SCIENTIFIC REPORTS
natureresearch

OPEN

Reduced network extremal ensemble learning (RenEEL) scheme for community detection in complex networks

Received: 15 May 2019
Accepted: 17 September 2019
Published online: 02 October 2019

Jiahao Guo[1,2], Pramesh Singh[1,2] & Kevin E. Bassler[1,2,3]

We introduce an ensemble learning scheme for community detection in complex networks. The scheme uses a Machine Learning algorithmic paradigm we call Extremal Ensemble Learning. It uses iterative extremal updating of an ensemble of network partitions, which can be found by a conventional base algorithm, to find a node partition that maximizes modularity. At each iteration, core groups of nodes that are in the same community in every ensemble partition are identified and used to form a reduced network. Partitions of the reduced network are then found and used to update the ensemble. The smaller size of the reduced network makes the scheme efficient. We use the scheme to analyze the community structure in a set of commonly studied benchmark networks and find that it outperforms all other known methods for finding the partition with maximum modularity.

Among the most basic and important problems in Network Science is to find the structure within a network[1,2]. One way of doing this is to find the community, or modular structure of the nodes. In many real-world networks, the community structure has been found to control much of their dynamical or functional behavior. Although there are many possible definitions of community[3,4], a commonly used definition assumes that a community is a group of nodes that are more densely connected than what would occur randomly. This intuitively appealing concept of community can be used to define a metric, called *Modularity Q*, that quantifies the extent to which a partition of the nodes of a network is modular[2]. The community structure of a given network can then be obtained by finding the partition of the network's nodes that has the maximum modularity Q_{max}. Finding this partition, however, is an NP-hard problem[5]. It is of considerable interest and importance to develop an algorithm that robustly finds an accurate solution to this optimization problem that completes in polynomial time. The accuracy of a solution can be measured by how close the value Q of the partition found is to the value of Q_{max}. Any solution provides a lower bound estimate of the value of Q_{max}. Thus, the higher a solution's value of Q is, the more accurate it and its estimate of Q_{max} is.

A number of polynomial time complexity algorithms for finding a network partition that enables Q_{max} to be estimated have been proposed. Some are quite fast, such as random greedy agglomeration[6–8] and the Louvain method[9]. These algorithms, however, don't generally find very accurate solutions. Far more accurate solutions can generally be found with spectral clustering algorithms[10,11] that iteratively bisect the set of nodes. The most accurate algorithm of this type[12] combines bi-sectioning based on the eigenvector of largest eigenvalue of the modularity matrix[10], tuning with generalized Kernighan–Lin refinements[13,14], and agglomeration. Until recently this was the most accurate algorithm known. Virtually all algorithms for maximizing modularity are partially stochastic, as they make random choices at intermediate steps among what are seemingly equivalent options at that point. These choices can affect the final partition, and, thus, different runs can produce different partitions. Because of this, to find the partition that provides the best estimate of the maximum modularity, algorithms are often run multiple times to produce an ensemble of partitions and the best of those partitions is chosen.

It has, however, recently been demonstrated that partitions with even more accurate estimates of Q_{max} can be obtained with a scheme that uses information contained within an ensemble of partitions generated with conventional algorithms. This idea is known as ensemble learning. Its use distinguishes a new class of modularity

[1]Department of Physics, University of Houston, Houston, Texas, 77204, USA. [2]Texas Center for Superconductivity, University of Houston, Houston, Texas, 77204, USA. [3]Department of Mathematics, University of Houston, Houston, Texas, 77204, USA. Correspondence and requests for materials should be addressed to K.E.B. (email: bassler@uh.edu)

maximizing algorithms[15,16]. An ensemble learning scheme known as Iterative Core Group Graph Clustering (CGGCi)[17] was the most accurate algorithm for finding the network partition that maximizes modularity in the 10th DIMACS Implementation Challenge[18]. The CGGCi scheme starts with an ensemble of partitions obtained by using a conventional "base algorithm" and identifies "core groups" of nodes that are grouped together in the same community in every partition in the ensemble. It then transforms the original network into a weighted reduced network by collapsing each of these core groups into a single "reduced" node and summing all link weights between original nodes to assign weights to the links between the reduced nodes. A base algorithm is then used to find an ensemble of partitions of the reduced network, and that ensemble is used to find a new reduced network. This procedure is iterated until no further improvement in Q is found. The best partition of the final reduced network is then mapped back onto the original network to identify the communities.

In this paper, we introduce a different ensemble learning scheme for network community detection. It uses an algorithmic paradigm we call Extremal Ensemble Learning (EEL). Our scheme, which we refer to as Reduced Network Extremal Ensemble Learning (RenEEL), starts with an ensemble of partitions obtained using a conventional base algorithm, and then iteratively updates the partitions in the ensemble until a consensus about which partition is best is reached within the ensemble. To find the partitions used to update the ensemble efficiently, core groups of nodes are identified and used to form a reduced network that is partitioned using a base algorithm. RenEEL then uses a partition of the reduced network to update the ensemble through extremal updating. We will show that an algorithm using the RenEEL scheme improves the quality of community structure discovered, especially for larger networks for which estimating the partition with Q_{max} becomes challenging. Testing our scheme on a wide range of real-world and synthetic benchmark networks, we show that it outperforms all other existing methods, consistently finding partitions with the highest values of Q ever discovered.

Methods

Community detection via modularity maximization. Modularity Q is a metric that quantifies the amount of modular structure there is in a given partition of a network's nodes into disjoint communities $P=\{c_1, c_2, \ldots, c_r\}$, where c_i is the ith community of nodes and r is the number of communities. It is defined as[2]

$$Q = \sum_i \left[\frac{m_i}{m} - \left(\frac{2m_i + e_i}{2m} \right)^2 \right] \tag{1}$$

where the sum is over communities, m_i and e_i are respectively the number of internal and external links of community c_i, and m is total number of links in the network. The first term in Eq. 1 is the fraction of links inside communities, and the second term is the expected fraction if all links of the network were randomly placed. For a weighted network, m_i, e_i and m are sums of link weights instead of numbers of links. Modularity measures the deviation of the structure of a network partition from that expected in a random null model. The *community structure of a network* corresponds to the partition P of its nodes that maximizes Q. The number of communities in P is free to vary. The challenge of detecting the community structure of a network, therefore, is to find the partition with the maximum modularity Q_{max}.

