Message passing on networks with loops

George T. Cantwell[a] **and M. E. J. Newman**[a,b]

[a]Department of Physics, University of Michigan, Ann Arbor, Michigan, USA; [b]Center for the Study of Complex Systems, University of Michigan, Ann Arbor, Michigan, USA

Message passing is a fundamental technique for performing calculations on networks and graphs with applications in physics, computer science, statistics, and machine learning, including Bayesian inference, spin models, satisfiability, graph partitioning, network epidemiology, and the calculation of matrix eigenvalues. Despite its wide use, however, it has long been recognized that the method has a fundamental flaw: it works poorly on networks that contain short loops. Loops introduce correlations that can cause the method to give inaccurate answers, or to fail completely in the worst cases. Unfortunately, most real-world networks contain many short loops, which limits the usefulness of the message passing approach. In this paper we demonstrate how to rectify this shortcoming and create message passing methods that work on any network. We give two example applications, one to the percolation properties of networks and the other to the calculation of the spectra of sparse matrices.

message passing | networks | percolation | matrix spectra

Networks occur in a wide range of contexts in physics, biology, computer science, engineering, statistics, the social sciences, and even arts and literature (1). Message passing (2–4), also known as belief propagation or the cavity method, is a fundamental technique for the quantitative calculation of a wide range of network properties, with applications to Bayesian inference (3), NP-hard computational problems (4, 5), statistical physics (4, 6, 7), epidemiology (8), community detection (9), and signal processing (10, 11), among many other things. Message passing can be used both as a numerical method for performing explicit computer calculations and as a tool for analytic reasoning about network properties, leading to new formal results about percolation thresholds (7), algorithm performance (9), spin glasses (12), and other topics. Many of the most powerful new results concerning networks in recent years have been derived from applications of message passing in one form or another.

Despite the central importance of the message passing method, however, it also has a substantial and widely discussed shortcoming: it only works on trees, i.e., networks that are free of loops (4). More generously, one could say that it works to a good approximation on networks that are "locally tree-like," meaning that they may contain long loops but no short ones, so that local neighborhoods within the network take the form of trees. However, most real-world networks that occur in practical applications of the method contain short loops, often in large numbers. When applied to such "loopy" networks the method can give poor results, and in the worst cases can fail to converge to an answer at all.

In this paper, we propose a remedy for this problem. We present a series of methods of increasing elaboration for the solution of problems on networks with loops. The first method in the series is equivalent to the standard message passing algorithm of previous work, which gives poor results in many cases. The last in the series gives exact results on any network with any structure, but is too complicated for practical application in most situations. In between lies a range of methods that give progressively better approximations, and which can be highly accurate in practice, as we will show, yet still simple enough for ready implementation. Indeed even the second member of the series—just one step better than the standard message passing approach—already gives remarkably good results in real-world conditions. We demonstrate our approach with two example applications. The first is to the solution of the bond percolation problem on an arbitrary network, including the calculation of the size of the percolating cluster and the distribution of sizes of small clusters. The second is to the calculation of the spectra of sparse symmetric matrices, where we show that our method is able to calculate the spectra of matrices far larger than those accessible by conventional numerical means.

A number of approaches have been proposed previously for message passing on loopy networks. The most basic, which goes by the name of "loopy belief propagation," is simply to apply the standard message passing equations, ignoring the fact that they are known to be incorrect in general. While this might seem rash, it gives reasonable answers in some cases (11) and there are formal results showing that it can give bounds on the true value of a quantity in others (4, 7). Perturbation theories that treat loopy belief propagation as a zeroth-order approximation have also been considered (13). Broadly, it is found that these methods are suitable for networks that contain a sub-extensive number—and hence a vanishing density—of short loops, but not for networks with a non-vanishing density.

