Edinburgh Research Explorer

Core-periphery detection in hypergraphs

OPEN ACCESS

Core-periphery detection in hypergraphs[*]

Francesco Tudisco[†] and Desmond J. Higham[‡]

Abstract. Core-periphery detection is a key task in exploratory network analysis where one aims to find a core, a set of nodes well-connected internally and with the periphery, and a periphery, a set of nodes connected only (or mostly) with the core. In this work we propose a model of core-periphery for higher-order networks modeled as hypergraphs and we propose a method for computing a core-score vector that quantifies how close each node is to the core. In particular, we show that this method solves the corresponding non-convex core-periphery optimization problem globally to an arbitrary precision. This method turns out to coincide with the computation of the Perron eigenvector of a nonlinear hypergraph operator, suitably defined in term of the incidence matrix of the hypergraph, generalizing recently proposed centrality models for hypergraphs. We perform several experiments on synthetic and real-world hypergraphs showing that the proposed method outperforms alternative core-periphery detection algorithms, in particular those obtained by transferring established graph methods to the hypergraph setting via clique expansion.

Key words. hypergraph partitioning, nonlinear Laplacian, Perron-Frobenius, power method, core-periphery

AMS subject classifications. 65F30, 05C65, 65F15

1. Introduction. Finding core-periphery structures in networks represented as graphs is an important task in exploratory networks analysis [14, 19, 41, 50]. Core-periphery structure has been detected and interpreted in many complex systems, including protein-protein interaction networks [30], metabolic and gene regulatory networks [43], social networks [7, 14], engineered networks such as the Internet, power-grids or transportation networks [50], and economic networks [46]. See also the review [18].

In a graph, a core set is defined as a set of nodes which has many connections both internally and outside the set, while the periphery is a set of nodes that only (or mostly) connects to the core. Partitioning the graph into core and periphery is reminiscent of other graph partitioning problems, the most popular being graph clustering, where one seeks two sets that are only (or mostly) connected internally. While this formulation of core-periphery detection is a binary classification problem, real-world complex networks modeled as graphs rarely allow a clear-cut core-periphery partitioning of the nodes; more frequently we can expect a 'smooth' transition between core and periphery (see e.g. [41, 50]). Mathematically, this translates into the problem of assigning a core-periphery score (or simply 'core-score') to the nodes of the graph, which indicates to what extent each node is peripheral or core. From this point of view, core-periphery detection can be interpreted as the problem of finding the most 'central' nodes in a graph and, in fact, centrality-based core-periphery detection approaches have been considered, see e.g. [18, 20, 34].

[†]School of Mathematics, Gran Sasso Science Institute, 67100 L'Aquila, Italy (francesco.tudisco@gssi.it)

[‡]School of Mathematics, University of Edinburgh, EH93FD Edinburgh, UK (d.j.higham@ed.ac.uk)

Recent years have seen a growth in interest towards hypergraphs and in general higher-order graph models that directly account for multiple node interactions that take place simultaneously, see e.g. [9, 12, 13, 47]. Moving from a graph to a hypergraph allows us to retain more information in many natural, social and data systems, including email exchanges, group messaging, meeting attendance, document coauthorship and supermarket baskets. However, this richer framework presents new challenges in terms of defining appropriate concepts and in the design and analysis of efficient computational algorithms.

If we are interested in separating two node sets by measuring the way these sets interact in a hypergraph, a multitude of definitions are possible. In the context of node clustering, for example, different notions of cuts can be considered leading to different hypergraph cut algorithms. While the cut between two node sets S and T in the graph case is uniquely measured as the number (or the sum of the weights) of the edges connecting S and T, in the hypergraph setting one has several choices. A commonly adopted and successful definition, sometimes referred to as 'all-or-nothing' hypergraph cut, measures the cut by counting how many hyperedges have at least one node in S and one node in T [27]. Another relatively standard cut function measures the cut in the projected graph obtained replacing each hyperedge with a clique (i.e. the clique-expanded graph) [55]. Other approaches propose different ways to weight the proportion of nodes in S and T per each hyperedge [54].

Similarly, in the core-periphery context, the number of hyperedges between core and periphery can be counted in several ways. As for the hypergraph cut, one may count hyperedges as showing a core-periphery behaviour if at least one node in the hyperedge is in the core. This is the approach we use in this work, were we additionally weight each hyperedge in terms of the number of nodes it contains (which allows us to e.g. penalize very large hyperedges). In a way, our approach is the core-periphery analog of the 'all-or-nothing' hypergraph cut function. However, unlike the hypergraph cut problem, we show that the corresponding non-convex core-periphery optimization problem can be solved globally to an arbitrary precision via a nonlinear eigenvector approach. While a similar globally convergent method is available for graphs [50], we find that direct use of the graph method applied to the clique-expanded graph leads to inefficient core-periphery detection in hypergraphs. This is in stark contrast to the cut setting, where hypergraph cut functions based on the clique-expanded graph typically show decent performance.

Our method computes a bespoke core-score vector for hypergraphs as the positive solution of an eigenvalue problem for a suitable hypergraph Laplacian-like operator $L(x) := Bg(B^\top f(x))$, where B is the hypergraph incidence matrix and f, g are entrywise nonlinearities. It is particularly interesting that this type of nonlinear Laplacian operator appears in many settings. For example, in the graph case, if $f = \mathrm{id}$ and $g(x) = |x|^{p-1}\mathrm{sign}(x)$, then L boils down to the graph p-Laplacian operator [16, 23, 42, 52]. Exponential- and logarithmic-based choices of f and g give rise to nonlinear Laplacians used to model chemical reactions [37, 53] as well as to model consensus dynamics and opinion formation in hypergraphs [35]. Trigonometric functions such as $g(x) = \sin(x)$ are used to model network oscillators [8, 33, 44]. Entrywise powers and generalized (power) means are used for node classification [5, 29, 36, 48], network centrality and clustering coefficients [4].

In particular, in [48] this type of hypergraph mapping is used to define generalized eigenvector centrality scores for hypergraphs which include as special cases hypergraph centralities

based on tensor eigenvectors [10]. Thus, the proposed hypergraph core-periphery score can be interpreted as a particular hypergraph centrality designed specifically for core-periphery problems and gives mathematical support to the intuition that centrality measures for hypergraphs may be an indication of core and periphery [3].

The paper is structured as follows. In §2 we motivate the core-periphery concept. §3 introduces a random model for generating hypergraphs with core-periphery structure and relates the model to a maximum likelihood problem. This is used to justify the optimization problem that we propose and analyse in §4. In §5 we briefly discuss relevant work in the graph setting, and in particular we review how the clique expansion can be used as a means to approximate a hypergraph with a graph. Computational experiments are presented in §6 and concluding remarks are given in §7. Appendix A contains a proof of our underpinning theoretical result.

The main contributions of this work are

- the optimization formulation (4.1) for core-periphery detection at the higher-order hypergraph level, motivated by the generative random hypergraph model (3.1) (see Theorem 3.1),
- existence and uniqueness theory for this optimization problem, and a practical, globally convergent iteration scheme (Theorem 4.1),
- an interpretation of this approach as a nonlinear spectral method (Corollary 4.2),
- a core-periphery profile definition for hypergraphs, allowing methods to be compared on real data sets (§6.1).

2. Motivation. We consider a hypergraph $H = (V, E)$ with $|V| = n$ nodes and $|E| = m$ hyperedges. We assume that the nodes and hyperedges are labelled from 1 to n and from 1 to m, respectively. So, each hyperedge $e \in E$ has the form $(i_1, \ldots, i_r)$, where $1 \leq i_1 < i_2 < \cdots < i_r \leq n$, and we refer to r, the number of nodes present, as the size of the hyperedge. We let $B \in \mathbb{R}^{n \times m}$ denote the corresponding incidence matrix, so $B_{ij} = 1$ if node i is present in hyperedge j, and $B_{ij} = 0$ otherwise.

We are interested in the case where interactions can be described via a core-periphery mechanism, and we seek an algorithm that can uncover this structure when it is present in the data. Loosely, core nodes are those that enable interactions, whereas peripheral nodes may only take part in an interaction if at least one core node is also present. Suppose first that the nodes may be split into disjoint subsets: the core and the periphery. In the strictest interpretation, we could argue that the resulting core-periphery hypergraph will consist of precisely those hyperedges that contain at least one core node. More generally, we may argue that a hyperedge is more likely to arise if it contains at least one core node. Moreover, we may extend the idea further to argue that a hyperedge (of a given size) is more likely to arise if it involves more core nodes. A second direction in which we can relax this definition is to argue that instead of a binary split, the nodes can be ranked in terms of their coreness.

Using these ideas, which are discussed further in §5, we build on notions that have been proposed and tested in the standard network context to motivate the new model and algorithm.

