}{\| \mathbf{v}_i(t) \|} - \frac{\mathbf{v}_j(t)}{\| \mathbf{v}_j(t) \|} \right) \right\| \\
= \left\| \frac{\mathbf{v}_i(t)}{\| \mathbf{v}_i(t) \|} - \alpha \frac{1}{k_i} \sum_{j=1}^{n} A_{ij} \frac{\mathbf{v}_j(t)}{\| \mathbf{v}_j(t) \|} \right\| \\
\leq (1 - \alpha) \left\| \frac{\mathbf{v}_i(t)}{\| \mathbf{v}_i(t) \|} \right\| + \alpha \frac{1}{k_i} \sum_{j} A_{ij} \left\| \frac{\mathbf{v}_j(t)}{\| \mathbf{v}_j(t) \|} \right\| \\
= (1 - \alpha) + \alpha \| \mathbf{v}_i(t+1) \| \leq 1.
\]

Thus, by induction, \( \| \mathbf{v}_i(t) \| \leq 1 \) for all \( i \).

#### Lemma 2

Given that \( \| \mathbf{v}_i(0) \| > 0 \) and \( \alpha < 0.5 \), for all particle \( i \), the norm of the velocity of every particle is strictly greater than 0 for any time \( t > 0 \).

**Proof**

For \( t = 0 \), \( \| \mathbf{v}_i(0) \| > 0 \) from the statement. Assume that \( \| \mathbf{v}_i(t) \| > 0 \) for some \( t \). We show by contradiction that \( \| \mathbf{v}_i(t+1) \| > 0 \).

If there exists \( \alpha = \alpha_0 \), \( 0 < \alpha_0 < 0.5 \), such that

\[
\| \mathbf{v}_i(t+1) \| = 0.
\]

Then,

\[
\| \mathbf{v}_i(t+1) \| = 0 = (1 - \alpha_0) \left\| \frac{\mathbf{v}_i(t)}{\| \mathbf{v}_i(t) \|} + \alpha_0 \frac{1}{k_i} \sum_{j} A_{ij} \frac{\mathbf{v}_j(t)}{\| \mathbf{v}_j(t) \|} \right\| \\
\implies - \frac{\mathbf{v}_i(t)}{\| \mathbf{v}_i(t) \|} = \alpha_0 \left( - \frac{\mathbf{v}_i(t)}{\| \mathbf{v}_i(t) \|} + \frac{1}{k_i} \sum_{j} A_{ij} \frac{\mathbf{v}_j(t)}{\| \mathbf{v}_j(t) \|} \right) \\
\implies \left\| - \frac{\mathbf{v}_i(t)}{\| \mathbf{v}_i(t) \|} \right\| = \alpha_0 \left( \left\| - \frac{\mathbf{v}_i(t)}{\| \mathbf{v}_i(t) \|} \right\| + \frac{1}{k_i} \sum_{j} A_{ij} \left\| \frac{\mathbf{v}_j(t)}{\| \mathbf{v}_j(t) \|} \right\| \right) \\
\implies 1 \leq \alpha_0 \left( \left\| - \frac{\mathbf{v}_i(t)}{\| \mathbf{v}_i(t) \|} \right\| + \frac{1}{k_i} \sum_{j} A_{ij} \left\| \frac{\mathbf{v}_j(t)}{\| \mathbf{v}_j(t) \|} \right\| \right) = 2\alpha_0 \\
\implies \alpha_0 \geq 0.5.
\]

By contradiction, there is no \( \alpha_0 < 0.5 \), such that \( \| \mathbf{v}_i(t+1) \| = 0 \).

Thus, by induction, \( \| \mathbf{v}_i(t) \| > 0 \) for all \( i \), \( t \).

#### 3.2 Alignment of the velocity vectors

The core mechanism in our method is measuring small misalignments between connected particles and deciding which edge will be removed. A question that rises is whether the velocity vectors converge to a single point or not, that is, if the particles align or not. We show that perfect alignment of particles in the same connected component is most likely to happen. The system would
also be in equilibrium if vectors are in perfect opposition to each other. Such condition would need not only very specific initial velocity vectors but also specific network configuration, and thus, this case is extremely unlikely.