Some progress has been made for the case of networks that are composed of small subgraphs or "motifs" which are allowed to contain loops but which on a larger scale are connected in a loop-free way (14–16). For such networks one can write down exact message passing equations that operate at the higher level of the motifs and which give excellent results for

Significance Statement

Message passing, a celebrated family of methods for performing calculations on networks, has led to many important results in physics, statistics, computer science, and other areas. The technique allows one to divide large network calculations into manageable pieces and hence solve them either analytically or numerically. However, the method has a substantial and widely recognized shortcoming, namely that it works poorly on networks that contain short loops. Unfortunately, most real-world networks contain many such loops, which can cause the method to give inaccurate answers, or even to fail completely in some cases. In this paper we give a solution for this problem, demonstrating how message passing can be extended to any network, regardless of structure, allowing it to become a general tool for the quantitative study of network phenomena.

problems such as structural phase transitions in networks, network spectra, and the solution of spin models (6, 14–17). While effective for theoretical calculations on model networks, however, this approach is of little use in practical situations. To apply it to an arbitrary network one would first need to find a suitable decomposition of the network into motifs, and no general method for doing this is currently known, nor even whether such a decomposition exists.

A third approach is the method known as "generalized belief propagation," which has some elements in common with the motif-based approach but is derived in a different manner, from approximations to the free energy (18, 19). This method, which is focused particularly on the solution of inference problems and related probabilistic calculations on networks, involves a hypergraph-like extension of traditional message passing that aims to calculate the joint distributions of three or more random variables at once, by contrast with the standard approach which focuses on two-variable distributions. Generalized belief propagation was not originally intended as a method for solving problems on loopy networks but can be used in that way in certain cases. It is, however, quite involved in practice, requiring the construction of a nested set of regions and sub-regions within the network, leading to complex sets of equations.

In this paper we take a different approach. In the following sections we directly formulate a message passing framework that works on real-world complex networks containing many short loops by incorporating the loops themselves directly into the message passing equations. In traditional message passing algorithms each node receives a message from each of its neighbors. In our approach they also receive messages from nodes they share loops with. By limiting the loops considered to a fixed maximum length, we develop a series of progressively better approximations for the solution of problems on loopy networks. The equations become more complex as loop length increases but, as we will show, the results given by the method are already impressively accurate even at shorter lengths.

Message passing with loops

Message passing methods calculate some value or state on the nodes of a network by repeatedly passing information between nearby nodes until a self-consistent solution is reached. The approach we propose is characterized by a series of message passing approximations defined as follows. In the zeroth approximation, which is equivalent to the standard message passing method, we assume there are no loops in our network. This implies that the neighbors of a node are not connected to each other, which means they have independent states. It is this independence that makes the standard method work. In the next approximation we no longer assume that neighbors are independent. Instead, we assume that any correlation can be accounted for by direct edges between the neighbors, which is equivalent to allowing the network to contain triangles, the shortest possible kind of loop. In the next approximation after this, we assume that neighbor correlations can accounted for by direct edges plus paths of length 2 between neighbors. Generally, in the rth approximation we assume that correlations between neighbors can be accounted for by paths of length r and shorter.

These successive approximations can be thought of as expressing the properties of nodes in terms of increasingly large neighborhoods and the edges they contain. The zeroth neighborhood $N_i^{(0)}$ of node i contains i's immediate neighbors and the edges connecting them to i, but nothing else. The first neighborhood $N_i^{(1)}$ contains i's immediate neighbors and edges plus all length one paths between neighbors of i. The second neighborhood $N_i^{(2)}$ contains i's neighbors and edges plus all length one and two paths between neighbors of i, and so forth. Figure 1 shows an example of how these neighborhoods are constructed.

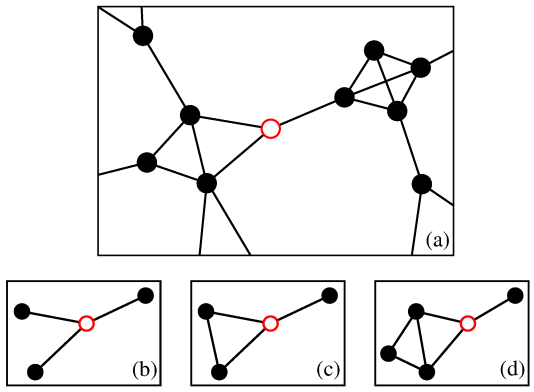

Fig. 1. (a) A node (open circle) and its immediate surroundings in a network. (b) In the zeroth (tree) approximation the neighborhood we consider consists of the neighbors of the focal node only. (c) In the first approximation we also include all length 1 paths between the neighbors. (d) In the second approximation we include all paths of length 1 and 2, and so forth.