3. Core-periphery hypergraph random model. We now propose a model that generates hypergraphs with a planted core-periphery structure, generalizing the logistic core-periphery random model for graphs introduced in [50]. Rather than a binary classification model where
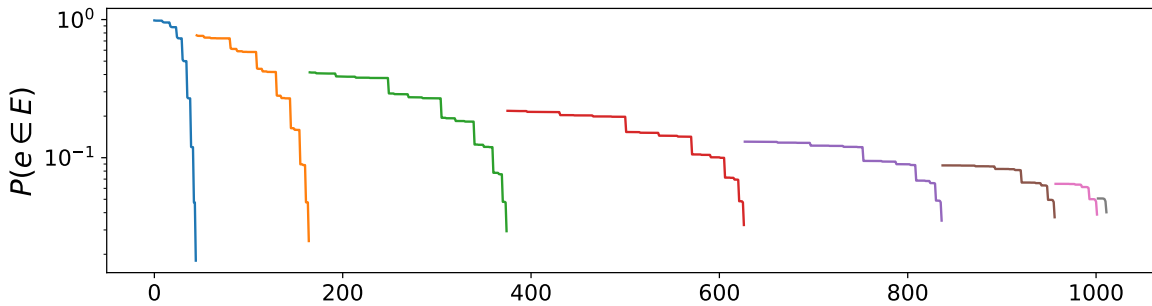
Figure 1: Hyperedge probability distribution $P(e \in E)$ as defined in (3.1), for a random hypergraph with $n = 10$ nodes and $2^n - (n+1) = 1013$ possible hyperedges. Here, $\xi(e) = 1/|e|$ and $q = 10$.

a node is first assigned to the core or periphery, the proposed random model generates hypergraphs where a smooth transition between the core and the periphery sets is allowed. As discussed in [50], we find this situation more realistic, as real-world interactions with a perfectly clear cut separation between core and periphery sets are rare.

We assume each node is assigned an integer index $i \in \{1, \dots, n\}$ with the convention that the planted core structure starts 'at the top', i.e., we want nodes with smaller indices to be in the core and those with large indices to be peripheral. Let $\sigma(\cdot)$ denote the sigmoid function, so that $\sigma(x) = 1/(1 + e^{-x})$. We introduce and study here the generative model with a fixed number of nodes and a fixed upper limit on the maximum hyperedge size, where each hyperedge e exists with independent probability

$$(3.1) \qquad\qquad P(e \in E) = \sigma\big(\xi(e)\mu(e)\big).$$

Here $\xi(e)$ is a function that decreases with the size of the hyperedge, for example $\xi(e) = 1/|e|$, and $\mu(e)$ is a function that attains large values if the nodes in e are near the core, i.e. they have small indices. For $q \geq 1$, an example choice of μ is

$$\mu(e) = \mu_q(e) := \Big(\sum_{i \in e} \Big(\frac{n-i}{n} \Big)^q \Big)^{1/q},$$

which is a smooth approximation of $\mu(e) = \max_{i \in e} \frac{n-i}{n}$, obtained for $q \to \infty$. Notice that μ_q corresponds to the q-norm of the vector with entries $(1 - 1/n, 1 - 2/n, \dots)$ restricted to the hyperedge e. We will often consider such μ_q in practice as it is directly connected to the optimization framework described in §4.

With this choice of μ and ξ, a hyperedge is more likely to exist when it contains a small number of nodes, at least one of which is part of the core. Figure 1 illustrates the behaviour of P as a function of the edge e on a random hypergraph with $n = 10$ nodes and $2^n - (n+1) = 1013$ possible hyperedges given by all possible subsets e of $\{1, \dots, n\}$ with $|e| \geq 2$. In the figure,

the hyperedges on the x-axis are sorted in lexicographical order and grouped by their size:

$$\{1,2\} \quad \{1,3\} \quad \cdots \quad \{1,n\} \quad \{2,3\} \quad \{2,4\} \quad \cdots \quad \{2,n\} \qquad \{1,2,3\} \quad \{1,2,4\} \quad \cdots \quad \{1,2,n\} \quad \{1,3,4\} \quad \{1,3,5\} \quad \cdots \quad \{1,3,n\} \quad \cdots$$

We use different colors to distinguish between edge size.

We now turn to the inverse problem where it is required to discover a node ordering that reveals core-periphery structure. We will let $\mathcal{P}$ denote the set of all permutations of $\{1, 2, \ldots, n\}$. So $v \in \mathcal{P}$ is a vector $v \in \mathbb{R}^n$ with distinct integer elements between 1 and n. We will associate $v \in \mathcal{P}$ with a node reordering such that node i is mapped to node v_i. Hence, if $v_j = 1$ and $v_k = n$, then, according to this new ordering, node j is the most core and node k is the most peripheral.

In this framework, given a hypergraph, it is reasonable to choose an ordering that maximizes the likelihood of the hypergraph arising under the model (3.1). This viewpoint has been found useful for various graph models and structures [25, 26]. In the next theorem, we show that the resulting maximum likelihood problem can be converted into a discrete optimization problem that turns out to be amenable to relaxation.

Theorem 3.1. *For a given hypergraph, a permutation vector $v \in \mathcal{P}$ corresponds to a maximum likelihood node reordering under the model (3.1) if and only if it maximizes the objective function $\sum_{e \in E} \xi(e)\mu(e)$.*

Proof. Under the model (3.1), the likelihood is

$$\prod_{e \in E} \sigma\big(\xi(e)\mu(e)\big) \prod_{e \in E'} \big(1 - \sigma\big(\xi(e)\mu(e)\big)\big),$$

where E' denotes the complement of E; that is, the hyperedges that are not present. We may rewrite this likelihood as

$$\prod_{e \in E} \frac{\sigma\big(\xi(e)\mu(e)\big)}{1 - \sigma\big(\xi(e)\mu(e)\big)} \prod_{\text{all edges}} \big(1 - \sigma\big(\xi(e)\mu(e)\big)\big).$$

Now, the second product is independent of the node ordering. So we solve the problem by maximizing the first product, which may be written

$$\prod_{e \in E} e^{\xi(e)\mu(e)}.$$

Finally, taking a logarithm shows that the original problem is equivalent to maximizing $\sum_{e \in E} \xi(e)\mu(e)$. ∎

4. Core-periphery detection via cost function optimization. Motivated by Theorem 3.1, we propose a model based on the optimization of a nonconvex core-periphery cost function. Our goal is to determine a core-periphery nonnegative score vector $x^\star$ that assigns large value to nodes in the core and small values to those in the periphery. Clearly, such a vector is 'scale invariant' in the sense that any positive rescaling of a core-periphery score vector

corresponds to the same core-periphery assignment. For this reason, given a hypergraph $H = (V, E)$, we formulate the core-periphery detection problem as the following norm-constrained optimization problem:

$$(4.1) \qquad \max_x f(x) \quad \text{s.t. } \|x\|_p = 1 \text{ and } x \succeq 0, \quad \text{where} \quad f(x) = \sum_{e \in E} \xi(e) \, \|x|_e\|_q \, ,$$

$x \succeq 0$ denotes a vector with nonnegative entries, and $x|_e$ denotes the restriction of x to the nodes in the hyperedge e, i.e. $x|_e = (x_{i_1}, \ldots, x_{i_r})$ if $e = (i_1, \ldots, i_r)$. As before, $\xi(e)$ is a function that assigns a weight to the hyperedges in H. In practice, ξ may be both 'data-driven', in the sense that it may incorporate a weight $w(e)$ of the hyperedge in the input dataset, and 'model-oriented', in the sense that it should decrease with the size of the hyperedges (e.g., $\xi(e)$ proportional to $1/|e|$) so to take into account our modeling assumption that, in the hypergraph core-periphery model, hyperedges with more nodes should make a smaller contribution to the assignment of the core-score.

Note that, as $x \succeq 0$, when q is large we have $\|x|_e\|_q \approx \max_{i \in e} x_i$ and thus $f(x)$ is large if many hyperedges contain at least one node with large core-periphery score value. This may be interpreted as the core-periphery analog of the widely used 'all-or-nothing' definition of hypergraph cut function in the context of hypergraph clustering and, in general, of cut-based hypergraph problems [27, 54]. However, similarly to the hypergraph cut setting, when considering a core-periphery score one may want to account for the fact that hyperedges may contain more than one core node, and give these hyperedges a greater importance. We note that this is somewhat automatically obtained by our choice of smooth function f. In fact, while in the graph setting the non-smooth limit case $q \to \infty$ is to be preferred [48] as each edge contains exactly either one, two or no core nodes, we argue that large but finite values of q are better suited to hypergraphs. In fact, when $1 \ll q < \infty$, the cost function $f(x)$ naturally handles possible ambiguity due to the presence of hyperedges with more than two nodes in the core: While the infinity norm $\|x|_e\|_\infty$ is large if there is at least one core node in e but ignores the presence of a larger number of core nodes, $\|x|_e\|_q$ (for large but finite q), is large when there is at least one core node in e but grows when the hyperedge contains a larger number of such nodes.

Although (4.1) is a nonconvex optimization problem, we show below that if we are interested in entry-wise positive solutions, then it admits a unique solution which we can always compute to an arbitrary accuracy via a linearly convergent method, provided that $p > q$. We move the relatively long proof of this result to Appendinx A.

