White Rose Research Online

eprints@whiterose.ac.uk
https://eprints.whiterose.ac.uk
Universities of Leeds, Sheffield and York

Deposited via The University of Sheffield.

White Rose Research Online URL for this paper:
https://eprints.whiterose.ac.uk/id/eprint/152521/

Version: Published Version

Article:
Clark, R., Punzo, G. and Macdonald, M. (2019) Network communities of dynamical influence. Scientific Reports, 9. 17590. ISSN: 2045-2322

https://doi.org/10.1038/s41598-019-53942-4

Reuse
This article is distributed under the terms of the Creative Commons Attribution (CC BY) licence. This licence allows you to distribute, remix, tweak, and build upon the work, even commercially, as long as you credit the authors for the original work. More information and the full terms of the licence here: https://creativecommons.org/licenses/

Takedown
If you consider content in White Rose Research Online to be in breach of UK law, please notify us by emailing eprints@whiterose.ac.uk including the URL of the record and the reason for the withdrawal request.

SCIENTIFIC REPORTS
nature research

OPEN

Network Communities of Dynamical Influence

Ruaridh Clark[1*], Giuliano Punzo[2] & Malcolm Macdonald[1]

Fuelled by a desire for greater connectivity, networked systems now pervade our society at an unprecedented level that will affect it in ways we do not yet understand. In contrast, nature has already developed efficient networks that can instigate rapid response and consensus when key elements are stimulated. We present a technique for identifying these key elements by investigating the relationships between a system's most dominant eigenvectors. This approach reveals the most effective vertices for leading a network to rapid consensus when stimulated, as well as the communities that form under their dynamical influence. In applying this technique, the effectiveness of starling flocks was found to be due, in part, to the low outdegree of every bird, where increasing the number of outgoing connections can produce a less responsive flock. A larger outdegree also affects the location of the birds with the most influence, where these influentially connected birds become more centrally located and in a poorer position to observe a predator and, hence, instigate an evasion manoeuvre. Finally, the technique was found to be effective in large voxel-wise brain connectomes where subjects can be identified from their influential communities.

Human capacity to create networked systems is expanding with the growing popularity of cloud services[1], improvements in mobile (cellular) network infrastructure and the expansion of the internet, including satellite based provision[2]. Increased connectivity is not without its drawbacks where denial of service attacks, exploiting Internet of Things (IoT) devices, are an example of the vulnerabilities that can emerge in large, complicated, networks[3]. To protect against this growing reliance on networked systems, this paper strives to further our understanding of how network topology influences dynamical response by identifying communities as key pathways of communication from the most influential network nodes.

Artificial networks are sophisticated but they cannot compete with nature's accomplishments, where the human brain is estimated to contain 100 billion neurons and 100 trillion synaptic connections[4]. For the small neuronal system of the *Caenorhabditis elegans*[5] and in a low resolution network representation of the macaque connectome[6], the flow of information has been simulated to reveal the existence of bottlenecks and clusters. Such numerical models of information flow become intractable at a sufficiently large scale, therefore this paper uses a spectral method to uncover communication dynamics in some of nature's most sophisticated networks, including human connectomes with around 850,000 vertices.

Community Detection

This paper presents an eigenvector-based community detection method. Most community detection, partitioning and graph clustering methods consider either the edge direction or the graph spectral properties to create network divisions, such as modularity, random walk based methods and spectral clustering[7]. Modularity detects communities by comparing the density of edges present with the density that would be present if all the edges were reassigned a new destination vertex at random[8]. Random walk based methods can either emulate the properties of the first eigenvector of the adjacency matrix, see PageRank[9], or be used to identify communities from modelled diffusion of information[6,10]. Two commonly used spectral clustering approaches are normalised cuts[11] and k-means clustering[12]. The approach, detailed herein, is most similar to that of k-means clustering whereby multiple eigenvectors of the Laplacian matrix are used to split the graph into communities. In the case of k-means clustering, it employs a heuristic algorithm (k-means) to separate the network into k clusters where each cluster is defined in relation to one of k centroids. These centroids iteratively adjust their positions with the final composition of the clusters dependent on the initial placement of the centroids[12]. There is no optimal solution to the

[1]Department of Mechanical and Aerospace Engineering, University of Strathclyde, Glasgow, United Kingdom. [2]Department of Automatic Control and Systems, University of Sheffield, Sheffield, United Kingdom. *email: ruaridh.clark@strath.ac.uk

selection of k for any given network instead multiple runs can be performed to determine the sensitivity of k to a chosen criteria[13].

In this paper, we detect communities without the use of recursive optimisation, instead the number of communities, their members, and influential vertices are all a product of the network's spectral properties. The communities, defined herein, are referred to as Communities of Dynamical Influence (CDI) as each is associated with a vertex that has a high dynamical influence. *Dynamical influence* is defined as a measure of how strongly a node's dynamical state affects the collective behaviour (i.e. state) of the network[14]. These influential vertices are detected with the first left eigenvector of the Laplacian matrix, which has been employed previously to identify the most effective vertices for spreading information or disease in a dynamical network[14]. The first eigenvector of the adjacency matrix is also widely used as a centrality metric[15], where it identifies vertices that would be visited most often by agents performing a random walk on the graph. Influential communities have been considered before from the perspective of identifying a node grouping that maximises its influence. To achieve these high influence groupings, community detection parameters are varied using for example modularity or *k*-core[16], with each vertex given an influence value that is defined either independently of the network topology, by applying a weighting to the vertices[17], or as a product of the topology, by using Katz centrality (a form of eigenvector centrality)[18]. Influence can also be assessed after community designation, where an influence propagation model can be used to assess each community's influential reach[19]. In this paper, we do not attempt to maximise the influence of community members but instead consider the influence that propagates from vertices during consensus. Information propagation of linear consensus has been used previously to define influential communities[20], where each community is composed of vertices that are on one of the most effective pathways from an influential vertex. In this paper, the same definition is applied for community detection but we do not rely on a model or numerical process, instead communities and dynamical influence can be detected from the relationships between the system's eigenvectors.

This paper includes the application of CDI on a starling flock model and voxel-wise human brain networks. Starlings are known to employ a near constant outdegree (six to seven for all flock members[21]), which has been found to maximise robustness and allows them to manage uncertainty in consensus[22]. It has also been observed that starlings on the edge of the flock are those that trigger predator avoidance manoeuvres, as they are first to observe predators[23,24]. The current paper builds on these findings be considering how the outdegree, maintained by starlings, can facilitate their responsiveness to birds attempting to trigger a predator evasion manoeuvre. While the brain analysis compares the influential communities present in the same subject but at different times, using scan-rescan magnetic resonance imaging (MRI) data from Landman *et al.*[25], to not only identify subjects from their neuronal communities but also detect changes in their functional activity.

Influence of Network Perturbations

To validate the claims that the CDI are the communities with the greatest influence over the consensus process, we investigate the optimisation of consensus leadership. The speed of consensus can be captured analytically by assessing the first eigenvalue of the system matrix[26]. Consensus can be driven to a target value by applying a constant input perturbation to a set of vertices that have a directed connection to all other vertices[26]. Similar perturbation optimisations have been studied extensively in the context of leadership selection for the control of multi-agent and swarm systems[26–30]. In these cases, the perturbation is often constrained so that only a set number of leaders are chosen with a binary option for perturbation input, i.e. vertices set as leaders or followers. In the context of multi-agent systems, there is significantly more literature on minimising the steady-state variance about an input perturbation[27,28] than there is on fast convergence to consensus. There has been an attempt to tackle both problems, but this work was restricted to 1-D community and ring graphs[29], and an examination of how the proportion of leader vertices affects the convergence rate to consensus in multiplex networks[30]. Of most relevance, to the work herein, is globally bounded input perturbations that can be applied with a variable distribution to any combination of vertices[26]. For such a case, the first left eigenvector of the Laplacian matrix was identified as a sub-optimal resource allocation (equivalent to an input perturbation) for achieving fast convergence to consensus[26]. An improvement in this allocation has since been developed for directed *k*-outdegree graphs, where a near-optimal perturbation vector can be produced by combining first left eigenvectors from manipulated versions of the adjacency matrix[31]. It is worth noting that the globally bounded perturbation optimisation problem, to maximise convergence rate to consensus, has no verifiable solution. A numerical optimiser can produce near-optimal solutions, but detection of the global minima is not guaranteed. A significant contribution of this article is the filtering out of local minima by highlighting the most effective network influencers, a process that functions at any network scale and is no longer restricted to *k*-outdegree graphs.