To perform the particle-alignment study, we use a continuous approximation of the evolution equations. Given that

$$\Delta \mathbf{v}_i(t) = \mathbf{v}_i(t + 1) - \mathbf{v}_i(t)$$

$$= \frac{\mathbf{v}_i(t)}{\| \mathbf{v}_i(t) \|} - \mathbf{v}_i(t)$$

$$+ \alpha \frac{1}{k_i} \sum_{j=1}^{n} A_{ij} \left( \frac{\mathbf{v}_j(t)}{\| \mathbf{v}_j(t) \|} - \frac{\mathbf{v}_i(t)}{\| \mathbf{v}_i(t) \|} \right)$$

$$= \frac{\mathbf{v}_i(t) - \| \mathbf{v}_i(t) \| \mathbf{v}_i(t)}{\| \mathbf{v}_i(t) \|}$$

$$+ \alpha \frac{1}{k_i} \sum_{j=1}^{n} A_{ij} \left( \frac{\mathbf{v}_j(t)}{\| \mathbf{v}_j(t) \|} - \frac{\mathbf{v}_i(t)}{\| \mathbf{v}_i(t) \|} \right),$$

we approximate the dynamics using

$$\frac{d}{dt} \mathbf{v}_i(t) = 1 - \| \mathbf{v}_i(t) \| \mathbf{v}_i(t)$$

$$+ \alpha \frac{1}{k_i} \sum_{j=1}^{n} A_{ij} \left( \frac{\mathbf{v}_j(t)}{\| \mathbf{v}_j(t) \|} - \frac{\mathbf{v}_i(t)}{\| \mathbf{v}_i(t) \|} \right).$$

In the next equations, the summation indices $i$, $j$, and $q$ always lie in the interval $[1, n]$. The intervals are dropped to simplify the notation.

We are not interested in the evolution of the unnormalized velocities but in their normalized forms. To improve readability, we set

$$\mathbf{x}_i(t) = \frac{\mathbf{v}_i(t)}{\| \mathbf{v}_i(t) \|} \quad \text{and} \quad \mathbf{z}_i(t) = \| \mathbf{v}_i(t) \|.$$

Thus, the governing equations of the normalized velocities $\mathbf{x}_i(t)$ are

$$\frac{d}{dt} \mathbf{x}_i(t) = \alpha \frac{1}{k_i z_i(t)} \sum_{q} A_{iq} \left( \mathbf{x}_q(t) - (\mathbf{x}_i(t) \cdot \mathbf{x}_q(t)) \mathbf{x}_i(t) \right)$$

$$\frac{d}{dt} \mathbf{z}_i(t) = 1 - \mathbf{z}_i(t) - \alpha \sum_{q} A_{iq} \left( 1 - \mathbf{x}_i(t) \cdot \mathbf{x}_q(t) \right),$$

where $\cdot$ stands for the dot product operator.

**Theorem 1** The aligned state $\mathbf{x}_i(t) = \mathbf{x}_j(t)$, for all particle $i$ that interacts with particle $j$, is stable in the sense of Lyapunov.

**Proof** We define the energy function $E$ that reaches zero only when the aligned state is reached, and it increases as the velocities misalign. Let

$$\mathbf{u}_{ij} = \mathbf{x}_j - \mathbf{x}_i;$$

the candidate Lyapunov function is

$$E = \frac{1}{4} \sum_{i} \sum_{j} A_{ij} \mathbf{u}_{ij} \cdot \mathbf{u}_{ij}.$$  

Its derivative is

$$\frac{d}{dt} E = \sum_{i} \sum_{j} \frac{\partial}{\partial \mathbf{u}_{ij}} E \frac{d}{dt} \mathbf{u}_{ij},$$

but

$$\frac{\partial}{\partial \mathbf{u}_{ij}} E = \frac{1}{4} A_{ij} (2 \mathbf{u}_{ij}) = \frac{1}{2} A_{ij} \mathbf{u}_{ij};$$

then

$$\frac{d}{dt} E = \frac{1}{2} \sum_{i} \sum_{j} A_{ij} \mathbf{u}_{ij} \frac{d}{dt} \mathbf{u}_{ij}$$

$$= \frac{1}{2} \sum_{i} \sum_{j} A_{ij} \left( \mathbf{x}_j - \mathbf{x}_i \right) \cdot \left( \frac{d}{dt} \mathbf{x}_j - \frac{d}{dt} \mathbf{x}_i \right).$$

However

$$\mathbf{x}_i \cdot \frac{d}{dt} \mathbf{x}_i = \alpha \frac{1}{k_i z_i} \sum_{q} A_{iq} (\mathbf{x}_i \cdot \mathbf{x}_q - \mathbf{x}_i \cdot \mathbf{x}_q) = 0;$$

then

$$\frac{d}{dt} E = - \sum_{i} \sum_{j} A_{ij} \mathbf{x}_j \cdot \frac{d}{dt} \mathbf{x}_i$$

$$= - \sum_{i} \sum_{j} \sum_{q} A_{ij} A_{iq} \frac{1}{k_i z_i} \left( \mathbf{x}_j \cdot \mathbf{x}_q - (\mathbf{x}_i \cdot \mathbf{x}_q) (\mathbf{x}_i \cdot \mathbf{x}_j) \right)$$