Theorem 4.1. *If $p > q > 1$ then* (4.1) *has a unique entry-wise positive solution $x^\star$. Moreover, for any entrywise positive starting vector, the iterative scheme*

- $y \leftarrow \text{Diag}(x)^{q-1} B \Xi (B^\top x^q)^{\frac{1}{q}-1}$
- $x \leftarrow (y/\|y\|_{p^*})^{\frac{1}{p-1}}$

where $p^* = p/(p-1)$ is the Hölder conjugate of p and where Ξ is the diagonal $|E| \times |E|$ matrix with diagonal values $\xi(e)$, converges to $x^\star$ with the linear rate of convergence $O(|q-1|/|p-1|)$.

It is interesting to note that the vector computed in this way can be interpreted as a 'nonlinear eigenvector centrality' for the nodes in the hypergraph, as per the model introduced in [4, 51]. In fact, the following direct corollary of the previous result shows that the core score

$x^\star \succ 0$ solution to (4.1) coincides with an eigenvector of a nonlinear hypergraph Laplacian operator of the form $L(x) = B\Xi g(B^\top f(x))$, for particular nonlinear choices of f and g.

Corollary 4.2. *Let $x^\star \succ 0$ be the unique solution of* (4.1). *Then, $x^\star$ is the unique nonnegative eigenvector of the nonlinear eigenvector problem*

$$B\Xi g(B^\top f(x)) = \lambda x \tag{4.2}$$

with $g(x) = x^{-1+1/q}$ and $f(x) = x^{q/(p-q)}$.

Proof. Let $F(x) = \mathrm{Diag}(x)^{q-1} B\Xi (B^\top x^q)^{\frac{1}{q}-1}$. Assume $x \succ 0$. From the fixed point identity $x = (F(x)/\|F(x)\|_{p^*})^{1/(p-1)}$ we have $F(x)^{1/(p-1)} = \|F(x)\|_{p^*}^{1/(p-1)} x$. Multiplying this identity entrywise by $x^{(1-q)/(p-1)}$ on the left and then taking the $(p-1)$-th of power both sides, we get

$$B\Xi (B^\top x^q)^{\frac{1}{q}-1} = \|F(x)\|_{p^*} x^{p-q}.$$

Finally, the change of variable $x \mapsto x^{1/(p-q)}$ shows that $x^\star \succ 0$ is the limit of the iterative scheme in Theorem 4.1 if and only if $x^\star$ is such that $B\Xi g(B^\top f(x^\star)) = \lambda x^\star$ with $g(x) = x^{1/q-1}$, $f(x) = x^{q/(p-q)}$ and $\lambda = \|F((x^\star)^{1/(p-q)})\|_{p^*} > 0$. ∎

Before moving on, we briefly point out how Theorem 4.1 compares with the main theorem in [51] in view of the corollary above. A direct consequence of Corollary 4.2 combined with [51, Thm. 2.3] shows that the nonlinear power method proposed in [51] for general nonlinear singular value problems, can be used to compute a solution to (4.2), provided $|p - q| \geq |q - 1|$ and the bipartite graph representation of the underlying hypergraph is connected. Note that the condition on p and q in this case boils down to $p \geq 2q - 1$, when $p, q > 1$. Using a different argument, Theorem 4.1 shows that for the particular choice of g and f which correspond to the core-periphery optimization problem (4.1), the less stringent condition $p > q$ is enough to ensure convergence of the proposed fixed point iteration to the solution of (4.2), without any requirement on the topology of the hypergraph.

5. Comparison with the graph setting. Following the seminal work by Borgatti and Everett [14], over the years several models for core-periphery detection on graphs have been developed, including methods based on degree and eigenvector centralities [34, 41], rank-1 approximations [15] and the optimization of a core quality fictional [41]. As observed in [50], several of these methods can be cast as the optimization of a core-periphery kernel function similar to (4.1) and a competitive core-periphery detection method available for graphs is obtained there by means of a nonlinear spectral method, which corresponds to the graph version of the method we propose here. All these methods can be directly applied to hypergraphs after a 'flattening' or 'projection' step, where the whole higher-order graph is approximated by a standard graph. A widely-used projection approach is the so-called (linear) 'clique-expansion' [1, 17, 28, 38, 39, 40, 55], where hyperedges in H are replaced by cliques in the flattened graph G_H, whose adjacency matrix A_H therefore becomes

$$(A_H)_{ij} = \sum_{e:\, i,j \in e} w(e) \tag{5.1}$$

with $w(e)$ the weights of the original hypergraph. While this is perhaps the most popular projection method, other approaches are possible, including clique averaging [2], where the weights $w(e)$ in the sum (5.1) are averaged with generalized mean functions, connectivity graph expansion [6, 21], where the weights in the clique expansion are based on hyperedge degrees, for example replacing $w(e)$ with $1/(|e| - 1)$ in (5.1), and the star expansion [56], where the flattened graph is obtained by introducing new vertices for each hyperedge, which are then connected according to the hypergraph structure.

However, graph core-periphery detection methods applied to a projected hypergraph may lead to poor core-periphery assignments on the original hypergraph. For example, when applied to the clique-expanded graph, the nonlinear spectral method proposed in [50] computes the global optimizer of

$$\sum_{ij} (A_H)_{ij}(|x_i|^q + |x_j|^q)^{1/q} = \sum_{e \in E} \left\{ \sum_{i,j \in e} w(e)(|x_i|^q + |x_j|^q)^{1/q} \right\}.$$

While this objective function reduces to the proposed hypergraph optimization problem (4.1) when all hyperedges contain exactly two nodes, the two optimization problems are significantly different in the general case. In particular, the hypergraph flattening loses track of the original hyperedges and thus only measures core and periphery structure in a pairwise fashion. Hence, approaches based on hypergraph flattening may fail to assign a correct score to the nodes even in simple hypergraph examples, as shown in the next section.

6. Experiments. In this section we perform experiments on a range of hypergraphs in order to validate the performance of the proposed nonlinear eigenvector method for the optimization of (4.1), which we will denote as `HyperNSM`. In all our experiments we use the scaling function $\xi(e) = 1/|e|$ and we choose $q = 10$ and $p = 11$ in (4.1). For a hyperedge weight $w(e) > 0$, we set $\xi(e) = w(e)/|e|$.

We compare our method with a number of existing approaches. One is the Union of Minimal Hitting Sets (`UMHS`) method by Amburg et al. [3], which is a greedy method designed to recover planted hitting sets in hypergraphs. The method is inherently local and thus requires several random initializations out of which the best assignment is selected. In our experiments we perform five random restarts due to time limitations. Two additional baselines are core-periphery detection algorithms for graphs, applied to the clique-expanded graph in (5.1):

1. `Borgatti-Everett`: The method by Borgatti and Everett [14], which is the pioneering approach for core periphery detection in graphs
2. `GraphNSM`: The nonlinear spectral method by Tudisco and Higham [50], which was shown to be a highly competitive core-periphery detection method for graphs.

6.1. Hypergraph core-periphery profile. In order to evaluate the quality of the core-periphery assignment we introduce a generalization of the core-periphery profile for graphs [22, 49] which we define as follows. For any subset of nodes $S \subseteq V$ consider the quantity

$$\gamma(S) = \frac{\# \text{ edges all contained in } S}{\# \text{ edges with at least one node in } S},$$

or its weighted version:

$$\gamma(S) = \Big(\sum_{e:\, e \subseteq S} \xi(e) \Big) \Big(\sum_{e:\, S \cap e \neq \emptyset} \xi(e) \Big)^{-1}.$$

Given a core-periphery score vector $x \succeq 0$ with distinct entries the *hypergraph core-periphery profile* is the function $\gamma(k)$ that to any $k \in \{1, \ldots, n\}$ associates the value $\gamma(S_k(x))$ where $S_k(x)$ is the set of k nodes with smallest core-periphery score in x. Given its definition, $\gamma(S)$ is small if S is largely contained in the periphery of the hypergraph. Thus a hypergraph has a strong core–periphery structure revealed by a core–score vector x if the corresponding profile $\gamma(k)$ attains small values as k increases from one, and then grows drastically as k crosses some threshold value k_0, which indicates that the nodes in $V \setminus S_{k_0}(x)$ form the core.

6.2. Hyperplane and hypercycle results.
Consider the hypergraph in the left and center panels of Figure 2a. Because of its airplane outline, we will refer to this hypergraph as a 'hyperplane'. Intuitively the clique-expanded approach will not work here as the large 'wing' hyperedge gives rise to a fully connected subgraph in G_H which would correspond to a core set in the projected graph, which is not present in the original hypergraph. This is confirmed by the central plot in Figure 2a, where the nodes in the hypergraph are shaded according to the core score computed with `GraphNSM` on the clique-expanded graph. By contrast, `HyperNSM` gives a more satisfactory result where each hyperedge has exactly one of the top-two core nodes, and the non-overlapping wing nodes are regarded as peripheral (left panel in Figure 2a). This difference is also reflected in the rightmost panel of Figure 2a where we compute the hypergraph core-periphery profiles introduced in §6.1. The score assigned by `HyperNSM` identifies a two-node core, while no hypergraph core-periphery structure seems to be captured by the graph method applied to the clique-expansion.