Results

The Communities of Dynamical Influence (CDI) are detected using the most dominant left eigenvectors of the Laplacian matrix, as described in the Methods section. The CDI are shown in Fig. 1 for 100 vertex, $k = 10$, nearest neighbour (*k*-NNR) networks. The *k*-NNR networks are generated by randomly distributing vertices in a square plane with each vertex connecting to its *k* nearest neighbours according to euclidean distance. The first left eigenvector ($\mathbf{v}_{L1}$) is always used to determine the dynamical influence of the CDI but, as shown in Fig. 1 where the axes are $\mathbf{v}_{L2}$ and $\mathbf{v}_{L3}$, the other input eigenvectors affect the community composition. This can be seen most clearly in Fig. 1b where the second eigenvector of the Laplacian matrix ($\mathbf{v}_{L2}$) divides the network in two as CDI only has the first two eigenvectors as input. $\mathbf{v}_{L1}$ has only positive entries, so community vertices form a trail behind their community leader that ends close to the origin of the eigenvector coordinate system.

The CDI can be used as an input to a perturbation optimiser, detailed in the Methods section (Algorithm 3), that optimises the network's convergence to consensus by maximising the first eigenvalue of the perturbed

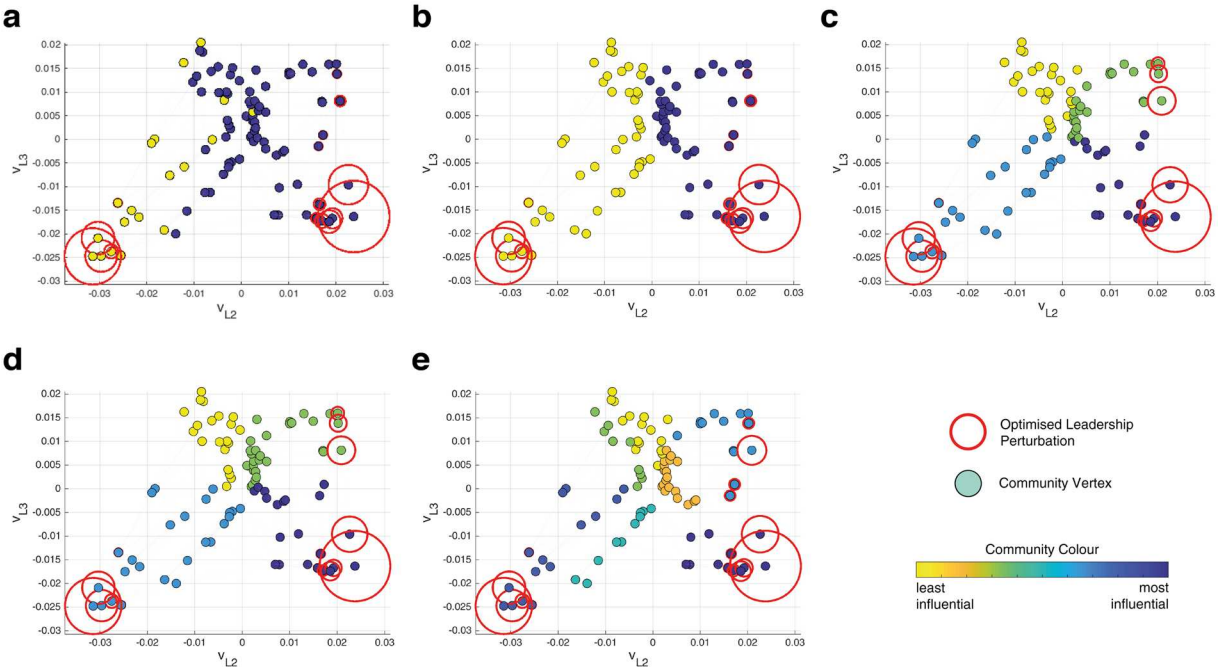

Figure 1. Communities defined by the CDI method with (**a**) 1 eigenvector, (**b**) 2 eigenvectors, (**c**) 3 eigenvectors, (**d**) 4 or 5 eigenvectors, (**e**) 6 eigenvectors. The network is k-NNR with 100 vertices, where $k = 10$. Communities are denoted by colour according to their influence over the whole network. An optimised perturbation using the chosen community detection overlays the network with circles that are proportional to the perturbation magnitude.

Laplacian matrix. Essentially, the optimisation identifies the most effective leaders in the network, those with the highest v_{L1} values in each community, and applies a perturbation of variable magnitude to those vertices. The leadership perturbations applied from the CDI-based optimisation are detailed in Fig. 1, where the 3, 4 and 5 eigenvector CDI (Fig. 1b,c) produce the same result and a faster convergence than the 1, 2 and 6 eigenvector CDI. The optimal number of eigenvectors, in terms of effective consensus leadership, varies depending on the network. Using only one or two eigenvectors makes it more likely that important community divisions are not identified, as seen in Fig. 1a,b, which produce the slowest convergence by only identifying two communities. Using more eigenvectors can also result in a sub-optimal performance with 6 eigenvectors outperforming the 1 and 2 eigenvector case but not the 3, 4 or 5 eigenvector optimisation.

In the following section, three eigenvectors are used for the detection of dynamical influence. CDI with three eigenvectors does not always produce the fastest consensus but in the following section it is shown to produce consistently good results (see Fig. 2). Three eigenvectors provides accurate community division whilst ensuring all communities are amongst the most influential. The issue with using more eigenvectors is that the detected communities may no longer reflect the most effective for leading network consensus. This has already been seen in Fig. 1e where a community formed (light blue - top right) with vertices that had high v_{L1} values because they were connected to more influential vertices in other communities, and not because they were effective network leaders on their own. A perturbation optimisation comparisons between 3 and 4 eigenvector CDI, using the series of k-NNR networks from Fig. 2b, revealed that 4 eigenvector CDI often produced a superior community division, and hence faster convergence, but it was occasionally susceptible to significant inaccuracies. Such inaccuracies were a result of poor community designation for the same reason that Fig. 1e produced a sub-optimal result.

Validation of community influence. When referring to the influence of a community, we are referring to the influence of the influential vertices in that community. These vertices wield the greatest global influence within their local cluster, but this usually means that they are also the most locally influential vertex. To validate the claim that the CDI are influential, we performed a series of analyses comparing perturbations optimised to drive convergence in linear consensus. The vertices highlighted by this optimisation are those that can lead the network effectively to a new state of consensus, i.e. those with strong local and global network influence, which should align with the influential CDI vertices and their communities.

The CDI and k-means clustering algorithms are used as an input to the perturbation optimiser, detailed in the Methods section (Algorithm 3). It should be noted that the k-means clustering requires the number of divisions to be defined and, hence, is set to detect the same number of communities as found by the CDI algorithm. The results of these optimisations are compared with the Communities of Influence (CoI) method[31] and a numerical optimiser[32] in Fig. 2. The CoI method generates optimised perturbations by detecting influence, using the first left eigenvector, and investigating how this influence changes when key vertices are removed from the network.

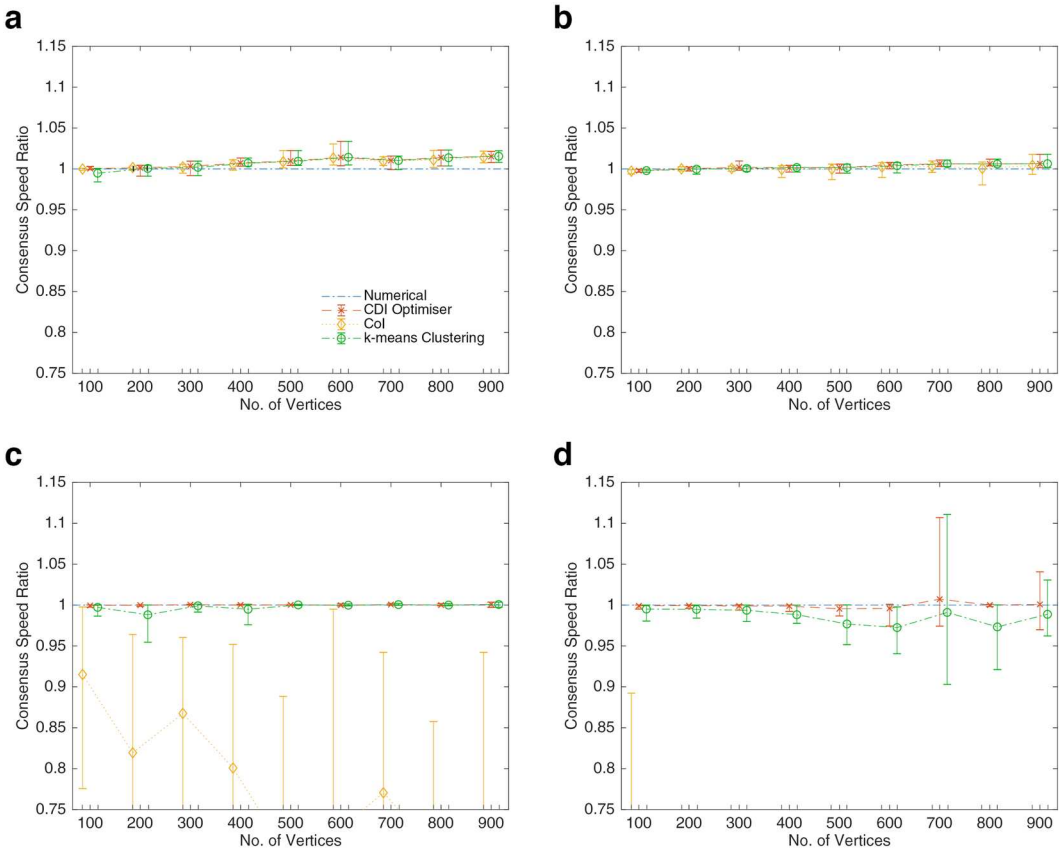

Figure 2. Consensus Speed Ratio, with reference to a numerical optimiser, for three different optimisations with 10 graphs at each vertex size interval. Outdegree is set at $k = 10$ for (**a**) Erdös-Rényi random networks and (**b**) k-NNR networks. Outdegree is varied between $k = 3$ and $k = 10$ for each vertex, by sampling at random from a uniform distribution, for (**c**) Erdös-Rényi random networks and (**d**) k-NNR networks.