$$= \sum_{i} \sum_{j} \sum_{q} A_{ij} A_{iq} \frac{1}{k_i z_i} \left( \mathbf{x}_i \cdot \mathbf{x}_q \right) (\mathbf{x}_i \cdot \mathbf{x}_j)$$

$$- \sum_{i} \sum_{j} \sum_{q} A_{ij} A_{iq} \frac{1}{k_i z_i} \mathbf{x}_j \cdot \mathbf{x}_q$$

$$= \alpha \sum_{i} \sum_{j} \frac{1}{k_i z_i} A_{ij} (\mathbf{x}_i \cdot \mathbf{x}_j) \sum_{q} A_{iq} (\mathbf{x}_i \cdot \mathbf{x}_q)$$

$$- \alpha \sum_{i} \sum_{j} \frac{1}{k_i z_i} A_{ij} (\mathbf{x}_i \cdot \mathbf{x}_j) \sum_{q} A_{iq} \mathbf{x}_q$$

$$= \alpha \sum_{i} \sum_{j} \frac{1}{k_i z_i} A_{ij} (\mathbf{x}_i \cdot \mathbf{x}_j) \mathbf{x}_i \cdot \mathbf{x}_q$$

$$- \alpha \sum_{i} \sum_{j} \frac{1}{k_i z_i} A_{ij} (\mathbf{x}_i \cdot \mathbf{x}_j) \sum_{q} A_{iq} \mathbf{x}_q.$$  

Let $\mathbf{x}_{N_i} = \sum_{q} A_{iq} \mathbf{x}_q$ be the summation over all the velocities of the neighbors of particle $i$, and then
\[
\frac{dE}{dt} = \alpha \sum_{i} \frac{1}{k_i z_i} \sum_{j} A_{ij} (x_i \cdot x_j) (x_i \cdot x_{N_i}) - \alpha \sum_{i} \frac{1}{k_i z_i} \sum_{j} A_{ij} (x_j \cdot x_{N_i}) \\
= \alpha \sum_{i} \frac{1}{k_i z_i} (x_i \cdot x_{N_i}) x_i \cdot \sum_{j} A_{ij} x_j - \alpha \sum_{i} \frac{1}{k_i z_i} x_{N_i} \cdot \sum_{j} A_{ij} x_j \\
= \alpha \sum_{i} \frac{1}{k_i z_i} (x_i \cdot x_{N_i}) (x_i \cdot x_{N_i}) - \alpha \sum_{i} \frac{1}{k_i z_i} (x_{N_i} \cdot x_{N_i}). \tag{18}
\]

However, given any vectors \(a, c \in \mathbb{R}^D\), we have \(a \cdot c \leq ||a|| \cdot ||c||\) and \(a \cdot a = ||a||^2\), thus

\[
\frac{dE}{dt} = \alpha \sum_{i} \frac{1}{k_i z_i} \langle x_i \cdot x_{N_i} \rangle^2 - \alpha \sum_{i} \frac{1}{k_i z_i} \langle x_{N_i} \cdot x_{N_i} \rangle \\
\leq \alpha \sum_{i} \frac{1}{k_i z_i} (||x_i|| \cdot ||x_{N_i}||)^2 - \alpha \sum_{i} \frac{1}{k_i z_i} ||x_{N_i}||^2 = 0. \tag{19}
\]

**Remark 1** Perfect alignment most likely happens, since \(\frac{dE}{dt}\) is zero only if \(x_i\) and \(x_{N_i}\) are codirectional, that is, when all neighbors are either aligned or opposed to each other. While this condition does not hold, \(\frac{dE}{dt} < 0\), and the aligned state is asymptotically stable.

**4 Experimental results**

In this section, we present an extensive set of experimental results conducted on different classes of computer-generated and real-world networks.

**4.1 Illustrative example**

In this subsection, we present illustrative examples of the application of our method. Input networks are built by employing the following methodology:

1. Given a desired average vertex degree \(\langle k \rangle_{\text{des}}\), each vertex \(i\) randomly chooses \(\langle k \rangle_{\text{des}}/2\) vertices \(j\) to connect to. For each \(j\) to be selected, \(j\) is taken from the same community of \(i\) with a probability \(p_{\text{in}}\), or accordingly taken from a different community with a probability \(p_{\text{out}} = 1 - p_{\text{in}}\). The selection of the same vertex \(j\) twice or more is allowed, as it is also allowed for \(i\) to connect to itself. Also, \(j\) being selected by \(i\) does not prevent \(i\) being also selected by \(j\).