We observe similar behaviour on the 'hypercycle' hypergraph shown in Figure 2b. Here, each hyperedge shares exactly two nodes with exactly two other hyperedges, in a periodic fashion, and we have five hyperedges of size 3, 4, 5, 6 and 15. As one hyperedge is much larger than the others, the clique expansion approach assigns all of its nodes to the core. `HyperNSM` instead assigns high coreness to each of the 5 'overlap' nodes. In the right panel we see that the hypergraph core-periphery profile starts to increase when the first two overlap nodes are included, since every hyperedge contains exactly two of these nodes.

6.3. Real-world datasets with planted core.
We consider here two real-world hypergraph datasets—*W3C* and *Enron*—with a planted core set that arises directly from the data collection process, as discussed in [3]. Both the datasets are email hypergraphs, in which nodes are email addresses. Each hyperedge records a set of email addresses that appear on the same email. Table 1 reports a summary of statistics of the two datasets.

These datasets are characterized by a planted core, which consists of a 'hitting set' of nodes such that every hyperedge contains at least one of these nodes. (In practice, the data was collected by examining the email accounts of the hitting set.) The greedy algorithm `UMHS`, based on the union of minimal hitting sets, was designed in [3] with the aim of recovering a planted core of this type. Even though this definition of core differs from the one we consider in this paper, the experiments illustrated in Figures 3 and 4 show that the performance of `HyperNSM` is on par with that of `UMHS` (with five random restarts).
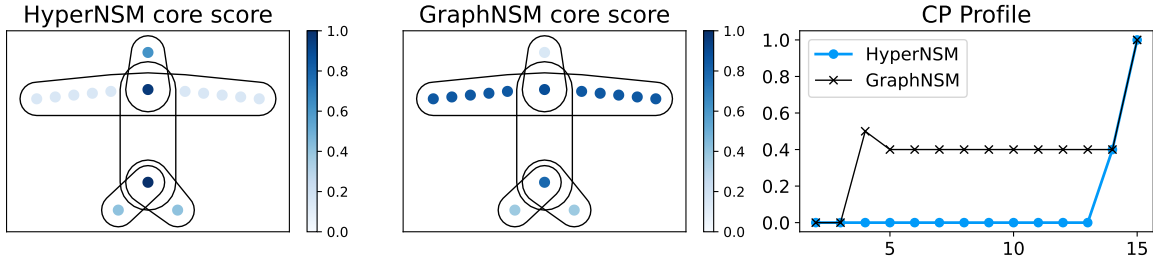
(a) Hyperplane with five hyperedges and 15 nodes.
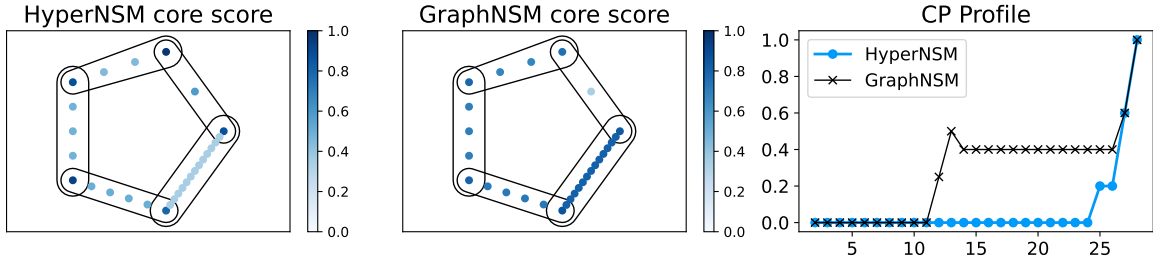
(b) Hypercycle with five hyperedges and 28 nodes.

Figure 2: Left and central panels: hypergraph drawing with nodes colored according to the core score obtained by means of the proposed `HyperNSM` approach (left) and the purely graph method `GraphNSM` (center). Right panel: hypergraph core-periphery profile corresponding to the two core score assignments, i.e. $\gamma_x(k)$ as defined in §6.1, plotted as a function of k, for the two core score vectors x.

| | #nodes | #edges | #nodes in core | $w(e)$ | | | $|e|$ | | |
				max	mean	std	max	mean	std
Enron	12722	5734	132	419	2.73	9.07	25	5.25	5.1
W3C	15458	19821	1509	1	1	0	25	2.22	0.98

Table 1: Basic statistics for the two email datasets with planted hitting set

Precisely, Figure 3 shows the core-periphery profiles corresponding to the core score computed with the four methods `HyperNSM`, `GraphNSM`, `Borgatti-Everett` and `UMHS`. As `UMHS` is originally designed to output a ordered list of nodes such that the nodes at the top of the list belong to the core, we treat this list as a ranking and use the corresponding ordering as a discrete core-score vector for `UMHS`. The core-periphery profiles for `HyperNSM` and `UMHS` are visually indistinguishable in both cases.
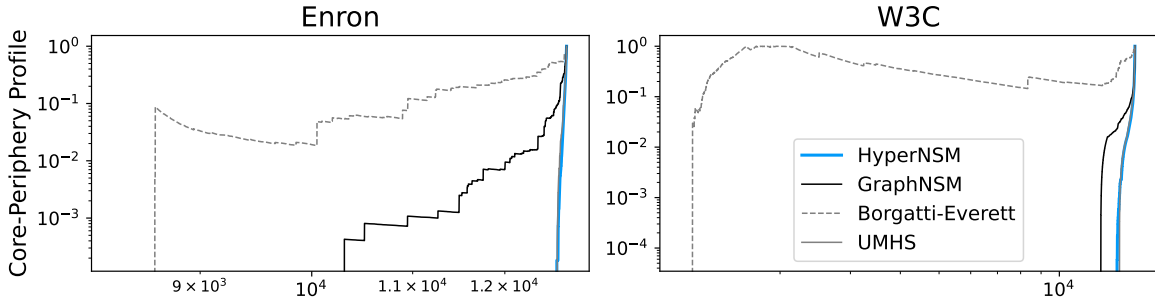
Figure 3: Core-periphery profile for different core-periphery detection methods. The core-periphery profiles for `HyperNSM` and `UMHS` are visually indistinguishable in both cases.
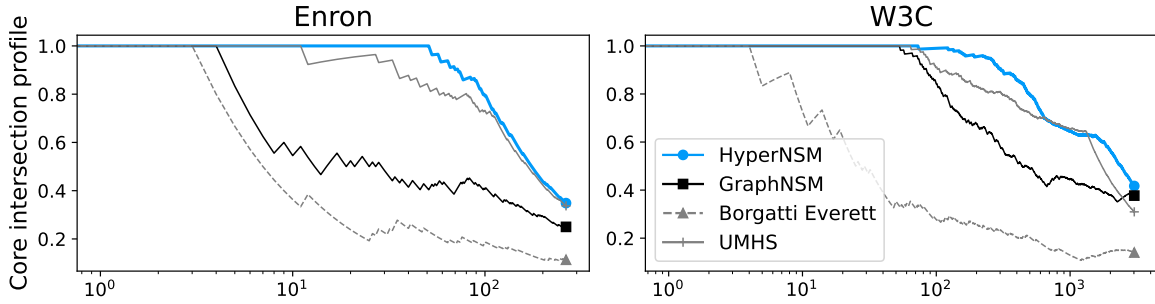

Figure 4: Intersection between the top k nodes as ranked by different core-periphery methods and the planted core set, for increasing values of k (x-axis).

Figure 4, instead, shows the 'core intersection profile', which is computed as follows. Given the planted core set C, for any set S let

$$\iota(S) = |S \cap C|/|S|\,.$$

Similarly to the core-periphery profile, the core intersection profile corresponding to a vector $x \succeq 0$ is the function $\iota_x(k)$ which to any k associates the value $\iota(S_k(x))$ where $S_k(x)$ is the set of top k nodes according to x. Thus, the core-score vector x well-captures the planted core C if its corresponding core intersection profile remains ≈ 1 as k is increased up to the size of the planted core.