This was shown to be effective in k-outdegree networks where the CoI method, using 5 input vectors, produced similar results to the output of a numerical optimiser[31]. In Fig. 2, the CDI optimiser is shown to produce similar result in k-outdegree networks to the CoI method, but achieves notably superior results when applied to variable k networks. In fact the CoI results are too low to be included for many of the networks sizes reported in Fig. 2c,d. The CDI-based optimiser frequently exceeds the results of the numerical optimiser, which struggles to find globally optimal minima.

The k-means clustering algorithm developed by Ng *et al.*[33]. also employs eigenvectors and a form of machine learning to define clusters. Figure 2 shows that the k-means clustering algorithm, when provided with the number of communities found by the CDI algorithm, can often identify the most effective network leaders by placing them in separate communities. The main differences between CDI and k-means clustering are seen in the variable outdegree k-NNR case, Fig. 2d, where the CDI performs consistently better.

The superiority of CDI in comparison with k-means clustering is made clearer by reducing the weight of each edge for a given graph. This alteration constricts the flow of information through the graph, with the result that perturbations need to be spread to a greater number of vertices to overcome this restriction. This phenomena is seen in Fig. 3a,b for a 100 vertex k-NNR topology with edge weight set at 1 and 0.2 respectively. If the edge weight was reduced to 0, then the optimal perturbation would provide all vertices with the same perturbation, i.e. lead them all individually. By requiring more network leaders, the difference between CDI and k-means clustering can be seen even for the k-NNR topology, where the methods performed similarly in Fig. 2b. Comparing CDI in Fig. 3b with k-means clustering in Fig. 3c shows that the clusters generated by both methods are similar. But in the bottom right quadrant of the figures it can be seen that CDI detects a division that k-means does not and this results in a faster convergence speed for the CDI-based optimisation. These small differences in community designation are also what differentiates the methods in Fig. 2d.

Responsive starling flock topology. Starling flocks tend towards a thicknesses of between 0.13 and 0.27, where the flock thickness is the ratio of the smallest to largest dimension of an ellipsoid having the same principal moments of inertia as the flock[22]. In Fig. 4, five examples of 1200 bird starling flock networks are presented with a thickness of 0.2, where the flock is modelled by randomly distributing birds from a uniform distribution within a rectangular prism[22]. The community distribution and optimised perturbations are shown for these representative

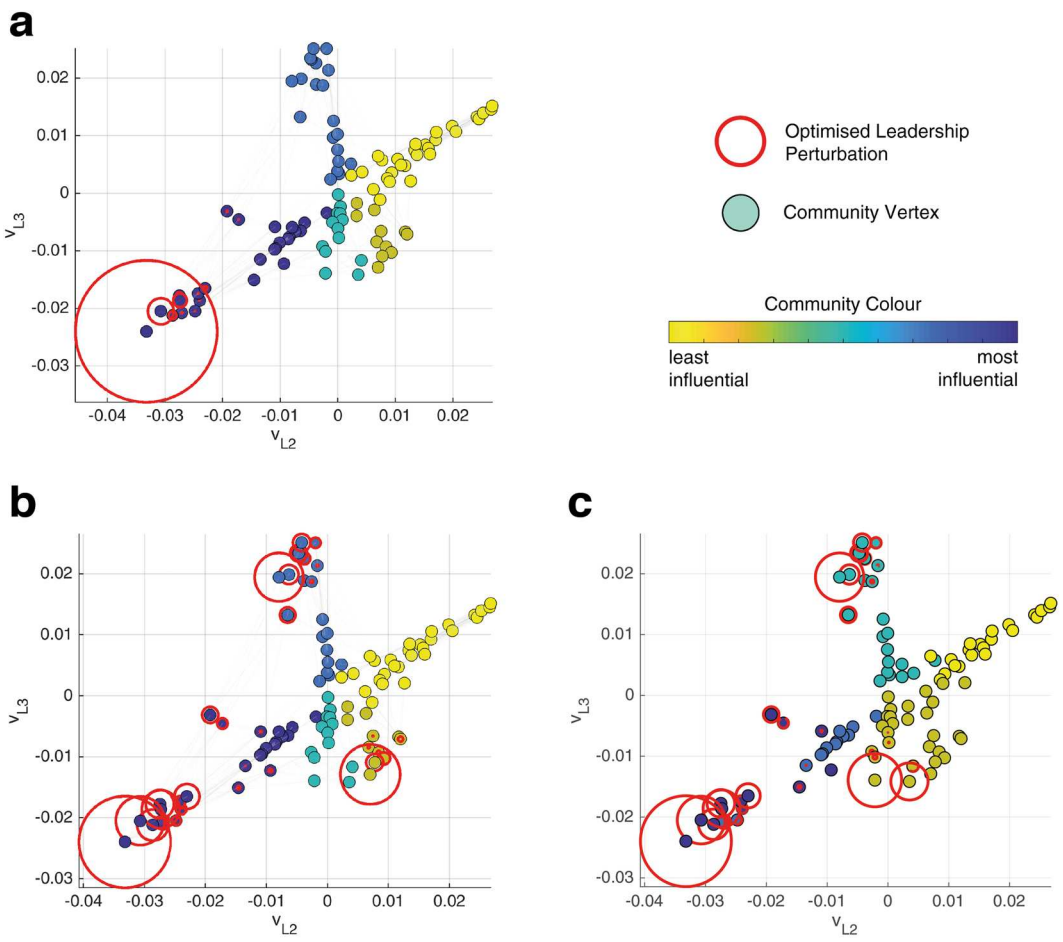

Figure 3. Communities defined by the CDI method with 3 eigenvectors for (**a**) edge weights of 1, and (**b**) edge weights of 0.2 and in (**c**) k-means clustering is applied for edge weights of 0.2. The network is *k*-NNR with 100 vertices, where the outdegree varies between 3 and 10 by sampling at random from a uniform distribution. Communities are denoted by colour according to their influence over the whole network. An optimised perturbation using the chosen community detection overlays the network with circles that are proportional to the perturbation magnitude.

examples that employ *k*-NNR topologies for the following outdegrees; $k = 5$, 7, 15, 25 and 50. The starling vertices remain in the same position for each analysis with the number of nearest neighbours, *k*, the only change that affects the network structure.

What is of particular interest, from these results, is the position of the influential vertices in each example. In high *k* outdegree examples, such as Fig. 4c–e, the influential vertices, associated with the most influential communities, are centrally located. An optimised perturbation (using 3 eigenvector CDI) is shown to align with the most influential communities, therefore the perturbations become increasingly centrally located as the outdegree increases. For the $k = 50$ example, in Fig. 4e, these perturbations are primarily located in the centre of the model flock. For lower outdegree examples, such as Fig. 4a,b, the influential vertices, especially those associated with the most influential communities, are more evenly distributed throughout the flock, aided by the increase in the number of CDI present. Considering an actual starling flock where $k \approx 7$, the topology makes it more likely, in comparison with a higher outdegree scenario, that one of the most influential birds will be near the site of a predator attack. Therefore, the low outdegree topology is more likely to have a highly influential bird involved in leading a predation avoidance manoeuvre.