Just as the conventional message passing algorithm is exact on trees, our algorithms will be exact on networks with short loops. We define a *primitive cycle* of length r starting at node i to be a loop such that at least one node is not on a shorter loop beginning and ending at i. Then our rth approximation is exact on networks that contain primitive cycles of length $r+2$ or less only. For networks that contain longer primitive cycles it will be an approximation, although as we will see it may be a good one.

Applications

Our approach is best demonstrated by example. In this section we derive message passing equations on loopy networks for two specific applications: the calculation of cluster sizes for bond percolation and the calculation of the spectra of sparse matrices.

Percolation. Consider the bond percolation process on an undirected network of n nodes, where each edge is occupied independently with probability p (20, 21). Occupied edges form connected clusters and we wish to know the distribution of the sizes of these clusters and whether there exists a giant or percolating cluster that occupies a non-vanishing fraction of the network in the limit of large network size.

Let us define the rth neighborhood $N_i^{(r)}$ of node i as previously, then define a random variable Γ_i for our percolation process to be the set of nodes within $N_i^{(r)}$ that are reachable from i by traversing occupied edges only. Our initial goal will be to compute the probability $\pi_i(s)$ that node i belongs to a non-percolating cluster of size s. We will do this in two stages. First we will compute the conditional probability $\pi_i(s|\Gamma_i)$ of belonging to a cluster of size s given the set of reachable nodes. Then we will average over Γ_i to get the full probability $\pi_i(s)$.

Suppose that node i belongs to a cluster of size s. If our network contains no primitive cycles longer than $r+2$, then the set of nodes Γ_i would become disconnected from one another were we to remove all edges in the neighborhood $N_i^{(r)}$—the removal of these edges removes any connections within the neighborhood and there can be no connections via paths outside the neighborhood since such a path would constitute a primitive cycle of length longer than $r+2$. Hence the sizes s_j of the clusters to which the nodes in $N_i^{(r)}$ would belong after this removal must sum to $s-1$ (the sth and last node being provided by i itself). This observation allows us to write

$$\pi_i(s|\Gamma_i) = \sum_{\{s_j : j\in\Gamma_i\}} \Bigl[\prod_{j\in\Gamma_i} \pi_{i\leftarrow j}(s_j)\Bigr] \delta\bigl(s-1, \textstyle\sum_{j\in\Gamma_i} s_j\bigr), \quad [1]$$

where $\pi_{i\leftarrow j}(s)$ is the probability that node j is in a cluster of size s once the edges in $N_i^{(r)}$ are removed.

We can now write a generating function for $\pi_i(s|\Gamma_i)$ as follows

$$\begin{aligned} H_i(z|\Gamma_i) &= \sum_{s=1}^{\infty} \pi_i(s|\Gamma_i)\, z^s \\ &= \sum_{s=1}^{\infty} z^s \Biggl\{ \sum_{\{s_j : j\in\Gamma_i\}} \Bigl[\prod_{j\in\Gamma_i} \pi_{i\leftarrow j}(s_j)\Bigr] \delta(s-1, \textstyle\sum_{j\in\Gamma_i} s_j) \Biggr\} \\ &= z \prod_{j\in\Gamma_i} \sum_{s_j=1}^{\infty} z^{s_j} \pi_{i\leftarrow j}(s_j). \end{aligned} \quad [2]$$

To calculate the full probability $\pi_i(s)$ we average $\pi_i(s|\Gamma_i)$ over sets Γ_i to get $\pi_i(s) = \bigl\langle \pi_i(s|\Gamma_i) \bigr\rangle_{\Gamma_i}$, with the average weighted according to the sum of the probabilities of all edge configurations that correspond to a particular Γ_i. The probability of any individual edge configuration is simply $p^k(1-p)^{m-k}$, where p is the edge occupation probability as previously, m is the number of network edges in the neighborhood $N_i^{(r)}$, and k is the number that are occupied. Performing the same average on Eq. (2) gives us

$$H_i(z) = \sum_{s=1}^{\infty} \pi_i(s)\, z^s = z G_i\bigl(\mathbf{H}_{i\leftarrow}(z)\bigr), \quad [3]$$

where $G_i(\mathbf{y}) = \bigl\langle \prod_{j\in N_i^{(r)}} y_j^{w_{ij}} \bigr\rangle_{\Gamma_i}$ is a generating function for the random variable w_{ij}, which takes the value 1 if $j\in\Gamma_i$ and 0 otherwise, and $\mathbf{H}_{i\leftarrow}(z)$ is the vector with elements $H_{i\leftarrow j}(z)$ for nodes j in $N_i^{(r)}$. (A detailed derivation of Eq. (3) is given in the Supplementary Information.)