While having comparable performance to `HyperNSM` on these test datasets, `UMHS` is significantly more expensive than the other algorithms, as shown by the computational times in Table 2. For `UMHS` we report the execution time for one run as implemented in [3], where the method finds a hitting set processing the hyperedges in a random order, then prunes the set to reduce it to a minimal hitting set. For the other methods, we show the time elapsed for the relative norm of the difference of two consecutive iterates to reach the tolerance `1e-8`. In terms of computational complexity, the cost of each iteration of `HyperNSM` is dominated

	HyperNSM	GraphNSM	Borgatti-Everett	UMHS
Enron	1.61	4.99	0.33	29.63
W3C	3.32	3.74	0.04	1269.17

Table 2: Execution time (sec) for different methods on the two email datasets from §6.3. The table shows time for one run of `UMHS`. The other methods are run until the relative norm of the difference of two consecutive iterates is smaller than `1e-8`.

by the matrix vector products Bv and $B^\top v$, which are linear in the number of input data, i.e., the cost of each step is $O(\sum_e |e|)$. Thus, for sparse hypergraphs, the method is fast. For `GraphNSM`, the cost of each iteration is dominated by the matrix vector product $A_H v$. As the number of nonzero entries in A_H is $O(\sum_e |e|)$, `GraphNSM` has comparable computational complexity to `HyperNSM`. However, `GraphNSM` also requires A_H to be formed from H, which may be expensive and memory demanding when very large hyperedges are present. The cost per iteration of `Borgatti-Everett` is also linear in the number of nonzeros of A_H but, as this is a purely-linear iteration (no entriwise powers are required) it is in practice faster than the nonlinear spectral iterative counterparts.

We also note that `UMHS` is based on the strict definition that the core nodes form a hitting set; here every hyperedge must contain at least one core node. For this reason, the algorithm is not well-suited to more general data sets where there is not a perfect planted core. `HyperNSM` is designed to tolerate spurious or noisy information—note that the model (3.1) may admit hyperedges involving only peripheral nodes, albeit with low probability.

As a final experiment, we show in Figure 5 colored sparsity plots for the clique-expanded graph of the two email hypergraph datasets. Each sparsity plot shows the nonzero entries of the weighted adjacency matrix of the clique-expanded graph, as defined in (5.1). Nonzero entries are shaded according to their relative value; darker $(A_H)_{ij}$ correspond to larger edge weights. Each column in the figure shows the colored sparsity pattern obtained by permuting the entries of the matrix according to the core-periphery score vector obtained with one of the methods considered. This figure further highlights how the core-periphery detection problem in hypergraphs fundamentally differs from the same problem on the projected graph. Although `GraphNSM` fails to recover the planted hypergraph cores, as we saw in Figure 4, it finds more compact core structures than `HyperNSM` on the clique expansion graph (5.1).

6.4. Real-world datasets with no available core information. Next, we test `HyperNSM`, `UMHS` and `GraphNSM` on a set of hypergraphs where the presence of a core set is not known a priori. Basic details about the datasets used in this section are summarized in Table 3.

The first collection of datasets are co-citation hypergraphs *Cora*, *Citeseer* and *Pubmed* [24, 32, 45]. All nodes in the datasets are documents and hyperedges are based on co-citation (all papers cited in one manuscript form a hyperedge). These hypergraphs are unweighted. The second collection of hypergraphs is built starting from a timestamped sequence of simplices, as in [11]. Given a temporal sequence of simplices where each simplex is a set of nodes, we represent the dataset as a hypergraph with one hyperedge for each simplex (ignoring the time
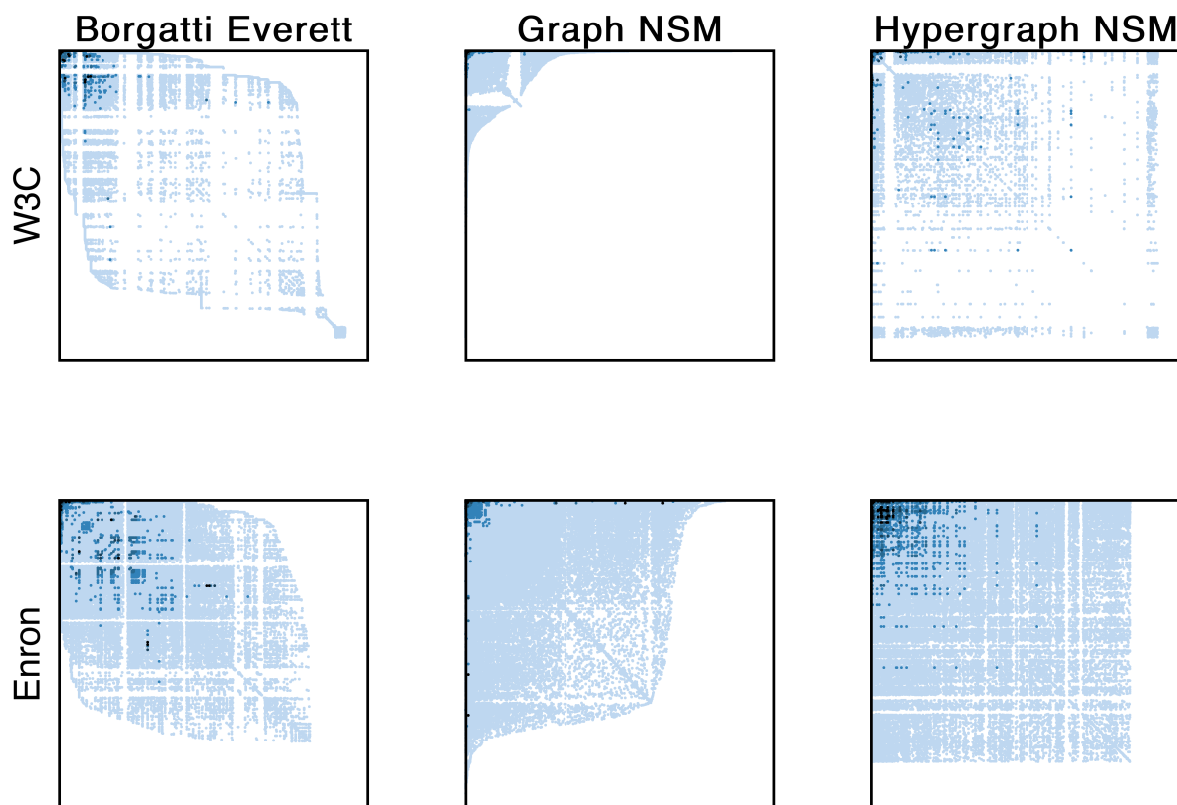
Figure 5: Colored sparsity plots of the clique-expanded graph's adjacency matrix. Nonzero elements in the matrix are colored according to their value (the larger the darker) and are permuted according to the entries of the core-periphery score vector of three different methods.

stamp), weighted with an integer counting how many times that hyperedge appears in the data. Nodes in *Email EU* are email addresses at a European research institution and each hyperedge is a set of emails sent to one or multiple recipients. Nodes in the *NDC substances* dataset are drugs, and hyperedges are formed by all the drugs corresponding to a National Drug Code by the U.S.A. Food and Drug Administration, while the *NDC classes* dataset is made out of one hyperedge per drug and the nodes are class labels applied to the drugs. Finally, nodes in *Tags ask-ubuntu* are tags and hyperedges are the sets of tags applied to questions on `https://askubuntu.com/`.

From the core-periphery profiles in Figure 6 we see that for this citation data both `HyperNSM` and `UMHS` find a convincing core structure. The clique expansion based method, `GraphNSM`, also gives a sharp core-periphery transition, albeit with a much larger core. For the timestamped simplex data, however, as shown in Figure 7 both the clique expansion and hitting set approaches fail to reveal core-periphery structure, whereas `HyperNSM` gives a sharp transition in each case.

	# nodes	# edges	$w(e)$			$|e|$		
			max	mean	std	max	mean	std
Cora	2708	1579	1	1	0	6	4.03	1.02
Citeseer	3306	1079	1	1	0	27	4.2	2.02
Pubmed	19717	7963	1	1	0	172	5.35	5.67
NDC classes	1161	1090	2083	45.62	150.11	39	5.97	4.99
NDC substances	5556	10273	2419	10.99	59.13	187	6.62	9.3
Tags ask-ubuntu	3029	147222	1373	1.84	11.76	5	3.39	1.02
Email EU	1005	25148	4875	9.36	50.35	40	3.56	3.4

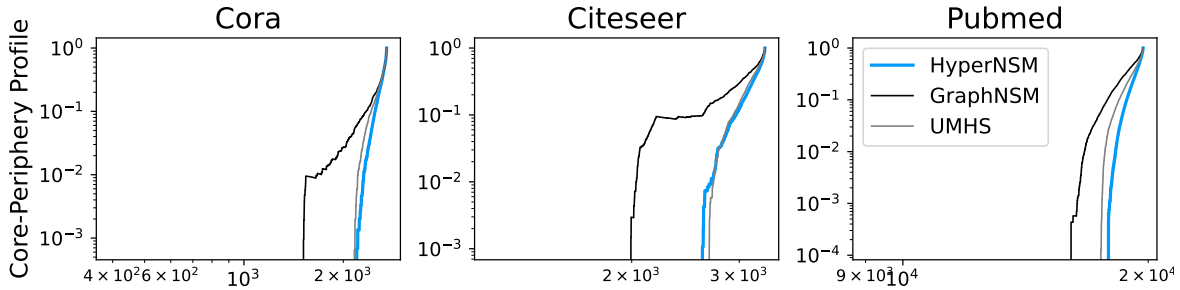Table 3: Basic statistics for the real-world hypergraph datasets with no planted core.