2. After establishing all the connections, the network is simplified, in the sense that loops (self-connections) are removed and multiple edges that connect the same pair of vertices become a single, undirected edge. As a result, the actual average degree \(\langle k \rangle\) may be reduced, but for large networks, it remains very close to the desired degree \(\langle k \rangle_{\text{des}}\).

For comparison purpose, in Fig. 1, we also show the results of applying CFG and Louvain algorithms on a network with four communities, average degree \(\langle k \rangle \approx 10\), and \(p_{\text{in}} = 0.66\). In this case, CFG achieves modularity \(Q = 0.33\) and Louvain, \(Q = 0.31\). We apply our method in this network, setting \(\alpha = 0.1\). In each round, we run the dynamical system until \(t = 100\) with ten independent initial configurations. The most (average) misaligned edge per round is removed until there is no more edges to remove. We choose the partition that yields the best modularity over different rounds. The results of the iterative edge-removal process are better than those of CFG and Louvain, reaching \(Q = 0.38\).

Figure 2 shows the modularity evolution over different rounds. Since communities are defined by connected components, the first edge-removal rounds just result in a single community, making the modularity score be very low. Those abrupt transitions reveal different components becoming completely disconnected from each other, i.e., they reveal those rounds in which the last edge that links two large components is removed. After achieving four major communities—and consequently the highest score—further removal starts destroying them. Such a local optimum partition is the one that should be returned by the proposed iterative edge-removal method.

To further illustrate our method, we provide another simulation. We show (in Sect. 3) that all the vertices in a connected component will align as time tends to infinity. As a result, one might wonder how our edge-removal strategy works in practice. To demonstrate its effectiveness, the method is applied on a simpler community-detection problem: detecting the 4 communities with the same size in a random clustered network that comprises 800 nodes with average degree \(\langle k \rangle = 9\). Connections are randomly assigned, such that every node has probability \(p_{\text{in}} = 0.9\) of connecting to another node in its community. We run our method with 3-dimensional velocity vectors and \(\alpha = 0.05\). For the sake of visualization, Fig. 3 depicts the evolution of the normalized velocity vectors (projected in two dimensions) in different situations. Line colors represent communities, and the transparency decreases in function of time \(t\). Plot (a) depicts 1000 iterations of our dynamical system in the original network. One can notice that all particles are aligned, but the velocity vectors of particles in different communities converge from different directions. This phenomenon is utterly important, since it enables us to distinguish the communities. It also explains the difference between our method and Kuramoto-based ones [3, 21]. In the synchronization-based techniques, each element usually is a fixed low-dimensional dynamical system. In the Kuramoto oscillator model, only a single real value is associated with each node, which corresponds to the phase. Thus, nodes
Fig. 1 Overview of the community-detection results. The input network has 4 communities, each one made of 200 vertices, $\langle k \rangle \approx 10$, and $p_{in} = 0.66$. The results of applying CFG (a) and Louvain (b) algorithms on the original network are shown. Communities are represented by different colors. In the last figure c, we display the results after applying 1766 rounds of the proposed iterative edge-removal process. In this case, communities are just different connected components. In each cycle, ten independent runs are performed and the most misaligned edge is removed. Only the remaining edges are shown in c. We set $\alpha = 0.1$.

can only synchronize “from two different directions”. Using three or more dimensions in our method, we bring an infinitude of possible directions. In the same figure, we also show three snapshots of the edge-removal process. After running up to $t = 1000$, we remove the 10 edges with highest values of misalignment coefficient. We repeat this process until 600 edges have been removed. The evolution of the normalized velocity vectors at $t = 1, 2, \ldots, 1000$ after removing 200, 400, and 600 edges are illustrated in subplots (b), (c), and (d), respectively. After the removal of 200 edges, we observe that one of the communities disconnects from the others, becoming a single connected component, and thus, the velocity vectors converge to a different point. With 400 edges removed, another community detaches. And finally, after 600 removals, each community becomes a connected component of the network, achieving our goal.

4.2 Analysis of the evolution of misalignment coefficient

In the previous section, we claimed that the misalignment coefficients of edges connecting distinct communities become usually higher than those connecting vertices inside the same community. Let us now present, in Fig. 4, how misalignment coefficients change over time. To reduce the dependence on the initial condition—random assignment of velocities—the population of misalignment coefficient values is obtained from ten independent runs. The input network is the same for all of these runs. The network has 800 vertices and $\langle k \rangle \approx 10$, what gives a total of approximately 4000 edges.