Varying the outdegree for the starling flock model, while using the same distribution of 1200 birds, and applying an optimised leadership perturbation changes the value of the dominant eigenvalue λ_1 of the perturbed Laplacian matrix, which represents convergence speed. For outdegrees between $k = 5$ and $k = 50$, λ_1 results roughly conform ($R^2 = 0.962$) to a power law distribution ($\lambda_1 = 0.0031k^{-0.184}$). A higher convergence speed is indicative of a graph that can be more effectively led by key vertices. The highest convergence speeds, therefore, usually belong to networks with lower outdegrees. Given that a lower outdegree tends to result in faster perturbation driven consensus, the reason for the outdegree remaining as high as 7 may be due to the requirements of maintaining a connected flock. If the outdegree is too low the flock may split whenever a perturbation is applied and may not reconnect. Defining a lower bound that ensures connectivity is beyond the scope of this paper, but

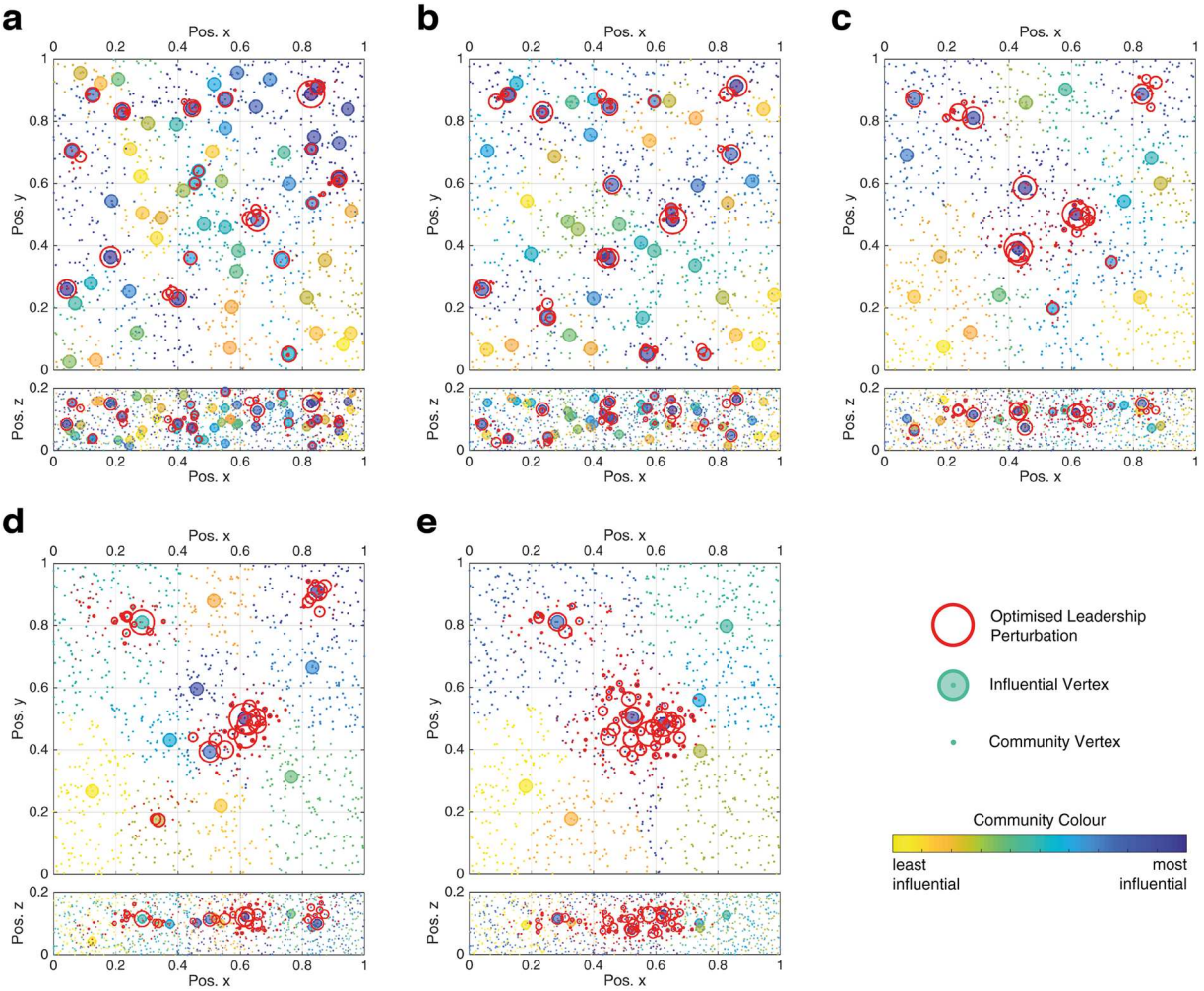

Figure 4. Starling communities are denoted by colour according to the CDI method with the three most dominant eigenvectors for a 1200 vertex k-NNR starling flock model where (**a**) $k=5$, (**b**) $k=7$, (**c**) $k=15$, (**d**) $k=25$ and (**e**) $k=50$. A single influential vertex is displayed for each community with a shaded circle and the optimised leadership perturbation using 3 eigenvector CDI is denoted with circles that are proportional to the perturbation magnitude.

lower bounds have been identified for static topologies including k-NNR graphs where the required k increases with the number of vertices[34].

Brain similarity detection. The CDI method is applied in this section on large-scale human connectomes generated by Roncal *et al.*[35] from magnetic resonance imaging (MRI) scans carried out by Landman *et al.*[25] Landman *et al.* used 21 healthy volunteers, where each subject was scanned twice with a short break between scan and rescan (scan 1 and scan 2 respectively)[25]. Note that one of the scans, for *subject 127*, could not be sourced and, hence, this article shall consider the remaining 20 volunteers.

The networks generated by Roncal *et al.* are undirected and the adjacency matrix, rather than the Laplacian is used, due to the scale of the network (see Methods section). Therefore, the CDI no longer identifies the most effective leaders of system consensus as every vertex both leads and follows equally. This makes it impossible to differentiate between important sources and sinks of information in the network. The use of the adjacency matrix also means that the communities are ranked by popularity, i.e. the frequency with which a random walker would visit each vertex in the graph, rather than effective consensus leadership. Therefore, instead of drawing conclusions about a community's network leadership, the CDI is used as a similarity metric by detecting changes in the influential communities (whether sources or sinks) of brain connectomes.

In the work by Roncal *et al.*[35], the Frobenius Norm was successfully used to detect the similarity of these scan-rescan matrices created from Landman *et al.*'s study[25]. The Frobenius Norm is an established matrix distance measure[36], referred to as Frobenius Distance when assessing graph similarity. The result from Roncal *et al.*'s study[35] has not proven to be exactly reproducible with the published dataset[25], as the scan-rescan comparisons