Reduced networks. To find a reduced network G' starting from a network G and an ensemble of partitions of it $\mathcal{P}$, we first identify the core groups in G. A *core group* is a set of nodes that are found together in the same community in every partition in the ensemble. Any node that is not found in the same community with some other node in every partition in $\mathcal{P}$ is itself a core group. G' is then formed by collapsing core groups of nodes into single reduced nodes and combining their links to other nodes by summing their weights. An example of this is shown in Fig. 1. Each circle containing multiple nodes of G that are colored the same in Fig. 1(a) denotes a core group. Two nodes that do not belong to any circle are shown in black and dark green. The core groups are collapsed to reduced nodes of the same color in the reduced network G' shown in Fig. 1(b). The link weights in the reduced network are the sum of link weights between core groups in the original network. The weighted self-loops in G' result from the total internal weights of the core groups in G.

Reduced network extremal ensemble learning scheme. The RenEEL scheme is summarized in the flowchart shown in Fig. 2 and is described as follows. First, an ensemble $\mathcal{P}$ of at most k_{max} partitions P of the network G is obtained from multiple runs of a base algorithm. The base algorithm can be, for example, any of the conventional ones that have been developed to find a partition to estimate Q_{max}. Alternatively, a set of base algorithms can be used to find $\mathcal{P}$. The partitions in $\mathcal{P}$ are then ordered according to their modularity values, from the one with the largest value P_{best} to the one with the smallest value P_{worst}. Next, the core groups of nodes in the ensemble $\mathcal{P}$ are identified and used to construct the reduced network G'. An ensemble $\mathcal{P}'$ consisting of k' partitions P' of G' is then obtained using a base algorithm. The base algorithm used for this step can either be the same as or different from the base algorithm used to find $\mathcal{P}$. The steps in which a base algorithm is used to find the ensembles $\mathcal{P}$ and $\mathcal{P}'$ are shown in red in Fig. 2(a). The partition in $\mathcal{P}'$ with the largest modularity value P'_{best} is then identified and used to perform an extremal update of ensemble $\mathcal{P}$. This step is shown in blue in Fig. 2(a) and detailed in Fig. 2(b). If $Q(P'_{best}) > Q(P_{worst})$, then P'_{best} is expanded into a partition of G and either used in place of P_{worst} in $\mathcal{P}$ (if $k = k_{max}$) or added to the ensemble $\mathcal{P}$ (if $k < k_{max}$) as shown in Fig. 2(b). In doing so $\mathcal{P}$ is enriched with a better quality partition. However, it is possible that at any iteration either P'_{best} is already contained in $\mathcal{P}$, or $Q(P'_{best}) < Q(P_{worst})$. In both cases, in order to move toward consensus within $\mathcal{P}$, its current size k is reduced by 1 by deleting P_{worst} from it. This procedure is repeated until there is only one partition left in the ensemble $\mathcal{P}$. This consensus partition is the partition that has the largest modularity. It can be used to identify the communities of the network, and its modularity Q_{best} estimates Q_{max}.

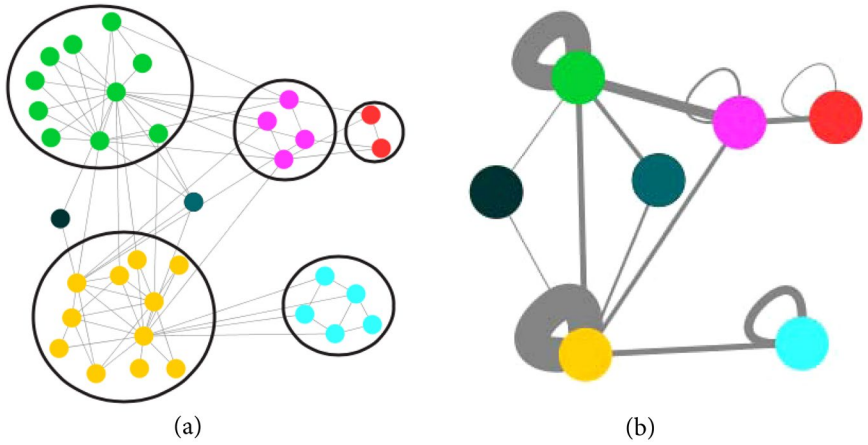

Figure 1. Construction of a reduced network. (**a**) An example network showing seven core groups of nodes. The nodes of the same color belong to the same core group. The nodes inside each of the five circles are collapsed to single nodes in the reduced network, and the two isolated nodes also become nodes in the reduced network. (**b**) The reduced network after collapsing the core groups into single nodes. The nodes in the reduced network are colored according to the core group nodes in the original network and thickness of each link is proportional to its weight.