Figure 6: Core-periphery profiles of citation hypergraphs for three different core-periphery detection methods.

7. Conclusion. Our aim in this work was to extend existing graph-based core-periphery concepts and algorithms to the hypergraph setting, in order to account for the group-level interactions captured by many modern data collection processes. Our key take-home messages are that

- the widely used approach of solving a suitable optimization problem can be generalized to the hypergraph case, and a globally convergent iteration scheme is available,
- in this core-periphery setting, the general-purpose recipe "flatten the hypergraph into a weighted clique expansion graph and apply a graph algorithm" does not reduce the computational complexity and does not lead to satisfactory results,
- an extension of the graph core-periphery profile can be used to judge performance, and the new nonlinear spectral method revealed core-periphery structure in a range of real datasets.

Data Statement. The code used for the computations described here can be found at https://github.com/ftudisco/core-periphery-hypergraphs and the real datasets were obtained via the references cited in the text.
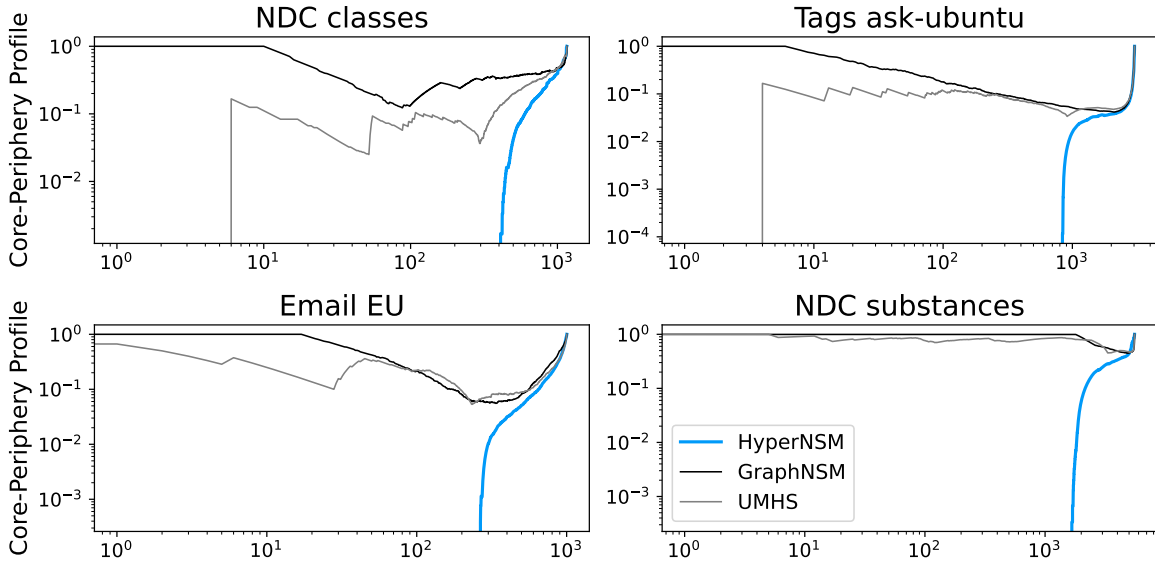
Figure 7: Core-periphery profile of four hypergraphs obtained from timestamped simplex data, for three different core-periphery detection methods.

Appendix A. Proof of the main result.

We devote this section to the proof of Theorem 4.1. Let f be defined as in (4.1), with $\xi(e)$ a positive scaling function of the hyeperedges. As f is positively one-homogeneous, i.e., we have $f(\lambda x) = \lambda f(x)$ for all $\lambda > 0$, the constrained optimization problem in (4.1) coincides with the unconstrained optimization of g defined as $g(x) = f(x/\|x\|_p) = f(x)/\|x\|_p$. Thus, $x^\star$ is a solution to (4.1) if and only if $\nabla g(x^\star) = 0$.

A direct computation of the gradient of g implies that $\nabla g(x) = 0$ if and only if $\nabla f(x) = g(x)\nabla\{\|x\|_p\}$. Suppose $x \succeq 0$. We have $\nabla\{\|x\|_p\} = (\|x\|_p)^{1-p}x^{p-1}$ and $\nabla f(x) = F(x)$ with

$$F(x) = \operatorname{Diag}(x)^{q-1}B\Xi(B^\top x^q)^{\frac{1}{q}-1}.$$

Thus, $\nabla g(x) = 0$ with $x \succeq 0$ if and only if $F(x) = \lambda x^{p-1}$, for some $\lambda > 0$.

For a $p > 1$, let $p^* = p/(p-1)$ denote its Hölder conjugate such that $1/p + 1/q = 1$. Define

$$H(x) = \frac{F(x)^{p^*-1}}{\|F(x)^{p^*-1}\|_p} = \frac{F(x)^{\frac{1}{p-1}}}{\|F(x)^{\frac{1}{p-1}}\|_p} = \left(\frac{F(x)}{\|F(x)\|_{p^*}}\right)^{\frac{1}{p-1}}.$$

From $F(x) = \lambda x^{p-1}$, we have that $\nabla g(x) = 0$ for $x \succeq 0$ if and only if $H(x) = x/\|x\|_p$ and, for a point such that $\|x\|_p = 1$, if and only if $H(x) = x$. We will show next that H has a unique fixed point such that $x^\star \succ 0$ and $\|x^\star\|_p = 1$ and that the sequence $x^{(k+1)} = H(x^{(k)})$ converges to $x^\star$ for any $x^{(0)}$ with positive entries. This implies that $f(x)$ has only one critical point $x^\star$ with positive entries and of unit p-norm. Note that the sequence $x^{(k)}$ coincides with the

iterations generated by scheme in the statement of Theorem 4.1, thus showing the convergence of $x^{(k)}$ and its convergence rate will conclude the proof.

To this end, we will show that H is a contraction with respect to the Thompson distance, defined as

$$d(x,y) = \|\ln x - \ln y\|_\infty \,,$$

for any two $x, y \succ 0$. Precisely, we will show that for any two points $x, y \in S_+ = \{x : x \succ 0, \|x\|_p = 1\}$ we have $H(x), H(y) \in S_+$ and

$$d(H(x), H(y)) \le \left| \frac{q-1}{p-1} \right| d(x,y)\,.$$

Since S_+ is complete with respect to d (see e.g. [31]), the Banach fixed point theorem will then directly imply the thesis.

In order to prove the contraction bound above we need a number of careful computations. First, using the mean value theorem we have

$$
\begin{aligned}
(\text{A.1}) \qquad d(H(x), H(y)) &= \|\ln(H(x)) - \ln(H(y))\|_\infty \\
&= \|\ln(H(\exp(\ln(x)))) - \ln(H(\exp(\ln(y))))\|_\infty \\
&\le \sup_{z \in \Omega(x,y)} \|\mathcal{F}(z)(\ln(x) - \ln(y))\|_\infty \le d(x,y) \sup_{z \in \Omega(x,y)} \|\mathcal{F}(z)\|_\infty,
\end{aligned}
$$

where $\Omega(x,y)$ is the line segment joining x and y, and $\mathcal{F}(z)$ denotes the Frechét derivative of the map $\ln \circ H \circ \exp$ evaluated at z. A direct computation using the chain rule shows that

$$\mathcal{F}(z) = \mathrm{Diag}(H(e^z))^{-1} \mathcal{D}H(e^z) \,\mathrm{Diag}(e^z) = \mathrm{Diag}(H(\widetilde{z}))^{-1} \mathcal{D}H(\widetilde{z}) \,\mathrm{Diag}(\widetilde{z}),$$

where $\widetilde{z} \in \widetilde{\Omega} := \exp(\Omega(x,y))$ and where $\mathcal{D}$ denotes the Jacobian operator.