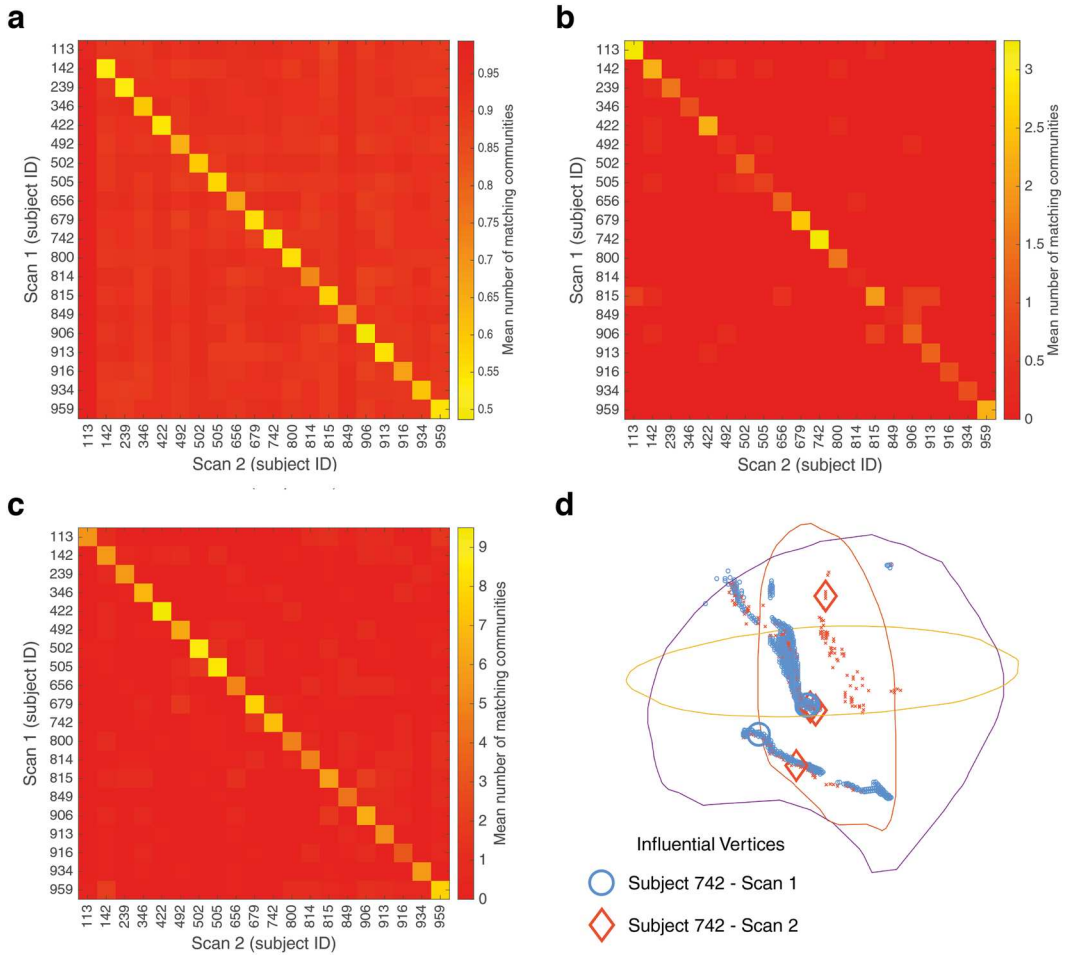

Figure 5. Comparison of scan 1 and scan 2 from Landman *et al.*[25] subjects using (**a**) *edge graph edit distance*, and the*mean number of matching communities* with CDI using (**b**) 3 eigenvectors and (**c**) 10 eigenvectors of the adjacency matrix. (**d**) displays the communities detected by CDI, using 3 eigenvectors, for scan 1 and 2 of subject 113 where the vertices are shown on a left lateral angled view of a brain, modelled with outlines from the x, y and z planes.

do not always produce the lowest values (i.e. greatest similarity). A superior similarity metric for this case was identified as Graph Edit Distance (GED), which is an inexact graph matching method defined as the cost of the least expensive sequence of edit operations that are needed to transform one graph into another[37]. Specifically, the results of *edge GED*[38] are displayed in Fig. 5a for the scan 1 and 2 (scan and rescan) comparisons. GED has been used as an identification method for matching fingerprints[37], but it fails to identify subject 113 from their scan-rescan comparison. The CDI based comparisons are shown in Fig. 5c,d for three and ten input eigenvectors respectively. The CDI communities with 3 input eigenvectors are also displayed in three-dimensional space (Fig. 5d) for subject 113, with the difference in community density providing an insight into why edge GED failed but CDI succeeded in recognising subject 113. The paths taken by communities in Fig. 5d are mostly similar with one non-matching community present from scan 2. However, the density of these communities are significantly different with far more vertices in the scan 1 communities than those from scan 2. This also translates to the density of the connectivity network, where scan 1 has 841,097 vertices with a non-zero outdegree and scan 2 only has 647,049 vertices for subject 113. This difference of 194,048 vertices is far larger than any other scan-rescan comparison, where the next largest difference in 25,554 vertices (mean number of non-zero outdegree vertices is 836,699 for the other scans). This difference in graph density appears to prevent edge GED from detecting a match.

The CDI community matching, with ten input eigenvectors (Fig. 5c), presents a similar accuracy to edge GED (Fig. 5a). But it is notable that 3 eigenvector CDI produces some poor matches in Fig. 5c, such as subject 814 and 849. This suggests that the most influential communities have changed between scan and rescan for these subjects as they are still recognisable when considering the less influential communities detected when using ten eigenvectors.

Similar results to Fig. 5c can be obtained by taking an approach based on normalised cut[11], where the Fiedler vector divides the graph by the sign of the vector's entries. This spectral bisection approach, as described in the

Methods section, does not manage to clearly identify all subjects when applying the *mean number of matching communities* procedure. It also creates more off-diagonal false matches than Fig. 5c and does not give insight into any changes in neuronal influence.

Discussion

In this article a method for detecting Communities of Dynamical Influence (CDI) is proposed that uses the relationship between a selection of the most dominant left eigenvectors of the Laplacian to identify network divisions. The communities are shown to be led by the most influential vertices by comparing with a perturbation optimiser that maximises the network convergence rate to consensus. CDI defines community divisions that can be similar to those detected by k-means clustering, but user selection is required to define k. In contrast, with CDI the number of community divisions are a product of the network topology and the number of input eigenvectors. Three input eigenvectors are shown to produce consistently good results when using CDI-based consensus optimisation, which is used herein as an assessment of the quality of the influential communities that are found. The selected number of input eigenvectors is a trade-off, for CDI, as more eigenvectors may highlight more or new communities but those communities may no longer represent the communities that form through association with an influential leader. There is scope to consider in future work weighting the contribution of less dominant eigenvectors to ensure high accuracy community division that is still focused on the most influential leaders.

For a starling flock model, the CDI reveal the benefit to starlings of maintaining a low outdegree. Higher outdegrees were seen to reduce the responsiveness of the network, in the CDI-based perturbation optimisations, with the flock also becoming composed of fewer CDI. It was also seen that the most influential vertices became more centrally located within the flock, where they are unlikely to detect an incoming predator. The CDI did not reveal any optimality in the chosen starling outdegree, instead it was suggested $k = 7$ may have emerged as a compromise between ensured connectivity and fast response.

In this article a series of human brain networks, around 850,000 vertices in size, were analysed to identify neuronal communities. The identified communities enabled separate MRI scans to be clearly recognised as belonging to the same subject, especially when using CDI with ten input eigenvectors. We conjecture that the subjects with the lowest mean number of matching communities, when using CDI with the first three eigenvectors, are those that display the greatest change in neuronal activity. Since these same subjects are clearly identified when considering the less influential communities detected by the 10 eigenvector CDI. Edge Graph Edit Distance is shown to be highly effective in identifying the scan similarity for the majority of matches with one exception. This exception highlights the effectiveness of the CDI approach, where pathways/community similarity was still clear even when comparing graphs and communities of significantly different sizes.

Methods

A graph is defined as $G = (V, E)$, where there is a set of V vertices and E edges, which are unordered pairs of elements of V for an undirected graph and ordered pairs for a directed graph.

The adjacency matrix, A, is a square $n \times n$ matrix when representing a graph of n vertices. This matrix captures the network's connections where $a_{ij} > 0$ (a_{ij} is the ijth entry of the graph's adjacency matrix) if there exists a directed edge from vertex i to j and 0 otherwise. Variable edge weights contain information on the relative strength of interactions, whilst uniform edge weighting either only represents the presence of a connection or is a result of all the edges having the same information carrying capacity. For an undirected graph, the adjacency matrix is symmetric with an edge $(i, j) \in E$ resulting in $a_{ij} = a_{ji} > 0$. For a directed graph, the indegree is equal to the column sum, $\sum_i a_{ij}$, and outdegree is equal to the row sum, $\sum_j a_{ij}$.

The Laplacian matrix is defined as $L = D - A$ where the degree matrix, D, is a diagonal matrix and the ith diagonal element is equal to the outdegree of vertex i. The first eigenvalue of the adjacency matrix (λ_1), referred to as the *Spectral Radius*, is the largest eigenvalue in magnitude and is associated with the *Perron vector*, which is an eigenvector that contains only positive entries. Whereas for the Laplacian matrix the first eigenvalue is associated with the eigenvalue $\lambda_1 = 0$[26]. For a directed graph, the left eigenvectors of the Laplacian matrix, $\mathbf{v}_L$, are row vectors satisfying $\mathbf{v}_L L = \lambda \mathbf{v}_L$.

Communities of dynamical influence. The Communities of Dynamical Influence (CDI) are found by analysing the left eigenvectors of the Laplacian matrix as presented in Algorithm 1. The algorithm defines a coordinate system using only the Real part of the selected eigenvectors for a chosen number of input eigenvectors. Hence, if any of the eigenvectors considered form a complex conjugate pair then the algorithm will only ever use one vector from the pair and then choose the next dominant eigenvector not in the pair, since the real part of the complex conjugates will be identical.

The CDI are based on first identifying the most effective community leaders. These vertices do not "follow" nodes with greater community influence, therefore they are identified as those with no outward connections to a vertex that is further from the origin in this eigenvector-based coordinate system.

A vertex is assigned to a community when there is a directed path from that vertex to a community leader, with each vertex on the path further from the origin of the eigenvector coordinate system than the previous. The number of communities is equal to the number of community leaders where a vertex can belong only to one community. If a vertex is assigned to multiple communities, then it is kept in the community where it is most aligned with the community leader and removed from the others. This alignment is determined by comparing the position vector of the vertex with respect to those of the most influential vertices using the scalar product.

Perturbation driven consensus. The networks considered herein have n agents connected via local communication with a static, time-invariant, topology. A uniform signal $u = \mathrm{u}[1, 1,\ldots, 1]^{\mathrm{T}} \in \mathbf{IR}^n$ is supplied to all agents with different positive gains c_i, where $i = 1, 2,\ldots, n$. The dynamics of this system are defined as

$$\dot{x}_i = \sum_{j=1}^{n} a_{ij}(x_j - x_i) + c_i(u - x_i) \tag{1}$$

where x_i is the state of the i^{th} agent and u is the scalar target value that all agents must achieve. The resource allocation, c_i, ranges from 0 to 1, is globally bounded as $\sum_i c_i = 1$, and scales the comparison between the uniform input signal, u, and the current state x_i.