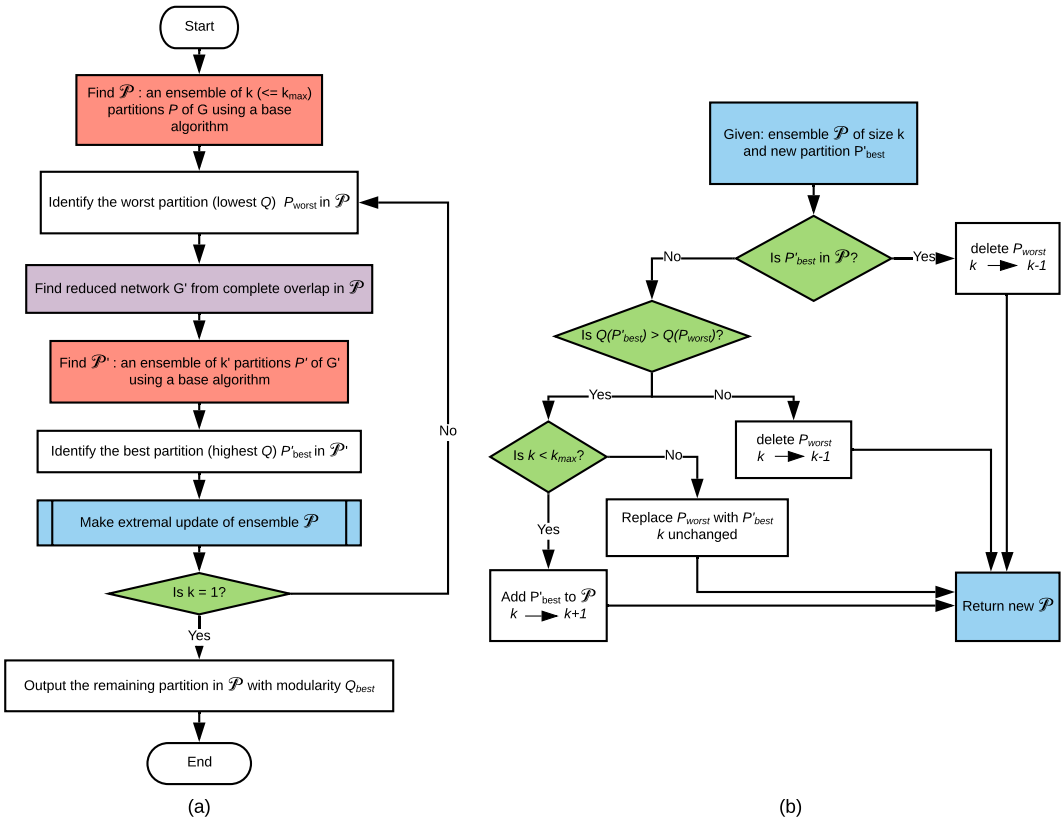

Figure 2. The RenEEL scheme. (**a**) The steps of an efficient ensemble learning scheme to find the network partition that maximizes modularity Q are shown in this flow chart. In the two steps shown in red a base algorithm is used to obtain an ensemble of partitions. The step shown in purple collapses the core groups to find the reduced network. The ensemble $\mathcal{P}$ gets updated with extremal criteria in the step shown in blue and is described in (**b**). The step shown in green guarantees algorithmic termination in a finite network. (**b**) The procedure of the extremal updating of ensemble $\mathcal{P}$.

Computational complexity and practical implementation. The most computationally complex and time consuming steps of the RenEEL scheme are those that use a base algorithm to find an ensemble of partitions. These steps are colored in red in the flowchart in Fig. 2. Assuming that the size of the ensembles $\mathcal{P}$ and $\mathcal{P}'$ are

fixed, the computational complexity of executing these steps is simply a fixed multiple of the computational complexity of the base algorithm used. The scaling of the computational complexity of base algorithms is typically between $\mathcal{O}(n^2)$ and $\mathcal{O}(n^3)$, where n is the number of nodes in the network. All other steps of the scheme have less complexity; the steps of network reduction, colored purple in Fig. 2, and network expansion both have a computational complexity that scales as $\mathcal{O}(n^2)$, and the rest all have a computational complexity that is $\mathcal{O}(1)$. Thus, since each iteration of the scheme has only one step that uses the base algorithm a fixed number of times, each iteration has a computational complexity that scales the same as that of the base algorithm used. As the scheme progresses, however, the size of the reduced network monotonically decreases, significantly increasing the speed of later iterations.

A RenEEL algorithm applied to a finite network is sure to complete since new partitions are added to the ensemble $\mathcal{P}$ only if they have a modularity that is greater than $Q(P_{\text{worst}})$ and the size of $\mathcal{P}$ is bounded. However, it is difficult to determine the precise scaling of number of iterations required in general for an algorithm implementing the scheme to complete, as it depends on the structure of the specific network under consideration. For the networks we analyzed, the number of iterations required was approximately proportional to $k_{\max}$. Thus, we find empirically that the overall complexity of a RenEEL algorithm scales roughly as the base algorithm times k' times $k_{\max}$.

The base algorithm used to obtain the results presented in this paper is a randomized greedy agglomerative hierarchical clustering algorithm[8]. It is commonly used to find the community structure in complex networks[17] and has an expected time complexity that scales as $\mathcal{O}(m \ln n)$[8], where m is the number of links in the network. There can be, at most, $\mathcal{O}(n^2)$ links. The overall complexity of the algorithm used here thus scales approximately as $\mathcal{O}(k_{\max}k'n^2 \ln n)$. The particular choice of parameters $k_{\max}$ and k' is important for the quality of community structure as well as the computational time. In general, higher k' and $k_{\max}$ yield higher Q_{best}.

Co-clustering analysis. In order to visualize the evolution of the clustering results in the RenEEL scheme, co-clustering matrices at various stages of the scheme are shown in Fig. 3. In Fig. 4 the results of the core group co-clustering at the different stages are combined to show their evolution. A co-clustering matrix S is a matrix whose elements s_{ij} are defined as the fraction of times node i and node j are in the same community in an ensemble of partitions $\mathcal{P}$. The order of the nodes in Figs 3 and 4 was determined using simulated annealing to optimize the block-diagonal structure of the matrices. Starting from a random ordering of the nodes, their order was rearranged to minimize a cost function, or "Hamiltonian", that is a function of minimum distance of matrix elements (i,j) from the diagonal d_{ij} assuming periodic boundary conditions on the order:

$$H = \sum_{i<j} s_{ij} \;\; d_{ij}^{\alpha}, \tag{2}$$

where α is an arbitrary factor that controls the non-linear dependence of H on d_{ij}. The results in Figs 3 and 4 were obtained using $\alpha = 3$. Simulated annealing seeks to find the order of nodes that minimizes H. For the Monte Carlo updates in our simulated annealing, Metropolis rates[19] with Boltzmann factor $e^{-(\Delta H)/T}$ were used. Starting from a relatively high temperature where the order of the nodes is random, the temperature was systematically lowered each Monte Carlo step until the node order stabilized.

To get the three co-clustering matrices shown in Fig. 3, which respectively show results at the initial, intermediate, and final stages of the RenEEL scheme, the following procedure was used in the simulated annealing Monte Carlo. First nodes were reordered by considering swaps of random pairs of nodes so as to minimize H in the final stage co-clustering matrix. Then, swaps of pairs of final stage core groups and swaps of pairs of nodes within the final stage core groups were considered to minimize H in the intermediate stage co-clustering matrix. Finally, swaps of pairs of final stage core groups, swaps of pairs of intermediate stage core groups within a finial stage core group, and swaps of pairs of nodes within an intermediate stage core group were considered to minimize H in the initial stage co-clustering matrix. The order of nodes that resulted is used in all three co-clustering matrices in Fig. 3 and in Fig. 4.