Using the chain rule several times we compute

$$\mathcal{D}H(z) = \frac{1}{p-1} \left(\frac{F(z)}{\|F(z)\|_{p^*}} \right)^{\frac{1}{p}-1} \left\{ \frac{\mathcal{D}F(z)}{\|F(z)\|_p} - \frac{F(z)(\nabla \|F(z)\|_p)^T \mathcal{D}F(z)}{\|F(z)\|_p^2} \right\}$$

which implies that

$$
\begin{aligned}
|\,\mathrm{Diag}(H(z))^{-1}\mathcal{D}H(z)| &= |\,\mathrm{Diag}(F(z))^{\frac{1}{1-p}} \|F(z)\|_{p^*}^{\frac{1}{p}-1} \mathcal{D}H(z)| \\
&= \frac{1}{|p-1|} \left| \mathrm{Diag}(F(z))^{-1}\mathcal{D}F(z) - \frac{\mathbb{1}\,(\nabla \|F(z)\|_{p^*})^\top \mathcal{D}F(z)}{\|F(z)\|_{p^*}} \right| .
\end{aligned}
$$

For $z \succ 0$, we have $\nabla \|F(z)\|_{p^*} = \|F(z)\|_{p^*}^{1-p^*} F(z)^{p^*-1}$. Thus

$$
\begin{aligned}
|p-1|\left(|\,\mathrm{Diag}(H(z))^{-1}\mathcal{D}H(z)| \right)_{ij} &= \left| \frac{\mathcal{D}F(z)_{ij}}{F(z)_i} - \sum_i \frac{\|F(z)\|_{p^*}^{1-p^*} F(z)_i^{p^*-1} \mathcal{D}F(z)_{ij}}{\left(\sum_k F(z)_k^{p^*} \right)^{1/p^*}} \right| \\
&= \left| \frac{\mathcal{D}F(z)_{ij}}{F(z)_i} - \sum_i \frac{\|F(z)\|_{p^*}^{1-p^*} F(z)_i^{p^*}}{\left(\sum_k F(z)_k^{p^*} \right)^{1/p^*}} \cdot \frac{\mathcal{D}F(z)_{ij}}{F(z)_i} \right| \\
&=: \left| C_{ij} - \sum_i \gamma_i C_{ij} \right|,
\end{aligned}
$$

where $C_{ij} = \mathcal{D}F(z)_{ij}/F(z)_i$ and $\gamma_i = \|F(z)\|_p^{-1} F(z)_i^p$. Notice that C_{ij} and γ_i are all nonnegative numbers (for all i,j) and that $\sum_i \gamma_i = 1$. Therefore,

$$|p-1|\left|\left(\operatorname{Diag} H(x)^{-1}|\mathcal{D}H(x)|\right)_{ij}\right| = |C_{ij} - \sum_i \gamma_i C_{ij}| \leq \max_{i=1,\ldots,n} C_{ij}$$

As a consequence, if $i_\star$ is the index such that $C_{i_\star j} = \max_{i=1,\ldots,n} C_{ij} > 0$, for any $z \succeq 0$, we obtain

$$\left\|\operatorname{Diag} H(z)^{-1}|\mathcal{D}H(z)|\operatorname{Diag}(z)\right\|_\infty = \max_i \sum_j \left|\left(\operatorname{Diag}(H(z))^{-1}|\mathcal{D}H(z)|\right)_{ij} z_j\right|$$

(A.2)
$$\leq \frac{1}{|p-1|}\sum_j |C_{i_\star j} z_j| = \frac{1}{|p-1|}|\sum_j C_{i_\star j} z_j|$$

$$= \frac{1}{|p-1|}\left|\frac{(\mathcal{D}F(z)z)_{i_\star}}{F(z)_{i_\star}}\right|.$$

Now, recall that for a generic node i we have $F(z)_i = z_i^{q-1}\sum_{e:i\in e}\xi(e)(\sum_{k\in e} z_k^q)^{1/q-1}$. Thus, if δ_{ij} denotes the Kronecker delta ($\delta_{ij} = 1$ if $i = j$, $\delta_{ij} = 0$ otherwise), we have

$$\frac{\partial F(z)_i}{\partial z_j} = (q-1)\Big\{\delta_{ij} z_i^{q-2}\sum_{e:i\in e}\xi(e)\big(\sum_{k\in e} z_k^q\big)^{\frac{1}{q}-1} - z_i^{q-1}\sum_{e:i,j\in e}\xi(e)\big(\sum_{k\in e} z_k^q\big)^{\frac{1}{q}-2} z_j^{q-1}\Big\}$$

so that

(A.3)
$$\frac{1}{|q-1|}\left|\frac{(\mathcal{D}F(z)z)_i}{F(z)_i}\right| = \frac{1}{|q-1|}\left|\frac{1}{F(z)_i}\sum_j \frac{\partial F(z)_i}{\partial z_j} z_j\right|$$

$$\leq \frac{1}{F(z)_i}\Big\{\Big| z_i^{q-1}\sum_{e:i\in e}\xi(e)\big(\sum_{k\in e} z_k^q\big)^{\frac{1}{q}-1} - z_i^{q-1}\sum_{e:i\in e}\xi(e)\big(\sum_{k\in e} z_k^q\big)^{\frac{1}{q}-2} z_i^q\Big|$$

$$+ \Big| z_i^{q-1}\sum_{j\neq i}\sum_{e:i,j\in e}\xi(e)\big(\sum_{k\in e} z_k^q\big)^{\frac{1}{q}-2} z_j^q\Big|\Big\}$$

$$= \frac{1}{F(z)_i}\Big\{\Big| z_i^{q-1}\sum_{e:i\in e}\xi(e)\big(\sum_{k\in e} z_k^q\big)^{\frac{1}{q}-1}\frac{\sum_{k\in e\setminus\{i\}} z_i^q}{\sum_{k\in e} z_k^q}\Big|$$

$$+ \Big| z_i^{q-1}\sum_{j\neq i}\sum_{e:i,j\in e}\xi(e)\big(\sum_{k\in e} z_k^q\big)^{\frac{1}{q}-1}\frac{z_j^q}{\sum_{k\in e} z_k^q}\Big|\Big\}$$

$$\leq \frac{1}{F(z)_i}\Big\{ z_i^{q-1}\sum_{e:i\in e}\xi(e)\big(\sum_{k\in e} z_k^q\big)^{\frac{1}{q}-1} + z_i^{q-1}\sum_{j\neq i}\sum_{e:i,j\in e}\xi(e)\big(\sum_{k\in e} z_k^q\big)^{\frac{1}{q}-1}\Big\} = 1 \blacksquare$$

where the final inequality follows from the fact that, for $z \succ 0$, we have

$$0 \leq \frac{\sum_{k \in e \setminus \{i\}} z_i^q}{\sum_{k \in e} z_k^q} \leq 1 \qquad \text{and} \qquad 0 \leq \frac{z_j^q}{\sum_{k \in e} z_k^q} \leq 1 \,.$$

Finally, combining (A.2) with (A.3), we obtain that

$$\sup_{z \in \Omega(x,y)} \|\mathcal{F}(z)\|_\infty = \sup_{z \in \widetilde{\Omega}} \left\| \operatorname{Diag} H(z)^{-1} |\mathcal{D}H(z)| \operatorname{Diag}(z) \right\|_\infty \leq |q - 1|/|p - 1|$$

concluding the proof.

REFERENCES

[1] S. AGARWAL, K. BRANSON, AND S. BELONGIE, *Higher order learning with graphs*, in International Conference on Machine Learning (ICML), 2006, pp. 17–24.

[2] S. AGARWAL, J. LIM, L. ZELNIK-MANOR, P. PERONA, D. KRIEGMAN, AND S. BELONGIE, *Beyond pairwise clustering*, in Conference on Computer Vision and Pattern Recognition (CVPR), vol. 2, IEEE, 2005, pp. 838–845.

[3] I. AMBURG, J. KLEINBERG, AND A. R. BENSON, *Planted hitting set recovery in hypergraphs*, Journal of Physics: Complexity, 2 (2021), p. 035004.

[4] F. ARRIGO, D. J. HIGHAM, AND F. TUDISCO, *A framework for second-order eigenvector centralities and clustering coefficients*, Proceedings of the Royal Society A, 476 (2020), p. 20190724.

[5] D. ARYA, D. K. GUPTA, S. RUDINAC, AND M. WORRING, *Adaptive neural message passing for inductive learning on hypergraphs*, arXiv preprint arXiv:2109.10683, (2021).

[6] A. BANERJEE, *On the spectrum of hypergraphs*, Lin. Alg. Appl., 614 (2021), pp. 82–110.

[7] R. BAÑOS, J. BORGE-HOLTHOEFER, N. WANG, Y. MORENO, AND S. GONZÁLEZ-BAILÓN, *Diffusion dynamics with changing network composition*, Entropy, 15 (2013), pp. 4553–4568.

[8] F. BATTISTON, E. AMICO, A. BARRAT, G. BIANCONI, G. FERRAZ DE ARRUDA, B. FRANCESCHIELLO, I. IACOPINI, S. KÉFI, V. LATORA, Y. MORENO, ET AL., *The physics of higher-order interactions in complex systems*, Nature Physics, 17 (2021), pp. 1093–1098.

[9] F. BATTISTON, G. CENCETTI, I. IACOPINI, V. LATORA, M. LUCAS, A. PATANIA, J.-G. YOUNG, AND G. PETRI, *Networks beyond pairwise interactions: Structure and dynamics*, Physics Reports, 874 (2020), pp. 1–92.

[10] A. R. BENSON, *Three hypergraph eigenvector centralities*, SIAM Journal on Mathematics of Data Science, 1 (2019), pp. 293–312.

[11] A. R. BENSON, R. ABEBE, M. T. SCHAUB, A. JADBABAIE, AND J. KLEINBERG, *Simplicial closure and higher-order link prediction*, Proceedings of the National Academy of Sciences, (2018).

[12] A. R. BENSON, D. F. GLEICH, AND J. LESKOVEC, *Higher-order organization of complex networks*, Science, 353 (2016), pp. 163–166.