The global dynamics of the network can be expressed with respect to the Laplacian matrix as

$$\dot{\mathbf{x}} = -L\mathbf{x} + C(\mathbf{u} - \mathbf{x}) \tag{2}$$

where C is the perturbation matrix, $C \leftarrow \mathrm{diag}\,(\boldsymbol{c}) = \mathrm{diag}\,(c_1, \ldots, cp)$. Spanning trees have been highlighted previously as a condition for consensus[39–41] since for a directed network G, defined by Eq. (2), consensus will eventually be achieved if all agents are reachable, via directed edges, from the vertices supplied with perturbation input.

Algorithm 1. Detecting Communities of Dynamical Influence (CDI).

Set y as the number of input eigenvectors
$\mathbf{v}_{Li}$ is the i^{th} eigenvector of the normalised Laplacian matrix , $L \in \mathbb{R}^{n\times n}$
$\boldsymbol{v}_1 \leftarrow \mathbb{R}[\mathbf{v}_{L1}]$, $k \leftarrow 1$, $j \leftarrow 2$
while $k < y$ **do**
  if $\mathbb{R}[\mathbf{v}_{Lj}] \neq \boldsymbol{v}_t \; \forall\, t = 1,\ldots,k$ **then**
    $k \leftarrow k + 1$
    $\boldsymbol{v}_k \leftarrow \mathbf{v}_{Lj}$ (i.e. include only one eigenvector from any complex conjugate pair)
  end if
  $j = j + 1$
end while
$\boldsymbol{e} \leftarrow [\boldsymbol{v}_1, .., \boldsymbol{v}_y]$, $\mathbf{s} \leftarrow |\boldsymbol{e}|$, $h \leftarrow 1$
for $j \leftarrow 1$ **to** n **do**
  $\boldsymbol{o}$ is the set of k vertices connected to vertex j where $L(j, \boldsymbol{o}[t]) < 0 \; \forall\, t \leftarrow 1$ to k
  if $\boldsymbol{o} \leftarrow \emptyset$ (empty set) **then**
    $s_o \leftarrow 0$
  end if
  if $s_j > s_{\boldsymbol{o}[t]} \forall\, t \leftarrow 1$ **to** k **then**
    $\beta_h \leftarrow j$ is a community leader vertex
    $h \leftarrow h + 1$
  end if
end for
for $j \leftarrow 1$ **to** h **do**
  $\mathbf{c}_j$ is a list of d vertices that lie on a directed path leading to β_j and composed of edges where s increases, i.e. $s_{source} < s_{destination}$
end for
for $j \leftarrow 1$ **to** n **do**
  $z_k \leftarrow (e_{\beta_k} \bullet e_j)/s_{\beta_k} \; \forall\, k$ where $j \in \mathbf{c}_k$ (i.e. scalar projection of vertex j on leader vertices β_k)
  Remove vertex j from $\mathbf{c}_k \; \forall\, k$ where $z_k \neq \max_k z_k$
end for
$\mathbf{c}' \leftarrow \mathbf{c}_i$ where i is a an ordered list of communities, descending order, according to their largest $\boldsymbol{v}_1$ value, eg. $j \in \mathbf{c}'_1$ where $\boldsymbol{v}_1[j] = \max \boldsymbol{v}_1$

Perturbation optimisation. A perturbation optimisation is presented as a method for validating the CDI algorithm's ability to identify the most influential communities and network leaders. The objective function of this optimisation is to maximise the system's convergence to consensus, by applying a globally bounded perturbation to the vertices. It has been demonstrated that, by changing the coordinates, Eq. 2 can be written as

$$\frac{d\mathbf{y}}{dt} = -\,(L + C)\mathbf{y}. \tag{3}$$

where the diagonal elements of C can be optimised to maximise the magnitude of $\lambda_1(-(L + C))$, which is the most dominant (rightmost) eigenvalue (i.e. eigenvalue with the largest real part) of the negated and perturbed Laplacian matrix[26].

The first step is to optimise a perturbation only applied to most influential vertex from each community, according to their $\mathbf{v}_{L1}$ value, as detailed in Algorithm 2. If the optimiser reduces the perturbation applied to the influential vertex to less than or equal to zero then the community is discarded from the optimisation. Once the optimiser has converged, only the communities associated with influential vertices that still have positive perturbations are included in the next step of the optimisation.

For each of the selected CDI from Algorithm 2, an input vector $\boldsymbol{\omega}_i$ is created for each CDI with entries populated with their $\mathbf{v}_{L1}$ entries if the vertex is in the community and values set to zero otherwise. These vectors are

then manipulated, using the *Power Optimisation* method[31] in Algorithm 4, and combined to produce the final optimised perturbation, with weighting variables used to determine the ratio of each vector's contribution in Algorithm 3.

The Power Optimisation focuses resources on the most effective leaders by raising an eigenvector to a power, η_i, for a given input vector, $\boldsymbol{\omega}_i$, according to

Algorithm 2. Community Leader Optimisation.

1: Set $\boldsymbol{v}_1 \leftarrow \mathbb{R}[\mathbf{v}_{L1}]$
2: Detect m ordered Communities ($\mathbf{c}'_i$) using Algorithm 1.
3: $\mathbf{c}'_i \subseteq V$ is the set of vertices belonging to community i, where $i \leftarrow 1$ **to** m
4: **for** $i \leftarrow 1$ **to** m **do**
5: $\quad \varepsilon_i \leftarrow \max_t \boldsymbol{v}_1[t] \; \forall \; t \in \mathbf{c}'_i$
6: $\quad \beta_i \leftarrow j$ where $\boldsymbol{v}_1[j] = \max_t \boldsymbol{v}_1[t]$
7: **end for**
8: Sort $\boldsymbol{\varepsilon}$ in descending order, $\boldsymbol{\varepsilon}' \leftarrow \{\varepsilon_{r_1}, \varepsilon_{r_2}, ..., \varepsilon_{r_m}\}$, where $\boldsymbol{r}$ is the ordered index vector such that $\varepsilon_{r_1} = \max_t \boldsymbol{v}_1[t] \; \forall \; t \in V$
9: Set perturbation vector $\mathbf{p}[1, ..., n] \leftarrow 0$ where n is the number of vertices (V)
10: **for** $i \leftarrow 1$ **to** m **do**
11: $\quad \mathbf{p}[\beta_i] \leftarrow \frac{1}{i}$
12: **end for**
13: Maximise $|\lambda_1(-(L+C))|$ where $C = \mathrm{diag}(\mathbf{p})$, by adjusting $\mathbf{p} \; \forall \; p_i > 0$ where $i \leftarrow 1$ **to** n
14: If $\mathbf{p}[\beta_i] \leq 0 \; \forall \; i \leftarrow 1$ **to** m, set $\mathbf{p}[\beta_i] \leftarrow 0$ and repeat Step 13, until optimiser convergence.

$$\mathbf{p}_i = \frac{\boldsymbol{\omega}_i^{\eta_i}}{\sum(\boldsymbol{\omega}_i^{\eta_i})} \tag{4}$$

where $\boldsymbol{\omega}_i^{\eta_i}$ indicates an element-wise operation and the denominator ensures that $\sum(\boldsymbol{\omega}_i^{\eta_i})$. When $\eta \rightarrow 0$ the vector, p_i, approaches a uniform vector state. As η is increased, the Power Optimisation method iteratively reduces the value of the smaller vector elements while increasing the value of the larger elements.

In the following equation t power optimised input vectors, p_i, are combined using weighting variables, $\boldsymbol{r} = \{r_1, \ldots, r_t\}$, to produce the optimised perturbation vector as follows

$$\boldsymbol{c} = \frac{\sum_{j=1}^{t} \frac{\mathbf{p}_j}{r_j}}{\sum_{j=1}^{t} \frac{1}{r_j}} \tag{5}$$

where the denominator ensures that $\sum_j (c)_j = 1$. Also note that $C = \mathrm{diag}(\boldsymbol{c})$ in the $-(L = C)$ system.

Algorithm 3 presents the perturbation optimisation procedure, where Eqs. 4 and 5 are used repeatedly with different inputs and constraints. A numerical optimiser, employing a sequential quadratic programming method[42], is used throughout the algorithm to maximise the dominant eigenvalue by optimising the power, $\boldsymbol{\eta} = \{\eta_1, \ldots, \eta_i\}$, and weighting, $\boldsymbol{r} = \{\eta_1, \ldots, \eta_i\}$, variables for the i input vectors. The algorithm first optimises the power variables using the power optimisation method for one input vector. This power is employed for checking if adding any more input vectors and numerically optimising only the new weighting variable will increase the value of $|\lambda_1(-(L + C))|$. If the convergence speed is improved then the new selection of input vectors will have their power variables numerically optimised, before repeating the search for new input vectors and optimising the new weighting variable. Once all input vectors have been checked both the weighting and power variables are optimised numerically.