Benchmark networks used for comparison. To test the effectiveness of our methods of community detection we studied a set of networks. All of these networks were used in the 10$^{\text{th}}$ DIMACS challenge[18]. The networks are unweighted and undirected. They also have no self-loops. They may be connected or disconnected. The networks we studied are listed and described in Table 1. These networks have been compiled from various sources and cover a wide range of sizes, functions and other characteristics. Hence, they are often used as benchmarks for testing community detection methods. The lists of links defining the Email, Jazz, PGPgc, Metabolic networks were downloaded from ref.[20]. For Adjnoun, Polblog, Netscience, Power, Astro-ph, As-22july06, Cond-mat-2005, they were downloaded from ref.[21]. For Memplus, it was downlaoded from ref.[22]. For Smallworld and CAIDARouterLevel, they were downloaded from ref.[23].

Results

Evolution of core groups. The essence of how the RenEEL scheme works and why it is efficient can be seen by the evolution of the co-clustering of the nodes across the ensemble $\mathcal{P}$. Figure 3 shows the co-clustering results during a typical realization of the scheme on the Email network[24] (see Table 1) at the initial, intermediate and final stages. In the three sub-figures, the intensity with which a pixel (i, j) is colored white corresponds to the frequency that nodes i and j are in the same community in the member partitions of $\mathcal{P}$. The pixels colored blue, red, and yellow indicate that the nodes are in the same community in all member partitions. The nodes in the blue, red, and yellow blocks on the diagonal are the core groups that are used to form the reduced network. Nodes are listed in the same order in each of the three sub-figures. Figure 4 shows the evolution of just the core groups in the same realization.

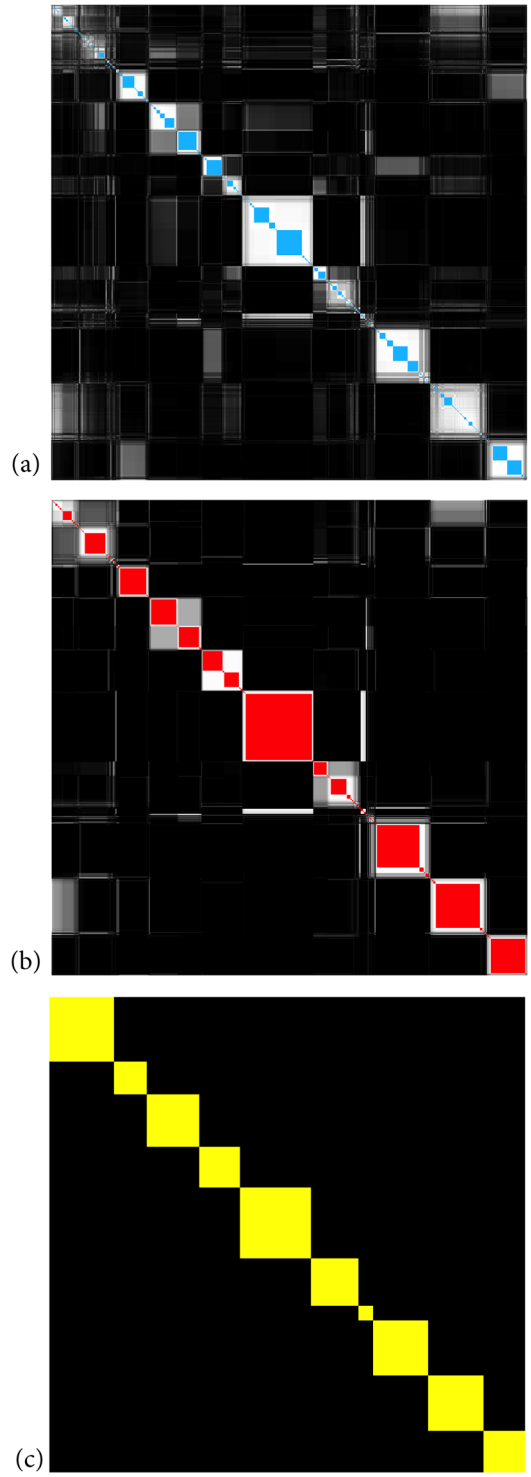

Figure 3. Ordered co-clustering matrix with core groups. Co-clustering matrix after the nodes have been reordered by simulated annealing. (**a**) after the first iteration (**b**) at the intermediate stage (**c**) at completion. The intensity of white in each pixel is proportional to the co-clustering frequency of the corresponding pair of nodes, except when the pair of nodes are always grouped together and, thus, belong to the same core group. In that case the pixel is colored blue in (**a**), red in (**b**), and yellow in (**c**).

The Email network has $n = 1133$ nodes. Initially, as shown in Fig. 3(a), there are 446 core groups, most of which contain only one or two nodes. After 100 iterations of the scheme, as shown in Fig. 3(b), the number of core groups is reduced to 192. Finally, in the stable state, after about 300 iterations of the scheme, only 10 core groups remain, as shown in Fig. 3(c). This reduction, from the original network of 1133 nodes to a reduced network of 10 nodes, is a tremendous simplification and greatly improves the overall speed of network clustering.

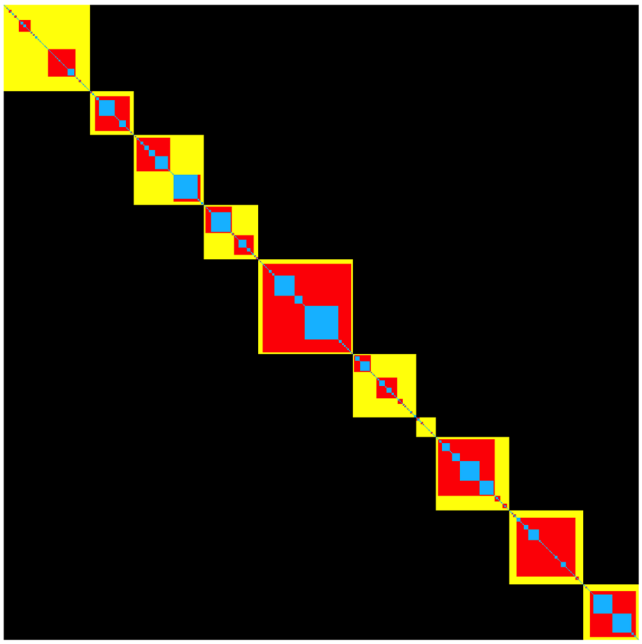

Figure 4. Growth of core groups. Colors blue, red, and yellow represent the core groups after the first iteration, at an intermediate stage, and at the end when the core groups have reached a stable state, respectively. The core groups can only grow. The process is agglomerative.