In the top plot of Fig. 4, we plot the average misalignment coefficient $H$ grouped by intra- and inter-edges. As we can see, coefficients fall down quickly. Intra-edges, however, align faster than the inter-edges. In the bottom plot, we show the ratio between intra- and inter-edges. As expected, a greater proportion of intra-edges have lower misalignment coefficient. Moreover, after many iterations, the velocity vectors become
inter- and intra-community edges

Bottom plot: ratio between the misalignment coefficient of intra- (dashed line) and inter-community (solid line) edges.

Measurements are taken from a single round. Top plot: average misalignment coefficient evolution separated between intra- and inter-community edges.

Results of removing a different number of edges per round are presented in this section. An evaluation index is used to objectively quantify the accuracy of the set of obtained communities. Specifically, we selected the adjusted Rand index (ARI), which is the corrected-for-chance version of the Rand index [13]. It measures the similarity between the partition obtained from some algorithm and a reference partition. ARI generates values between $−1$ and $1$. If two partitions match perfectly each other, it results in a value 1. On the other hand, it ensures a value close to 0 for a randomly labelled partition or a partition that assigns all the elements into a single group, given that the reference partition has more than one group, of course. Negative values stand for some anti-correlation between the pair of partitions.

Real-world applications, however, do not provide such a reference partition, so we also employ the modularity score. Unlike ARI, the modularity score does not make any assumption on a-priori knowledge of the network, e.g., vertex labels. By placing the ARI score against the modularity score, we can check for any relationship between them, i.e., check for a relationship between an information available in a real application (modularity) and a measure based on reference partitions (ARI).

First, let us consider the same network presented in Fig. 1, with $p_{in} = 0.66$. ARI and modularity scores along different rounds are presented in Fig. 5, arranged according to the number of edges removed per round.

Even when 40 edges ($\approx 1\%$ of the total number of edges) are removed per round, our method achieves higher ARI score than the CFG method: 0.64 (1 edge) and 0.60 (40 edges) against 0.45. Removing fewer edges per round is slightly better. Removing a larger fraction of the edges is less expensive though, for it demands fewer rounds to complete. However, we experience a significantly drop in the score if the fraction of removed edges per round is too big (around 2%).

Moreover, a desirable relationship is noticeable here: the highest modularity matches relatively high ARI scores. This result is useful for establishing when we should stop the edge-removal process and where the optimal round is, i.e., the round after which we are likely to find a good partition: stopping the process just before the modularity starts decreasing and taking such highest modularity’s network, using the connected components as the final set of communities. Although the results of Fig. 5 come from just one network instance, the same overall pattern is still present in other instances, even using different network param-
the same one depicted in Fig. 1: 4 communities, each one made of 200 vertices, the input network is exactly presented in this paper. Parameters, as we are going to see in the remaining results.

![ARI (solid lines) and modularity (dashed line) scores along different rounds.](image)

In Fig. 6, we present a comparison between removing 1, 40, and 80 edges per round on a population of 50 different networks. Each network has 4 communities, each one made of 200 vertices, (k) ≈ 10, and p_{in} = 0.66. Boxes in blue are the results of applying CFG and Louvain on the network. The remaining boxes (in red) are obtained by applying the proposed technique removing different amounts of edges. The partition that yields the best modularity is chosen. 10 runs per round are performed, and parameter α = 0.1 for all simulations. We stop the system at t = 100.

![Results of removing 1, 40, and 80 edges per round.](image)

4.4 Computational complexity

The edge-removal approach presented here is somewhat similar to the edge-betweenness-based algorithm proposed by Newman and Girvan [20]. The misalignment coefficient of an edge measures how different the incident vertices are in terms of their velocity vector. Edge betweenness, in turn, measures the relevance of an edge in terms of the number of shortest paths that include such an edge. Both concepts ultimately try to identify edges that connect distinct communities. An important drawback of the edge-betweenness approach is, however, its cubic time complexity. Precisely, for a network of m edges and n vertices, the time complexity is \( O(m^3n) \), or \( O(n^3) \) for sparse networks, in which case \( m \sim n \) [7, 8, 14, 20]. Therefore, it is necessary to have a discussion about the complexity of the approach proposed here.

Four variables are relevant to our time-complexity analysis: the number n of vertices; the number m of edges; the total amount of time-steps \( t_{\text{total}} \) per run; and the number of edge-removal rounds \( n_{\text{rounds}} \) required to obtain an appropriate partition of the network. The dimensionality of the velocity vectors is constant, and three dimensions are employed in our experiments, so it is not relevant to this analysis. Our hypothesis is that both \( t_{\text{total}} \) and \( n_{\text{rounds}} \) can be invariant no matter how big the network is.