To check that there are not any redundant input vectors, each vector is removed from the optimisation, starting with the first input vector, to check if that removal increases $|\lambda_1(-(L + C))|$. If removing the vector does not improve the performance then it is reintroduced and the process is repeated for the other community's input vectors. The final combination of input vectors (i.e. communities) are optimised numerically by varying their weighting and power variables to maximise $\lambda_1(-(L + C))$.

Matching brain communities. Each brain connectome graph contains 1,827,240 voxels that each represent a 1 mm3 volume of the brain. The centre of each volume (voxel) forms a three dimensional grid with 1 mm spacing between neighbouring voxels. Each edge in the graph is defined as any two vertices that are connected by at least a single fibre, where an edge of weight 1 would represent a single fibre connection. This results in an undirected network of weighted edges where around half of these voxels have connections in the subjects considered here.

Algorithm 3. Perturbation optimisation using CDI.

Input m CDI communities ($\mathbf{c}'$), perturbation vector ($\mathbf{p}$), leader vertex list ($\boldsymbol{\beta}$) from Algorithm 2, $h \leftarrow 1$, n is the number of vertices (V), $\boldsymbol{v}_1 \leftarrow \mathbb{R}[\mathbf{v}_{L1}]$
for $j \leftarrow 1$ **to** m **do**
  $\boldsymbol{\omega}_h[1,...,n] \leftarrow 0$
  if $\mathbf{p}[\beta_j] > 0$ **then**
    $\boldsymbol{\omega}_h \leftarrow \boldsymbol{v}_1[i]\ \forall\ i \in \mathbf{c}'_j$ (i.e. communities to be included in the optimisation)
    $h \leftarrow h+1$
  end if
end for
Maximise $|\lambda_1(-(L+C))|$, where $C \leftarrow \mathrm{diag}(\mathbf{p})$, by optimising η_1 in Eq. 4
Set $\Lambda \leftarrow |\lambda_1(-(L+C))|$, $\boldsymbol{r} \leftarrow \{1\}$, $\boldsymbol{\eta} \leftarrow \{\eta_1\}$, $\boldsymbol{\omega} \leftarrow \{\boldsymbol{\omega}_1\}$
for $j \leftarrow 2$ **to** h **do**
  q is the current number of input vectors ($\boldsymbol{\omega}$), $\boldsymbol{r} \leftarrow \{\boldsymbol{r}, \boldsymbol{r}[q-1]\}$, $\boldsymbol{\eta} \leftarrow \{\boldsymbol{\eta}, \boldsymbol{\eta}[q-1]\}$, $\boldsymbol{\omega} \leftarrow \{\boldsymbol{\omega}, \boldsymbol{\omega}_j\}$, $\boldsymbol{k} \leftarrow \boldsymbol{r}$
  Maximise $|\lambda_1(-(L+C))|$, where $C \leftarrow \mathrm{diag}(\boldsymbol{c})$, by optimising $\boldsymbol{r}[j]$ in Eq. 4 and 5
  if $|\lambda_1(-(L+C))| * 1.001 < \Lambda$ **then**
    $\boldsymbol{r} \leftarrow \{k_1,...,k_{q-1}\}$, $\boldsymbol{\eta} \leftarrow \{\eta_1,...,\eta_{q-1}\}$, $\boldsymbol{\omega} \leftarrow \{\boldsymbol{\omega}_1,...,\boldsymbol{\omega}_{q-1}\}$ (i.e. remove input vector $\boldsymbol{\omega}_j$ and parameters)
  else if $|\lambda_1(-(L+C))| * 1.001 \geq \Lambda$ **then**
    Maximise $|\lambda_1(-(L+C))|$, where $C \leftarrow \mathrm{diag}(\boldsymbol{c})$, by optimising η where $\boldsymbol{\eta} \leftarrow \{\eta,...,\eta\}$ in Eq. 4 and 5
    Set $\Lambda \leftarrow |\lambda_1(-(L+C))|$
  end if
end for
Maximise $|\lambda_1(-(L+C))|$ where $C = \mathrm{diag}(\mathbf{p})$ by optimising $\boldsymbol{r}$ and $\boldsymbol{\eta}$ in Eq. 4 and 5
Set $\Lambda \leftarrow |\lambda_1(-(L+C))|$, q is the current number of input vectors ($\boldsymbol{\omega}$)
for $i = 1$ **to** q **do**
  $\boldsymbol{k} \leftarrow \boldsymbol{r}$, then $\boldsymbol{r}[i] \leftarrow \infty$ (i.e. essentially removing input vector $\boldsymbol{\omega}_i$)
  Calculate $|\lambda_1(-(L+C))|$ where $C = \mathrm{diag}(\mathbf{p})$ with $\boldsymbol{r}$ and $\boldsymbol{\eta}$ in Eq. 4 and 5
  if $|\lambda_1(-(L+C))| * 1.001 < \Lambda$ **then**
    $\boldsymbol{r} \leftarrow \boldsymbol{k}$
  else if $|\lambda_1(-(L+C))| * 1.001 \geq \Lambda$ **then**
    Set $\Lambda = |\lambda_1(-(L+C))|$.
  end if
end for
$\boldsymbol{k} \leftarrow \emptyset$, $\boldsymbol{q} \leftarrow \emptyset$, $\boldsymbol{w} \leftarrow \emptyset$
for $i = 1$ **to** q **do**
  if $\boldsymbol{r}[i] \ll \infty$ **then**
    $\boldsymbol{k} \leftarrow \{\boldsymbol{k}, r_i\}$, $\boldsymbol{q} \leftarrow \{\boldsymbol{q}, \eta_i\}$, $\boldsymbol{w} \leftarrow \{\boldsymbol{w}, \boldsymbol{\omega}_i\}$
  end if
end for
$\boldsymbol{r} \leftarrow \boldsymbol{k}$, $\boldsymbol{\eta} \leftarrow \boldsymbol{q}$, $\boldsymbol{\omega} \leftarrow \boldsymbol{w}$
Maximise $|\lambda_1(-(L+C))|$ where $C = \mathrm{diag}(\mathbf{p})$ by optimising $\boldsymbol{r}$ and $\boldsymbol{\eta}$ in Eq. 4 and 5

For the large brain connectome graphs, the adjacency matrix was employed, rather than the Laplacian, to identify CDI. This was due to the difficulty that emerged in converging on $\lambda_1 = 0$ for such large matrices that contained more than one near zero eigenvalue. Both adjacency and Laplacian matrices can be used with the procedure in Algorithm 1. The left eigenvectors of these matrices are the same in certain cases, including graphs with a constant outdegree for each vertex. In other cases the eigenvectors vary but the first eigenvector contains all positive entries for both, while the following eigenvectors divide the network in a similar manner. It should also be noted that due to the undirected nature of the connectome data, the Laplacian matrix's ability to highlight the imbalance between outdegree and indegree is less relevant. In fact, for an undirected Laplacian matrix the first eigenvector is uniform with an imbalance in the indegree and outdegree of vertices required to determine influence from this eigenvector.

The vertices included in influential communities from different graphs were compared to determine similarity. For this similarity comparison, the CDI were reduced in size by only including vertices with a large eigenvector entry. This eigenvector entry threshold was set at 0.01 (i.e. $(v_A)_i > 0.01$ where v_A is any of the eigenvectors used in the CDI coordinate system). The similarity of two graphs was assessed by calculating the number of matching communities shared between both graphs. This comparison metric for assessing community matches, developed here, considers the shortest distance from all the vertices of one community to the nearest vertex that belongs to another. Vertices were considered overlapping if they are from the same voxel or they are in an adjacent voxel (i.e. maximum overlapping voxel distance $\sqrt{3} \approx 1.74$ mm). The percentage of overlapping vertices are calculated for each community comparison to find the highest percentage overlap between two communities in separate scans. The communities appear to reveal pathways in the brain as depicted in Fig. 5b. These pathways can sometimes vary in density of vertices and in length, which makes an exact match between two communities unlikely. Therefore, for a pair of communities to be considered a match their percentage overlap had to exceed a threshold value. The *mean number of matching communities* was determined by taking the mean number of matches from a range of threshold values between 50% and 90%, at 10% intervals. Note that any community can only be a member of one matching pair, i.e. if a community in one scan 1 matches with multiple communities in another scan only one of those matching pairs would be considered for the number of matching communities.

Finally, it is worth noting that there are always errors in the images produced from MRI scans, even when using the same equipment and procedures, with small errors occurring because of slight changes in image orientation and magnetic field instability[43]. The *mean number of matching communities* is, therefore, also able to accommodate any small positional errors when detecting overlapping communities.