Within a network G it is generally "easy" to determine that certain groups of nodes should be clustered together. All partitions group them together. These are the core groups of nodes. The hard work in finding the optimal partition is to determine whether nodes that are grouped together in only some of the partitions should indeed be in the same community, that is, to determine whether or not core groups should combine. This is precisely what RenEEL focuses on. The formation and evolution of core groups in RenEEL is an agglomerative process[25]. Once a core group is formed, RenEEL never subsequently divides it. As the scheme progresses, core groups grow and merge with each other and the number of core groups monotonically decreases.

Evolution of the ensemble $\mathcal{P}$. A defining characteristic of RenEEL is that the ensemble of partitions $\mathcal{P}$ evolves as the scheme progresses. The ensemble "learns" what the partition with Q_{best} is by using extremal updating to incorporate new partitions, replace existing ones with higher quality ones, or remove low quality partitions. The new partitions are partitions of the reduced network G'. They are used in RenEEL to improve the quality of $\mathcal{P}$ at every iteration of the scheme until a consensus is reached about what the optimal partition is.

A typical way that $\mathcal{P}$ evolves as the scheme progresses can be seen with the results shown in Fig. 5 from an example run of RenEEL that partitions the As-22july06 network[21]. (See Table 1). In this example run, $k_{\max} = 100$ and $k' = 20$. Figure 5(a,b) show the modularity value Q of P_{best} the best partition in $\mathcal{P}$ (red dots), of P_{worst} the worst partition in $\mathcal{P}$ (black dots), and of P'_{best} the new partition of G' considered for the enrichment of $\mathcal{P}$ (blue dots) as a function of the number of iterations. The main panel of Fig. 5(a) shows the full results of the scheme, from start to finish. An enlarged view of the results for the initial 150 iterations is shown in the inset of Fig. 5(a). The main panel of Fig. 5(b) shows an enlarged view of the vertical Q axis near the final result of the entire scheme. An enlarged view of both axes at the end stages of the scheme is shown in the inset of Fig. 5(b). Figure 5(c) shows the size of the reduced network, or equivalently the number of core groups, as a function of the number of iterations. The main panel of Fig. 5(c) shows the results on linear axis scales, and the inset shows the same results on log scales. Figure 5(d) shows the ensemble size k as function of the number of iterations.

In the example run, as can be seen from the inset of Fig. 5(a), for the first 100 iterations the modularity of the new partitions $Q(P'_{\text{best}})$ are all significantly better than that of the worst in the ensemble $Q(P_{\text{worst}})$. In fact, all the first 100 new partitions generated by RenEEL are better than every one the 100 original ones in $\mathcal{P}$ generated by the base algorithm. (The number of partitions in $\mathcal{P}$ initially is $k_{\max} = 100$). So, for the first $k_{\max}$ iterations RenEEL systematically replaced each of the original partitions. There is large increase in $Q(P_{\text{worst}})$ at iteration 100. Although it's difficult to see in the figure, there are other similar, significant increases in $Q(P_{\text{worst}})$ at iterations 200 and 300, indicating that RenEEL also replaces its first and second 100 new partitions with entirely new sets in the second and third 100 iterations, respectively. After the first 300 iterations, the quality of the new partitions starts to become comparable to the existing partitions. Throughout the process, the $Q(P_{\text{best}})$ intermittently raises when a new best partition is discovered.

Figure 5(c) shows that the size of the reduced network keeps decreasing as the scheme progresses. It initially decreases exponentially, then there is what appears to be a power-law decay from iteration 100 to iteration 1000 (see inset of Fig. 5(c)), followed by a sharp, perhaps exponential, decay in the final iterations of the scheme. The

Network	Node description	Link description
Adjnoun[10]	the most commonly occurring adjectives and nouns in the novel "David Copperfield" by Charles Dickens	pair of words that occur in adjacent position in the text of the book
Jazz[33]	musician	collaboration
Metabolic[34–36]	metabolites (e.g., proteins) (in C. elegans)	interaction between them
Email[24]	members	email interchanges
Polblog[37]	weblogs on US politics	hyperlink
Netscience[10]	scientists working on network theory and experiment	coauthorship
Power[38]	either a generator, a transformator or a substation	power supply line
PGPgc[39]	users of the Pretty Good Privacy (PGP) algorithm	interaction
Astro-ph[40]	scientists	coauthorship in preprints on the Astrophysics E-Print Archive between Jan 1, 1995 and December 31, 1999.
Memplus[41]	memory circuit elements	connections
As-22july06[21]	autonomous systems	data connection
Cond-mat-2005[40]	scientists	coauthorship in preprints on the Condensed Matter E-Print Archive between Jan 1, 1995 and March 31, 2005.
Smallworld[38]	synthetic	synthetic
CAIDARouterLevel[42]	routers	links

Table 1. Benchmark networks. A list of empirical and synthetic networks frequently used for benchmarking modularity optimization methods.

original size of this network, $n = 22963$, is reduced to 38 core groups at the termination step. The size of the ensemble, shown in Fig. 5(d), varies when new partitions are discovered and added to $\mathcal{P}$ or when low quality partitions are deleted as the scheme drives $\mathcal{P}$ toward consensus. The plot shows that as the ensemble learns, its size grows and shrinks multiple times before its size falls to unity and the scheme terminates. There are two main periods in which the size of the ensemble grows, one beginning at about iteration 900 and the other at about iteration 1200. During these periods the value of $Q(P_{\text{best}})$ increases quickly, as can be seen in the main panel and inset of Fig. 5(b). These are periods when the ensemble $\mathcal{P}$ has made a "breakthrough" by discovering a new set of high quality partitions. The example run ends with a consensus choice that a partition with modularity $Q_{\text{best}} = 0.678579$ is the one that maximizes modularity for this network, a value higher than that any previously reported partition. (See Table 2).

Distribution of results for Q_{best}. Since virtually all conventional algorithms are stochastic, ensemble learning schemes that use them as base algorithms will also be stochastic. Thus, a range of results for Q_{best} are possible with each realization of virtually all methods of modularity maximization. As an example, Fig. 6 shows the distribution of Q_{best} that three different methods of community detection produce for the Email network. Results from 250 realizations for each method are shown. Results from the RenEEL, CGGCi ensemble learning schemes, and naive ensemble analyses are shown in red, green, and blue, respectively. The results for all three of these schemes were obtained using a randomized greedy algorithm as the base algorithm and an ensemble size of $k_{\max} = 100$. Each of the blue data points were obtained by running the algorithm 100 times and choosing the largest value from those runs. The distributions from the three different methods are all non-overlapping, with the RenEEL results having the largest values, followed those of CGGCi and then those of the naive ensemble analyses with the conventional algorithm. The distribution of Q_{best} for RenEEL is also narrower than those of the other two schemes, which suggests that the results from RenEEL are close to the value of $Q_{\max}$ for the network.