In the case of removing just a single edge per round, \( n_{\text{rounds}} \) gets the order of m. However, the experiments presented in the previous section indicate that it is still possible to obtain satisfactory results by removing a fraction of all the edges per round, in which case \( n_{\text{rounds}} \) becomes constant. For instance, removing approximately 1% of the edges per round requires around 50 rounds to achieve good results. Also, concerning the number of runs per round, if one decides to perform a set of runs per round to reduce the influence of the initial random configuration, the number is still relatively small and does not scale with neither n nor m.

To provide evidence that supports our hypothesis—that neither \( t_{\text{total}} \) nor \( n_{\text{rounds}} \) scales with the network size—we present some results on bigger and denser networks. Particularly, a network that has ten times more vertices than those studied in the previous section and a network that has twice more edges while keeping the same number of vertices.

In Fig. 7, we present the evolution of the coefficients in the first round. We notice that increasing the net-
work: both of them with four communities and network, therefore, we can estimate that, for an arbitrarily large of time-steps needed to obtain satisfactory separation.

The exact number of edges removed per round is indicated between parentheses. For all runs, we set $\alpha = 0.1$ and the system runs until $t = 80$ for the dense network and $t = 250$ for the big network. The solid horizontal line indicates the ARI score obtained by the CFG method.

However, it is important to emphasize that the cost $O(m \log m)$ holds only from the asymptotic point of view, when networks become very large. For small- or medium-sized networks, the cost caused by $t_{\text{total}}$ and $n_{\text{rounds}}$ may become noticeable.

### 4.5 Experiments on unbalanced-communities networks

All the experiments presented so far are performed on a class of balanced networks, whose communities have the same size. To provide more-realistic examples, let us introduce now some results on networks whose communities have different number of vertices. These results are shown in Fig. 9.

We notice now a considerable drop in the quality of the partitions. Still, compared to traditional algorithms like CFG or Louvain applied to the original network, the proposed edge-removal process performs well. The performance of the proposed method, however, varies greatly. One possible explanation is that the fixed number of iterations $t = 250$ is not ideal. The study of heuristics for the system’s stop condition is left for future works.

### 4.6 Experiments on parameter dependency

An analysis of how $\alpha$ affects the resulting network partition is presented in Fig. 10. It clearly displays no significant change in both the ARI score and the modularity $Q$. As expected, however, the number of time-
Fig. 9 Results of removing 10, 40, and 80 edges per round on a population of 50 different networks. Each network has 4 communities, containing 800, 400, 200, and 100 vertices, ⟨k⟩≈ 10, and $p_{in} = 0.66$. Boxes in blue are the results of applying CFG and Louvain on the network. The remaining boxes (in red) are obtained by applying the proposed technique removing different amounts of edges. The partition that yields the best modularity is chosen. Ten runs per round are performed, and parameter $\alpha = 0.1$ for all simulations. We stop the system at $t = 250$ steps $t$ that the system needs to reach a stable configuration (out of the initial random assignment) highly depends on the choice of $\alpha$. The smaller the value of $\alpha$, the slower the vectors change. Therefore, we can conclude that the best choice is a reasonable large value of $\alpha$ that remains within the limit $\alpha < 0.5$, as discussed in Lemma 2.

Regarding the variability of results that can be seen in Fig. 10 for any given $\alpha$, it suggests that the proposed technique is sensitive to the initial random assignment of velocity vectors. However, as shown in Fig. 11, there is a clear correlation between the modularity $Q$ and the ARI score, especially for higher values of $Q$. Therefore, we can establish a simple strategy to obtain relatively good results: run the proposed technique multiple times independently and just select the partition that gives the best modularity.

One might wonder if running the dynamical system, Eq. 1, multiple times within each round could reduce the effects of initial configuration. Although that seems intuitive, the results presented in Fig. 12 do not show any significant variability reduction. Therefore, there is no clear advantage in running multiple simulations per round. It is much better to run the full set of rounds multiple times and select the partition based on modularity $Q$ as discussed before.

4.7 Experiments on Lancichinetti benchmark

Since real-world networks usually have heterogeneous degree distribution, we also apply our technique on the benchmark of Lancichinetti et al. [16]. Such benchmark produces networks in which both degree and community size distributions follow power law functions with arbitrary exponents. Motivated by typical values found in natural systems, we choose exponent 2 for the degree
distribution and 1 for the size of communities. Community sizes are in [3, 80]. The generated networks have a mixing parameter $\mu$ that controls the fraction of links between nodes of different communities. To assess our method, we use networks with 1000 nodes, average degree 10, and maximum degree 40. The mixing parameter $\mu$ varies between 0.1 and 0.6.