[13] G. BIANCONI, *Higher Order Networks: An Introduction To Simplicial Complexes*, Cambridge University Press, Cambridge, 2021.

[14] S. P. BORGATTI AND M. G. EVERETT, *Models of core/periphery structures*, Social Networks, 21 (2000), pp. 375–395.

[15] J. P. BOYD, W. J. FITZGERALD, M. C. MAHUTGA, AND D. A. SMITH, *Computing continuous core/periphery structures for social relations data with MINRES/SVD*, Social Networks, 32 (2010), pp. 125–137.

[16] T. BÜHLER AND M. HEIN, *Spectral clustering based on the graph p-Laplacian*, in Proceedings of the 26th Annual International Conference on Machine Learning, 2009, pp. 81–88.

[17] T. CARLETTI, F. BATTISTON, G. CENCETTI, AND D. FANELLI, *Random walks on hypergraphs*, Physical Review E, 101 (2020), p. 022308.

[18] P. CSERMELY, A. LONDON, L.-Y. WU, AND B. UZZI, *Structure and dynamics of core/periphery networks*, Journal of Complex Networks, 1 (2013), pp. 93–123.

[19] M. CUCURINGU, P. ROMBACH, S. H. LEE, AND M. A. PORTER, *Detection of core–periphery structure in networks using spectral methods and geodesic paths*, Eur. J. App. Math., 27 (2016), pp. 846–887.

[20] M. R. DA SILVA, H. MA, AND A.-P. ZENG, *Centrality, network capacity, and modularity as parameters to analyze the core-periphery structure in metabolic networks*, Proceedings of the IEEE, 96 (2008), pp. 1411–1420.

[21] G. F. DE ARRUDA, M. TIZZANI, AND Y. MORENO, *Phase transitions and stability of dynamical processes on hypergraphs*, Communications Physics, 4 (2021), pp. 1–9.

[22] F. DELLA ROSSA, F. DERCOLE, AND C. PICCARDI, *Profiling core-periphery network structure by random walkers*, Scientific Reports, 3 (2013), pp. 1–8.

[23] A. ELMOATAZ, O. LEZORAY, AND S. BOUGLEUX, *Nonlocal discrete regularization on weighted graphs: a framework for image and manifold processing*, IEEE transactions on Image Processing, 17 (2008), pp. 1047–1060.

[24] C. L. GILES, K. D. BOLLACKER, AND S. LAWRENCE, *Citeseer: An automatic citation indexing system*, in Proceedings of the third ACM conference on Digital libraries, 1998, pp. 89–98.

[25] X. GONG, D. J. HIGHAM, AND K. ZYGALAKIS, *Directed network Laplacians and random graph models*, Royal Society Open Science, (2021), p. 8:211144211144.

[26] P. GRINDROD, *Range-dependent random graphs and their application to modeling large small-world proteome datasets*, Phys. Rev. E, 66 (2002), pp. 066702–1 to 7.

[27] M. HEIN, S. SETZER, L. JOST, AND S. S. RANGAPURAM, *The total variation on hypergraphs-learning on hypergraphs revisited*, Advances in Neural Information Processing Systems, 26 (2013), pp. 2427–2435.

[28] D. J. HIGHAM AND H.-L. DE KERGORLAY, *Epidemics on hypergraphs: Spectral thresholds for extinction*, Proceedings of the Royal Society, Series A, 477 (2021).

[29] R. IBRAHIM AND D. GLEICH, *Nonlinear diffusion for community detection and semi-supervised learning*, in The World Wide Web Conference, 2019, pp. 739–750.

[30] P. M. KIM, J. O. KORBEL, AND M. B. GERSTEIN, *Positive selection at the protein network periphery: evaluation in terms of structural constraints and cellular context*, Proceedings of the National Academy of Sciences, 104 (2007), pp. 20274–20279.

[31] B. LEMMENS AND R. D. NUSSBAUM, *Nonlinear Perron-Frobenius theory*, Cambridge University Press, general ed., 2012.

[32] A. K. MCCALLUM, K. NIGAM, J. RENNIE, AND K. SEYMORE, *Automating the construction of internet portals with machine learning*, Information Retrieval, 3 (2000), pp. 127–163.

[33] A. P. MILLÁN, J. J. TORRES, AND G. BIANCONI, *Explosive higher-order Kuramoto dynamics on simplicial complexes*, Physical Review Letters, 124 (2020), p. 218301.

[34] R. J. MONDRAGÓN, *Network partition via a bound of the spectral radius*, Journal of Complex Networks, (2017), pp. 513–526.

[35] L. NEUHÄUSER, R. LAMBIOTTE, AND M. T. SCHAUB, *Consensus dynamics and opinion formation on hypergraphs*, arXiv preprint arXiv:2105.01369, (2021).

[36] K. PROKOPCHIK, A. R. BENSON, AND F. TUDISCO, *A nonlinear diffusion method for semi-supervised learning on hypergraphs*, arXiv preprint arXiv:2103.14867, (2021).

[37] S. RAO, A. VAN DER SCHAFT, AND B. JAYAWARDHANA, *A graph-theoretical approach for the analysis and model reduction of complex-balanced chemical reaction networks*, Journal of Mathematical Chemistry, 51 (2013), pp. 2401–2422.

[38] J. A. RODRIGUEZ, *On the Laplacian eigenvalues and metric parameters of hypergraphs*, Linear and Multilinear Algebra, 50 (2002), pp. 1–14.

[39] J. A. RODRIGUEZ, *On the Laplacian spectrum and walk-regular hypergraphs*, Linear and Multilinear Algebra, 51 (2003), pp. 285–297.

[40] J. A. RODRIGUEZ, *Laplacian eigenvalues and partition problems in hypergraphs*, Applied Mathematics Letters, 22 (2009), pp. 916–921.

[41] M. P. ROMBACH, M. A. PORTER, J. H. FOWLER, AND P. J. MUCHA, *Core-periphery structure in networks (revisited)*, SIAM Review, 59 (2017), pp. 619–646.

[42] S. SAITO, D. P. MANDIC, AND H. SUZUKI, *Hypergraph p-Laplacian: A differential geometry view*, in Thirty-Second AAAI Conference on Artificial Intelligence, 2018.

[43] K. S. SANDHU, G. LI, H. M. POH, Y. L. K. QUEK, Y. Y. SIA, S. Q. PEH, F. H. MULAWADI, J. LIM, M. SIKIC, F. MENGHI, ET AL., *Large-scale functional organization of long-range chromatin interaction*

networks, Cell Reports, 2 (2012), pp. 1207–1219.

[44] M. T. Schaub, N. O'Clery, Y. N. Billeh, J.-C. Delvenne, R. Lambiotte, and M. Barahona, *Graph partitions and cluster synchronization in networks of oscillators*, Chaos: An Interdisciplinary Journal of Nonlinear Science, 26 (2016), p. 094821.

[45] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, *Collective classification in network data*, AI magazine, 29 (2008), pp. 93–93.

[46] M. V. Tomasello, M. Napoletano, A. Garas, and F. Schweitzer, *The rise and fall of R & D networks*, Industrial and corporate change, 26 (2017), pp. 617–646.

[47] L. Torres, A. S. Blevins, D. S. Bassett, and T. Eliassi-Rad, *The why, how, and when of representations for complex systems*, SIAM Review, 63 (2021), pp. 435–485.

[48] F. Tudisco, A. R. Benson, and K. Prokopchik, *Nonlinear higher-order label spreading*, in Proceedings of the Web Conference, 2021.

[49] F. Tudisco and D. J. Higham, *A fast and robust kernel optimization method for core–periphery detection in directed and weighted graphs*, Applied Network Science, 4 (2019), pp. 1–13.

[50] F. Tudisco and D. J. Higham, *A nonlinear spectral method for core-periphery detection in networks*, SIAM J. Mathematics of Data Science, 1 (2019), pp. 269–292.

[51] F. Tudisco and D. J. Higham, *Node and edge nonlinear eigenvector centrality for hypergraphs*, Communications Physics, 4 (2021), pp. 1–10.

[52] P. Upadhyaya, E. Jarlbering, and F. Tudisco, *The self-consistent field iteration for p-spectral clustering*, arXiv:2111.09750, (2021).

[53] A. Van der Schaft, S. Rao, and B. Jayawardhana, *A network dynamics approach to chemical reaction networks*, International Journal of Control, 89 (2016), pp. 731–745.

[54] N. Veldt, A. R. Benson, and J. Kleinberg, *Hypergraph cuts with general splitting functions*, SIAM Review, to appear (2022).

[55] D. Zhou, J. Huang, and B. Schölkopf, *Learning with hypergraphs: Clustering, classification, and embedding*, in Advances in Neural Information Processing Systems (NeurIPS), 2007, pp. 1601–1608.

[56] J. Y. Zien, M. D. Schlag, and P. K. Chan, *Multilevel spectral hypergraph partitioning with arbitrary vertex sizes*, IEEE Trans. CAD of Integrated Circuits and Systems, 18 (1999), pp. 1389–1399.