Application to benchmark networks. To test the accuracy of the RenEEL scheme, we applied it to the benchmark networks listed in Table 1. In Table 2, the maximum modularity value Q_{best} found for these networks by RenEEL is compared to the best previously published values. Many of these values were the best result in the 10$^{\text{th}}$ DIMACS challenge[26]. To be consistent, all realizations had $k_{\max} = 100$ and $k' = 20$ and used the randomized greedy algorithm as a base network. 100 different realizations of RenEEL were run on the smaller networks, up to and including the Netscience network, and 5 were run on the larger networks. For the smaller networks the value of Q_{best} reported in table was consistently obtained. For the larger networks a range of results were obtained and the largest one is listed. As the table shows, the partitions found by RenEEL have a value of Q_{best} that is higher than or equivalent to the best previously reported value for every benchmark network. The difference between Q_{best} found by RenEEL and the previous best values increases with network size. This is due to the fact that for small networks it is generally easier to find the Modularity maximizing partition, but the task becomes more challenging for larger networks.

Our results are significant for every network studied. For the smallest networks, our best partition has the same modularity as that of the previous best result. This is presumably because we find the true best partition that other algorithms have also found. For larger networks, however, our results are better than any previously reported result. For some medium size networks, our value of Q_{best} may be only slightly better than the previous best, but, in these cases, finding any new better result is remarkable and mathematically noteworthy. Furthermore, for these networks, we may be discovering the true best partition. For larger networks, our accuracy improvement is substantial.

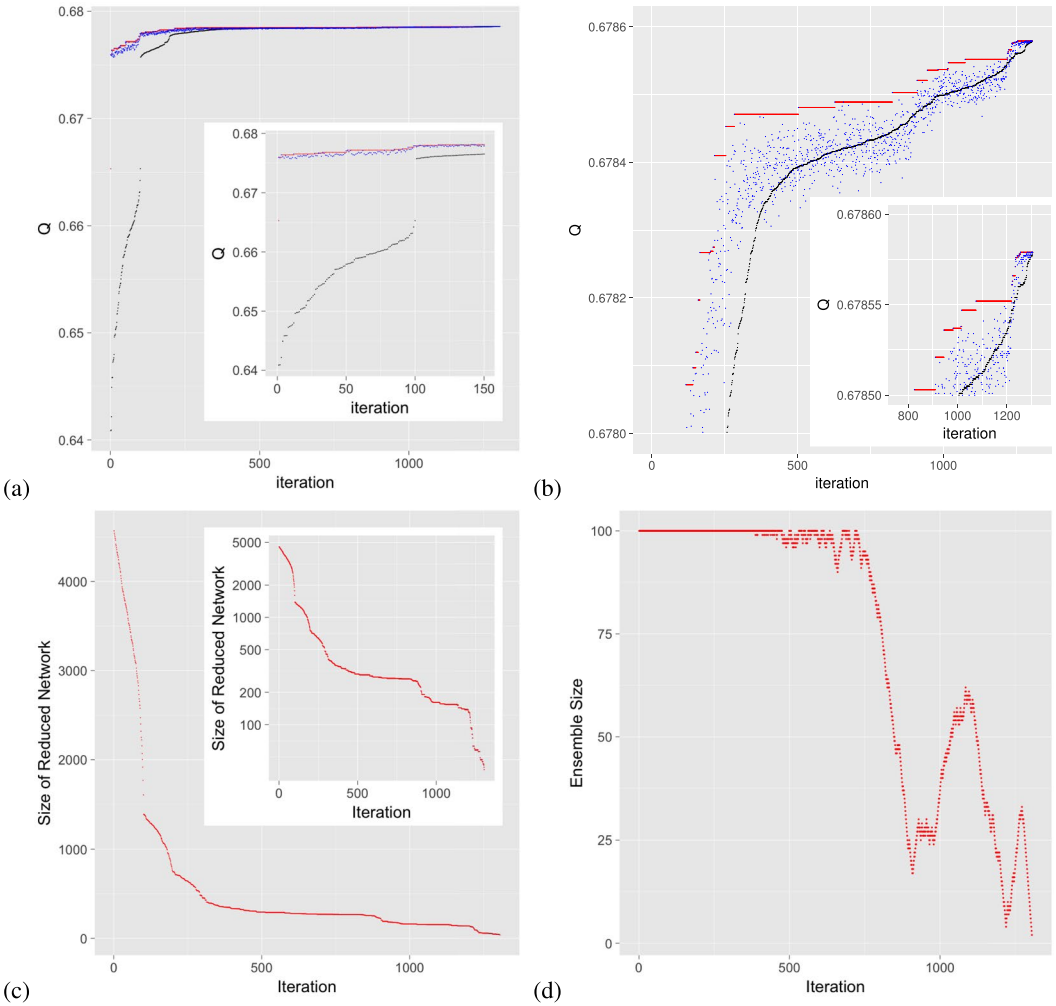

Figure 5. Evolution of the ensemble of partitions $\mathcal{P}$ for a typical run of RenEEL. (**a**) Modularity Q of partitions P_{best}, P'_{best} and P_{worst} at each iteration of the scheme is shown in red, blue, and black, respectively. The inset is an enlargement of the results for the first 150 iterations. (**b**) Same results as in (**a**), but showing only the upper portion of the plot. The inset shows an enlargement of the upper-right corner of the plot. (**c**) Evolution of the size of the reduced network. The inset shows the same plot on a logarithmic scale. (**d**) Evolution of the size of the ensemble $\mathcal{P}$.