We compare our method against CFG and Louvain in 30 independent trials. The performance is measured in terms of the normalized mutual information index, which measures the similarity of the predicted partition against the expected one. Moreover, such index is suggested by the authors of the benchmark. CFG and Louvain have no parameter, while in our technique, we set $\alpha = 0.05$. For stopping criterion, we run the system until the maximum change in each projection $\hat{v}_{i,d}$ is less than $10^{-3}$ and remove the edge with higher misalignment coefficient. We interpret each connected component as a community. We repeat the edge-removal process until the modularity of the partitioning starts decreasing.

Figure 13 reveals that our method performs significantly better than CFG and Louvain. Beyond the mark $\mu = 0.5$, communities are no longer defined in a strong sense, since each node has more neighbors in other communities than in its own.

### 4.8 Experiments on real-world datasets

We apply our algorithm to four well-known real-world networks: Zachary’s karate club [29], the bottlenose dolphins social network [17], the American College Football [10], and the Krebs’ books on US politics.1

Table 1 presents the best modularity scores obtained by our method. We set $\alpha = 0.1$, 30 independent runs, and vary the fraction of removed edges in 1%, 2%, ..., 10%. Also, we vary the number of iterations $t \in [30, 70]$. We compare our results against three other bio-inspired optimization methods. (Missing results have not been measured by the authors of the original paper.) Also, the optimal value of modularity is calculated using an exhaustive search in all possible partitions. Modularity optimization is an NP-complete problem, and known algorithms have exponential time complexity [20].

---

1 This network can be found at http://www-personal.umich.edu/~mejn/netdata/.
Table 1 Modularity score of community-detection methods in four classic real-world datasets

| Method          | Dolphin | Football | Karate | Political books |
|-----------------|---------|----------|--------|-----------------|
| Proposed        | 0.529   | 0.605    | 0.419  | 0.527           |
| Amiri [1]       | 0.515   | 0.597    | 0.417  | 0.518           |
| Honghao [12]    | 0.529   | 0.605    | 0.420  | 0.527           |
| Song [25]       | –       | 0.531    | 0.362  | 0.463           |
| Optimal         | 0.529   | 0.605    | 0.420  | –               |

Optimal modularity is shown for comparison

were not able to find out the optimal partition of the political books network.

We observe that our method reached either optimal or nearly optimal modularity scores. Although the ant-colony-based technique [12] had similar performance, our algorithm has lower computational cost, because it explores the solution space in a greedy manner.

5 Conclusions

Throughout an extensive set of experiments presented in this paper, we see that the proposed flocking-like dynamical system and the iterative edge-removal process perform well in many scenarios. The decentralized, self-organizing dynamical model is robust, thus applicable on a wide variety of networks.

The concept of misalignment coefficient defined here is, in some sense, similar to the concept of edge-betweenness. High-misaligned edges are supposed to link distinct communities. However, the cost $O(n^3)$ (on sparse networks) of the method proposed by Newman and Girvan [20] can be prohibitive for its application in large networks. On the other hand, we claim that our edge-removal process is asymptotically quasilinear: $O(n \log n)$, which is quite attractive for large-network community detection.

In further works, we will study heuristics to find out good values of the number of iterations and removed edges per round. Moreover, an investigation on the parameter $D$ is required.

Acknowledgements This research work was supported by the State of São Paulo Research Foundation (FAPESP) (Projects 13/08666-8, 13/25876-6, and 15/50122-0), by the Brazilian National Research Council (CNPq) (Project 303012/2015-3), and by the German Research Foundation (DFG) (Project IRTG/GRK 1740). C4AI (Project FAPESP/IBM/USP: 2019/07665–4) Ministry of Science and Technology of China under grant number: G20200226016. R.A. Gueleri and F.A.N. Verri programmed and performed the computer experiments, as well as produced the resultants plots and figures. F.A.N. Verri performed the analytical study. All authors contributed on conceiving the method, planning the experiments, and writing this manuscript. L. Zhao, in addition, supervised all this research work.

Data Availability Statement My manuscript has no associated data or the data will not be deposited.