Spectral bisection. The Fiedler vector is associated with the second smallest eigenvalue of the Laplacian matrix but the second eigenvector of the adjacency matrix, used here for the analysis of brain networks, also divides the network in a similar manner. These spectral bisections are completed three times to create 8 communities by first dividing the network according to the second eigenvector, with the sign of its entries determining community division. The second eigenvector is then assessed for both of these communities and more divisions made. For the final bisection of four communities into eight, the second eigenvector was used unless it did not generate two communities with values higher than the threshold used when assessing the *mean number of matching communities*, described previously. In this case the next eigenvector that divides the community, so that both divisions had values above the threshold, is selected. This ensures that all scans have eight eigenvectors with which to compare.

Received: 26 June 2019; Accepted: 18 October 2019;
Published online: 26 November 2019

References

1. IDG. 2018 cloud computing survey. https://www.idg.com/tools-for-marketers/2018-cloud-computing-survey/ (2019).
2. Henry, C. Amazon planning 3,236-satellite constellation for internet connectivity. https://spacenews.com/amazon-planning-3236-satellite-constellation-for-internet-connectivity/ (2019).
3. I., Andrea, C., Chrysostomou & G., Hadjichristofi Internet of things: Security vulnerabilities and challenges. In *2015 IEEE Symposium on Computers and Communication (ISCC)*, 180–187 (IEEE, 2015).
4. U., Braun, S. F., Muldoon & D. S., Bassett On human brain networks in health and disease. *eLS* (2015).
5. Mišić, B., Goñi, J., Betzel, R. F., Sporns, O. & McIntosh, A. R. A network convergence zone in the hippocampus. *PLoS computational biology.* **10**, e1003982 (2014).
6. Bacik, K. A., Schaub, M. T., Beguerisse-Daz, M., Billeh, Y. N. & Barahona, M. Flow-based network analysis of the caenorhabditis elegans connectome. *PLoS computational biology* **12**, e1005055 (2016).
7. Malliaros, F. D. & Vazirgiannis, M. Clustering and community detection in directed networks: A survey. *Physics Reports* **533**, 95–142 (2013).
8. Leicht, E. A. & Newman, M. E. Community structure in directed networks. *Physical review letters* **100**, 118703 (2008).
9. Page, L., Brin, S., Motwani, R. & Winograd, T. The pagerank citation ranking: Bringing order to the web. *Tech. Rep*., Stanford InfoLab (1999).
10. Pons, P. & Latapy, M. Computing communities in large networks using random walks. In *International symposium on computer and information sciences*, 284–293 (Springer, 2005).
11. Shi, J. & Malik, J. Normalized cuts and image segmentation. *Departmental Papers (CIS)* 107 (2000).
12. Bradley, P. S. & Fayyad, U. M. Refining initial points for k-means clustering. In *ICML*, vol. 98, 91–99 (Citeseer, 1998).
13. Von Luxburg, U. A tutorial on spectral clustering. *Statistics and computing* **17**, 395–416 (2007).
14. Klemm, K., Serrano, M. Á., Eguluz, V. M. & San Miguel, M. A measure of individual role in collective dynamics. *Scientific reports* **2**, 292 (2012).
15. Bonacich, P. Some unique properties of eigenvector centrality. *Social networks* **29**, 555–564 (2007).
16. Seidman, S. B. Network structure and minimum degree. *Social networks* **5**, 269–287 (1983).
17. Li, R.-H., Qin, L., Yu, J. X. & Mao, R. Finding influential communities in massive networks. *The International Journal on Very Large Data Bases* **26**, 751–776 (2017).
18. Zhan, J., Guidibande, V. & Parsa, S. P. K. Identification of top-k influential communities in big networks. *Journal of Big Data* **3**, 16 (2016).
19. Li, J. *et al*. Most influential community search over large social networks. In *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*, 871–882 (IEEE, 2017).
20. Stanoev, A., Smilkov, D. & Kocarev, L. Identifying communities by influence dynamics in social networks. *Physical Review E* **84**, 046102 (2011).
21. Ballerini, M. *et al*. Interaction ruling animal collective behavior depends on topological rather than metric distance: Evidence from a field study. *Proceedings of the national academy of sciences* **105**, 1232–1237 (2008).
22. Young, G. F., Scardovi, L., Cavagna, A., Giardina, I. & Leonard, N. E. Starling flock networks manage uncertainty in consensus at low cost. *PLoS computational biology* **9**, e1002894 (2013).
23. Attanasi, A. *et al*. Emergence of collective changes in travel direction of starling flocks from individual birds' fluctuations. *Journal of The Royal Society Interface* **12**, 20150319 (2015).
24. Herbert-Read, J. E., Buhl, J., Hu, F., Ward, A. J. & Sumpter, D. J. Initiation and spread of escape waves within animal groups. *Royal Society open science* **2**, 140355 (2015).
25. Landman, B. A. *et al*. Multi-parametric neuroimaging reproducibility: a 3-t resource study. *Neuroimage* **54**, 2854–2866 (2011).
26. Punzo, G., Young, G. F., Macdonald, M. & Leonard, N. E. Using network dynamical influence to drive consensus. *Scientific reports* **6**, 26318 (2016).
27. Fitch, K. & Leonard, N. E. Information centrality and optimal leader selection in noisy networks. In *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*. 7510–7515 (IEEE. 2013).
28. Lin, F. Fardad, M. & Jovanović, M. R. Algorithms for leader selection in large dynamical networks: Noise-corrupted leaders. In *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*. 2932–2937 (IEEE. 2011).
29. Patterson, S., McGlohon, N. & Dyagilev, K. Optimal k-leader selection for coherence and convergence rate in one-dimensional networks. *IEEE Transactions on Control of Network Systems* **4**, 523–532 (2017).
30. Gan, Z., Shao, H., Xu, Y. & Li, D. Performance of leader-following consensus on multiplex networks. *Physica A: Statistical Mechanics and its Applications* **509**, 1174–1182 (2018).
31. Clark, R. Punzo, G. & Macdonald, M. Consensus speed optimisation with finite leadership perturbation in k-nearest neighbour networks. In *Decision and Control (CDC), 2016 IEEE 55th Conference on*. 879–884 (IEEE. 2016).
32. MathWorks. Constrained nonlinear optimization algorithms. http://www.mathworks.se/help/optim/ug/constrained-nonlinear-optimization-algorithms.html (2015).
33. Ng, A. Y. Jordan, M. I. & Weiss, Y. On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems*. 849–856 (2002).

34. Balister, P., Bollobás, B., Sarkar, A. & Walters, M. Connectivity of random k-nearest-neighbour graphs. *Advances in Applied Probability* **37**, 1–24 (2005).
35. Roncal, W. G. *et al*. Migraine: Mri graph reliability analysis and inference for connectomics. In *Global Conference on Signal and Information Processing (GlobalSIP), 2013 IEEE*. 313–316 (IEEE. 2013).
36. Zuo, W., Zhang, D. & Wang, K. An assembled matrix distance metric for 2dpca-based image recognition. *Pattern Recognition Letters* **27**, 210–216 (2006).
37. Robles-Kelly, A. & Hancock, E. R. Graph edit distance from spectral seriation. *IEEE transactions on pattern analysis and machine intelligence* **27**, 365–378 (2005).
38. Manrique, R. Cueto-Ramirez, F. & Mariño, O. Comparing graph similarity measures for semantic representations of documents. In *Colombian Conference on Computing*. 162–176 (Springer. 2018).
39. Shao, H. Mesbahi, M. & Xi, Y. The relative tempo of discrete-time consensus networks. In *Control Conference (CCC), 2015 34th Chinese*. 7362–7367 (IEEE. 2015).
40. Jadbabaie, A., Lin, J. & Morse, A. S. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *Automatic Control, IEEE Transactions on* **48**, 988–1001 (2003).
41. Mesbahi, M. & Egerstedt, M. *Graph theoretic methods in multiagent networks* (Princeton University Press. 2010).
42. MathWorks. fminunc unconstrained minimization. http://uk.mathworks.com/help/optim/ug/fminunc-unconstrained-minimization.html (2015).
43. Morey, R. A. *et al*. Scan–rescan reliability of subcortical brain volumes derived from automated segmentation. *Human brain mapping* **31**, 1751–1762 (2010).

Acknowledgements

This work was supported, in part, by the Engineering and Physical Sciences Research Council [EP/L505080/1].

Author contributions

R.C., G.P. and M.M. devised the study; R.C. developed the algorithms, performed the analyses, wrote the paper and prepared the figures; All authors reviewed the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to R.C.

Reprints and permissions information is available at www.nature.com/reprints.

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

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this license, visit http://creativecommons.org/licenses/by/4.0/.

© The Author(s) 2019