Network	Nodes	Links	RenEEL result	Previous best
Adjnoun	112	425	0.313367	0.313367[43]
Jazz	198	2742	0.445144	0.445144[43]
Metabolic	453	2025	0.453248	0.453248[43]
Email	1133	5451	0.582829	0.582829[43]
Polblog	1490	16715	0.427105	0.427105[43]
Netscience	1589	2742	0.959900	0.959900[17]
Power	4941	6594	0.940938	0.940851[43]
PGPgc	10680	24316	0.886853	0.886564[26]
Astro-ph	16706	121251	0.745614	0.744621[43]
Memplus	17758	54196	0.700591	0.700473[26]
As-22july06	22963	48436	0.678579	0.678360[17]
Cond-mat-2005	40421	175693	0.748187	0.746445[17]
Smallworld	100000	499998	0.793175	0.793099[17]
CAIDARouterLevel	192244	609066	0.872086	0.872042[26]

Table 2. Comparison of results using RenEEL to the previous best results for benchmark networks. Maximum modularity Q_{best} obtained by the RenEEL scheme compared to the previous best reported values.

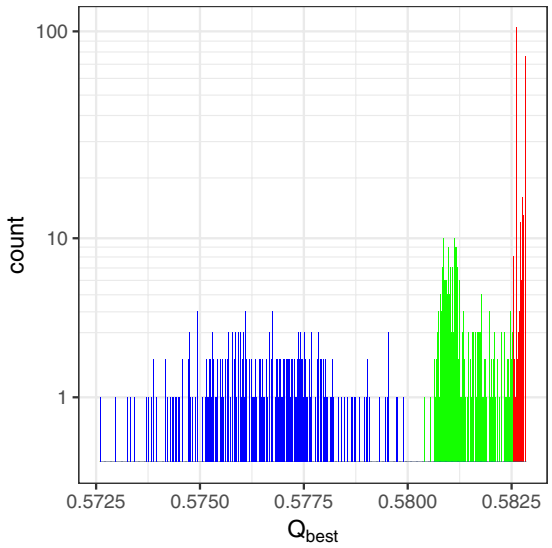

Figure 6. Distribution of Q_{best} obtained by various methods. Frequency plot of Q_{best} for the Email network obtained by multiple realizations of three different methods. Blue corresponds to a naive ensemble analysis scheme, green corresponds to CGGCi scheme, and red corresponds to RenEEL scheme. The y-axis has a logarithmic scale. In this particular example, there is no overlap between the distributions from the different methods.

Perhaps a better way of quantifying the mathematical significance of our results would be, if one knew what the value of true best Modularity Q_{max} is, to consider results for $1/\Delta Q$, where $\Delta Q \equiv Q_{max} - Q$, instead of the results for Q. Unfortunately, that's not possible as the value of Q_{max} for most networks is not known. If we could though, it would be clear that our results are indeed highly significant, for every network studied.

Discussion

Recent advances in Machine Learning and Artificial Intelligence have enabled progress to be made toward solving a range of difficult computational problems[27]. In this paper, we have introduced a powerful algorithmic paradigm for graph partitioning that we call Extremal Ensemble Learning (EEL). EEL is a form of Machine Learning. An EEL scheme creates an ensemble of partitions and then uses information within the ensemble to find new partitions that are used to update the ensemble using extremal criteria. Through the updating procedure, the ensemble learns how to form improved partitions, as it works toward a conclusion by achieving consensus among its member partitions about what the optimal partition is.

The particular EEL scheme we have introduced, Reduced Network Extremal Ensemble Learning (RenEEL), uses information in the ensemble of partitions to create a reduced network that can be efficiently analyzed to find a new partition with which to update the ensemble. We have used RenEEL to find the partition that maximizes the modularity of networks. This is a difficult, NP-hard computational problem[5]. We have shown that an algorithm using the RenEEL scheme outperforms all existing modularity maximizing algorithms when analyzing a variety of commonly studied benchmark networks. For those networks it finds partitions with the largest modularity ever discovered. For the larger benchmark networks, the partitions that we discovered are novel.

Although we have only demonstrated the effectiveness of our algorithm for the well-known problem of finding the network partition that maximizes modularity, the EEL paradigm and the RenEEL scheme can be used to solve other network partitioning problems. For example, the algorithm we used can be straightforwardly adapted to optimize other metrics such as modularity density[28], or excess modularity density[29]. Work is underway to explore the effectiveness of RenEEL for solving those problems. Its potential effectiveness for finding the partition that maximizes excess modularity density may be especially important. Using excess modularity density largely mitigates the resolution limit problem in community detection by maximizing modularity[30], making it a preferred metric for applications where the resolution limit is problematic, such as finding the community structure in gene regulatory networks[31,32].

There is potential to improve upon our results using the RenEEL scheme. As previously discussed, any conventional algorithm can be used as the base algorithm of the scheme. There is also freedom to vary the size of the ensembles used in the scheme. Which base algorithm and what ensemble sizes are best to use depends on the network to be analyzed. Using a high quality base algorithm though, such as the Iterative Spectral Bisectioning, Tuning and Agglomeration algorithm[12], is likely to yield more accurate results for many of the networks studied. There is also potential to improve the RenEEL scheme itself. For instance, currently, a naive ensemble analysis of partitions of the reduced network is used to find a new partition with which to update the ensemble. Another method, such as a recursive use of the RenEEL scheme, may yield better results. Also, currently, once the original ensemble of partitions is created, no new information is ever added to the system during the learning processes. It may be beneficial to occasionally use a new partition of the original network instead of the reduced network to update the ensemble. Work is in progress to explore if these ideas lead to improved results.

Finally, the principal reasons why the RenEEL scheme is both efficient and effective should be noted. Its efficiency stems from its use of an ensemble of partitions to form reduced networks. The smaller size of the reduced networks allows them to be partitioned much more quickly than the full network. Also, because the scheme is so effective, highly accurate results can be obtained even if a fast, but low quality, base algorithm is used. This allows significantly larger networks to be analyzed than what would otherwise be possible. The remarkable effectiveness of RenEEL, even relative to other Ensemble Learning schemes, is mainly due to its extremal updating of the ensemble of partitions. It is of course just one example of a scheme using the EEL paradigm. Its success, though, suggests that EEL is an algorithmic paradigm that will be useful for solving a variety of graph theoretic problems.

Data Availability

The data used in this study are publicly available from the sources that are cited in the main text.