References

1. B. Amiri, L. Hossain, J.W. Crawford, R.T. Wigand, Community detection in complex networks: multi-objective enhanced firefly algorithm. Knowl. Based Syst. 46(Supplement C), 1–11 (2013)
2. A. Arenas, A. Díaz-Guilera, Synchronization and modularity in complex networks. Eur. Phys. J. Spec. Top. 143(1), 19–25 (2007)
3. A. Arenas, A. Díaz-Guilera, J. Kurths, Y. Moreno, C. Zhou, Synchronization in complex networks. Phys. Rep. 469(3), 93–153 (2008)
4. A. Arenas, A. Díaz-Guilera, C.J. Pérez-Vicente, Synchronization processes in complex networks. Phys. D Nonlinear Phenom. 224(1–2), 27–34 (2006)
5. V.D. Blondel, J.-L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks. J. Stat. Mech. Theory Exp. 10, 2008 (2008)
6. Q. Cai, M. Gong, S. Ruan, Q. Miao, H. Du, Network structural balance based on evolutionary multiobjective optimization: a two-step approach. IEEE Trans. Evol. Comput. 19(6), 903–916 (2015)
7. A. Clauset, M.E.J. Newman, C. Moore, Finding community structure in very large networks. Phys. Rev. E 70(6), 066111(2004)
8. S. Fortunato, Community detection in graphs. Phys. Rep. 486(3–5), 75–174 (2010)
9. Z.-K. Gao, Y.-X. Yang, P.-C. Fang, N.-D. Jin, C.-Y. Xia, L.-D. Hu, Multi-frequency complex network from time series for uncovering oil–water flow structure. Sci. Rep. 5, 8222 (2015)
10. M. Girvan, M.E.J. Newman, Community structure in social and biological networks. Proc. Natl. Acad. Sci. 99(12), 7821–7826 (2002)
11. E.M. Heemskerk, F.W. Takes, The corporate elite community structure of global capitalism. New Polit. Econ. 21(1), 90–118 (2016)
12. C. Honghao, F. Zuren, R. Zhigang, Community detection using ant colony optimization, in 2013 IEEE Congress on Evolutionary Computation (2013), pp. 3072–3078. https://doi.org/10.1109/CEC.2013.6557944
13. L. Hubert, P. Arabie, Comparing partitions. J. Classif. 2(1), 193–218 (1985)
14. A. Lancichinetti, S. Fortunato, Community detection algorithms: a comparative analysis. Phys. Rev. E 80(5), 056117 (2009)
15. A. Lancichinetti, S. Fortunato, Limits of modularity maximization in community detection. Phys. Rev. E 84(6), 066122 (2011)
16. A. Lancichinetti, S. Fortunato, F. Radicchi, Benchmark graphs for testing community detection algorithms. Phys. Rev. E 78(4), 046110 (2008)
17. D. Lusseau, K. Schneider, O. Boisseau, P. Haase, E. Slooten, S. Dawson, The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations. Behav. Ecol. Sociobiol. 54(4), 396–405 (2003)
18. P. Lv, J. Zhang, H. Zhang, A social network graphics segmentation algorithm based on community-detection, in 2016 2nd IEEE International Conference on Computer and Communications (ICCC) (2016), pp. 619–623. https://doi.org/10.1109/CompComm.2016.7924775
19. K.H. Nagai, Y. Sumino, R. Montagne, I.S. Aranson, H. Chaté, Collective motion of self-propelled particles with memory. Phys. Rev. Lett. 114(16), 168001(2015)
20. M.E.J. Newman, M. Girvan, Finding and evaluating community structure in networks. Phys. Rev. E 69(2), 026113 (2004)
21. E. Oh, K. Rho, H. Hong, B. Kahng, Modular synchronization in complex networks. Phys. Rev. E 72(4), 047101 (2005)
22. C. Pizzuti, Evolutionary computation for community detection in networks: a review. IEEE Trans. Evol. Comput. 22(3), 464–483 (2018). https://doi.org/10.1109/TEVC.2017.2737600
23. M.G. Quiles, E.E.N. Macau, N. Rubido, Dynamical detection of network communities. Sci. Rep. 6, 25570 (2016)
24. T.C. Silva, L. Zhao, Machine Learning in Complex Networks (Springer International Publishing, Berlin, 2016)
25. A. Song, M. Li, X. Ding, W. Cao, P. Ke, Community detection using discrete bat algorithm. Int. J. Comput. Sci. 1, 37–44 (2016)
26. C. Thiemann, F. Theis, D. Grady, R. Brune, D. Brockmann, The structure of borders in a small world. PLoS One 5(11), 1–7 (2010)
27. F.A.N. Verri, P.R. Urio, L. Zhao, Network unfolding map by vertex-edge dynamics modeling. IEEE Trans. Neural Netw. Learn. Syst. 29(2), 405–418 (2018). https://doi.org/10.1109/TNNLS.2016.2626341
28. T. Vicsek, A. Zafeiris, Collective motion. Phys. Rep. 517, 71–140 (2012)
29. W. Zachary, An information flow model for conflict and fission in small groups. J. Anthropol. Res. 33(4), 452–473 (1977)
30. C. Zhong, S.M. Arisona, X. Huang, M. Batty, G. Schmitt, Detecting the dynamics of urban structure through spatial network analysis. Int. J. Geogr. Inf. Sci. 28(11), 2178–2199 (2014)