References

1. Fortunato, S. Community detection in graphs. *Physics Reports* **486**, 75–174 (2010).
2. Newman, M. E. & Girvan, M. Finding and evaluating community structure in networks. *Physical Review E* **69**, 026113 (2004).
3. Schaub, M. T., Delvenne, J.-C., Rosvall, M. & Lambiotte, R. The many facets of community detection in complex networks. *Applied Network Science* **2**, 4 (2017).
4. Peel, L., Larremore, D. B. & Clauset, A. The ground truth about metadata and community detection in networks. *Science Advances 3* **5**, e1602548 (2017).
5. Brandes, U. *et al*. On modularity clustering. *IEEE Transactions on Knowledge and Data Engineering* **20**, 172–188 (2008).
6. Clauset, A., Newman, M. E. & Moore, C. Finding community structure in very large networks. *Physical Review E* **70**, 066111 (2004).
7. Newman, M. E. Fast algorithm for detecting community structure in networks. *Physical Review E* **69**, 066133 (2004).
8. Ovelgönne, M. & Geyer-Schulz, A. Cluster cores and modularity maximization. In Data MiningWorkshops (ICDMW), *2010 IEEE International Conference on*, 1204–1213 (IEEE 2010).
9. Blondel, V. D., Guillaume, J.-L., Lambiotte, R. & Lefebvre, E. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment* P10008 (2008).
10. Newman, M. E. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E* **74**, 036104 (2006).
11. Newman, M. E. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences* **103**, 8577–8582 (2006).
12. Treviño, S. III., Nyberg, A., Del Genio, C. I. & Bassler, K. E. Fast and accurate determination of modularity and its effect size. *Journal of Statistical Mechanics: Theory and Experiment* P02003 (2015).
13. Kernighan, B. W. & Lin, S. An efficient heuristic procedure for partitioning graphs. *The Bell System Technical Journal* **49**, 291–307 (1970).
14. Sun, Y., Danila, B., Josić, K. & Bassler, K. E. Improved community structure detection using a modified fine-tuning strategy. *Europhysics Letters* **86**, 28004 (2009).
15. Polikar, R. Ensemble based systems in decision making. *IEEE Circuits and Systems Magazine* **6**, 21–45 (2006).
16. Sagi, O. & Rokach, L. Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* **8**, e1249 (2018).
17. Ovelgönne, M. & Geyer-Schulz, A. An ensemble learning strategy for graph clustering. *Graph Partitioning and Graph Clustering* **588**, 187 (2012).
18. 10th DIMACS Implementation Challenge., https://www.cc.gatech.edu/dimacs10/.
19. Kirkpatrick, S., Gelatt, C. D. & Vecchi, M. P. Optimization by simulated annealing. *Science* **220**, 671–680 (1983).
20. Alex A datasets., http://deim.urv.cat/~alexandre.arenas/data/welcome.htm.
21. Network data., http://www-personal.umich.edu/~mejn/netdata/.
22. Hamm/memplus|SuiteSparse Matrix Collection., https://sparse.tamu.edu/Hamm/memplus.
23. 10th DIMACS Implementation Challenge., https://www.cc.gatech.edu/dimacs10/archive/clustering.shtml.
24. Guimera, R., Danon, L., Diaz-Guilera, A., Giralt, F. & Arenas, A. Self-similar community structure in a network of human interactions. *Physical Review E* **68**, 065103 (2003).
25. Rokach, L. & Maimon, O. *Clustering Methods*, 321–352 (Springer US, Boston, MA, 2005).
26. Index of/dimacs10/results., https://www.cc.gatech.edu/dimacs10/results/.
27. Mohammed, M., Khan, M. B. & Bashier, E. B. M. *Machine Learning: Algorithms and Applications*. (CRC Press, 2016).
28. Chen, M., Kuzmin, K. & Szymanski, B. K. Community detection via maximization of modularity and its variants. *IEEE Transactions on Computational Social Systems* **1**, 46–65 (2014).
29. Chen, T., Singh, P. & Bassler, K. E. Network community detection using modularity density measures. *Journal of Statistical Mechanics: Theory and Experiment*, 053406 (2018).
30. Fortunato, S. & Barthelemy, M. Resolution limit in community detection. *Proceedings of the National Academy of Sciences* **104**, 36–41 (2007).
31. Treviño, S. III., Sun, Y., Cooper, T. F. & Bassler, K. E. Robust detection of hierarchical communities from escherichia coli gene expression data. *PLOS Computational Biology* **8**, 1–15 (2012).
32. Mentzen, W. I. & Wurtele, E. S. Regulon organization of arabidopsis. *BMC Plant Biology* **8**, 99 (2008).
33. Gleiser, P. M. & Danon, L. Community structure in jazz. *Advances in Complex Systems* **6**, 565–573 (2003).
34. Duch, J. & Arenas, A. Community detection in complex networks using extremal optimization. *Physical Review E* **72**, 027104 (2005).
35. Jeong, H., Tombor, B., Albert, R., Oltvai, Z. N. & Barabási, A.-L. The large-scale organization of metabolic networks. *Nature* **407**, 651 (2000).
36. Overbeek, R. *et al*. WIT: integrated system for high-throughput genome sequence analysis and metabolic reconstruction. *Nucleic Acids Research* **28**, 123–125 (2000).
37. Adamic, L. A. & Glance, N. The political blogosphere and the 2004 us election: divided they blog. In *Proceedings of the 3rd International Workshop on Link discovery*, 36–43 (ACM, 2005).
38. Watts, D. J. & Strogatz, S. H. Collective dynamics of 'small-world' networks. *Nature* **393**, 440 (1998).
39. Boguñá, M., Pastor-Satorras, R., Díaz-Guilera, A. & Arenas, A. Models of social networks based on social distance attachment. *Physical Review E* **70**, 056122 (2004).
40. Newman, M. E. The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences* **98**, 404–409 (2001).
41. Davis, T. A. & Hu, Y. The university of florida sparse matrix collection. *ACM Transactions on Mathematical Software (TOMS)* **38**, 1 (2011).
42. CAIDA Skitter Router-Level Topology and Degree Distribution., http://www.caida.org/data/router-adjacencies.
43. Aloise, D. *et al*. Modularity maximization in networks by variable neighborhood search. In *Graph Partitioning and Graph Clustering* (2012).

Acknowledgements
We thank Peter Grassberger and Eve S. Wurtele for fruitful discussions. This work was supported by the NSF through grants DMR-1507371 and IOS-1546858. Some of the computations in this work were done on the uHPC cluster at the University of Houston, acquired through NFS Award Number 1531814.

Author Contributions
J.G., P.S. and K.E.B. conceived of the project. J.G. performed the simulations. J.G., P.S. and K.E.B. analyzed the results and wrote the paper. All authors read and approved the manuscript.

Additional Information
Competing Interests: The authors declare no competing interests.

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