To complete the calculation we need to evaluate $H_{i\leftarrow j}(z)$, whose computation follows the same logic as for $H_i(z)$, the only difference being that in considering the neighborhood of node j we must remove the entire neighborhood of i first, as described above. Doing this leads to

$$H_{i\leftarrow j}(z) = z G_{i\leftarrow j}\bigl(\mathbf{H}_{j\leftarrow}(z)\bigr), \quad [4]$$

where $G_{i\leftarrow j}(\mathbf{y})$ is the equivalent of $G_i(\mathbf{y})$ when $N_i^{(r)}$ is removed. If we can solve this equation self-consistently for $\mathbf{H}_{j\leftarrow}(z)$, we can substitute the solution into Eq. (3) to compute the full cluster size generating function. The message passing method involves solving Eq. (4) by simple iteration: we choose suitable starting values, for instance at random, and iterate the equations to convergence.

From the cluster size generating function we can calculate a range of quantities of interest. For example, the probability that node i belongs to a small cluster (of any size) is $H_i(1) = \sum_s \pi_i(s)$. If it does not belong to a small cluster then necessarily it belongs to the percolating cluster and hence the expected fraction S of the network taken up by the percolating cluster is

$$S = 1 - \frac{1}{n} \sum_i H_i(1). \quad [5]$$

Similarly, the average value of s_i is

$$\begin{aligned} \langle s_i \rangle &= \sum_{s=1}^{\infty} s\pi_i(s) = H_i'(1) \\ &= H_i(1) + \sum_{j\in N_i^{(r)}} H_{i\leftarrow j}'(1)\, \partial_j G_i(\mathbf{H}_{i\leftarrow}), \end{aligned} \quad [6]$$

where H' is the derivative of H and $\partial_j G_i$ is the partial derivative of G_i with respect to its jth argument. $H_{i\leftarrow j}'(1)$ can be found by differentiating Eq. (4) and setting $z=1$ to give the self-consistent equation

$$H_{i\leftarrow j}'(1) = H_{i\leftarrow j}(1) + \sum_{k\in N_{j\setminus i}^{(r)}} H_{j\leftarrow k}'(1)\, \partial_k G_{i\leftarrow j}\left(\mathbf{H}_{j\leftarrow}\right), \quad [7]$$

where $N_{j\setminus i}^{(r)}$ denotes the neighborhood $N_j^{(r)}$ with $N_i^{(r)}$ removed.

While these equations are straightforward in principle, implementing them in practice presents some additional challenges. Computing the generating functions $G_i(\mathbf{y})$ and $G_{i\leftarrow j}(\mathbf{y})$ can be demanding, since it requires us to perform an average over the occupancy configurations of all edges within the neighborhoods $N_i^{(r)}$ and $N_{j\setminus i}^{(r)}$, and the number of configurations increases exponentially with neighborhood size. For small neighborhoods, such as those found on low-dimensional lattices, it is feasible to average exhaustively, but for many complex networks this is not possible. In such cases we instead approximate the average by Monte Carlo sampling of configurations—see the Supplementary Information for details. A nice feature of the Monte Carlo procedure is that the samples need be taken only once for the entire calculation and can then be reused on successive iterations of the message passing process.

In practice the method gives excellent results. We show example applications to two real-world networks in Fig. 2, the first a social network of coauthorship relations between scientists in the field of condensed matter physics (22) and the second a network of trust relations between users of the PGP encryption software (23). Both networks have a high density of short loops. For each network the figure shows, as a function of p, several different estimates of both the average size $\langle s\rangle$ of a small cluster and the size S of the percolating cluster as a fraction of n. First we show an estimate made using standard message passing (dashed line)—the $r=0$ approximation in our nomenclature—which ignores loops and is expected to give poor results. Second, we show the next two approximations in our series, those for $r=1$ and $r=2$ (dotted and solid lines respectively), with $G_i(\mathbf{y})$ and $G_{i\leftarrow j}(\mathbf{y})$ estimated by Monte Carlo sampling as described above. We use only eight samples for each node i but the results are nonetheless impressively

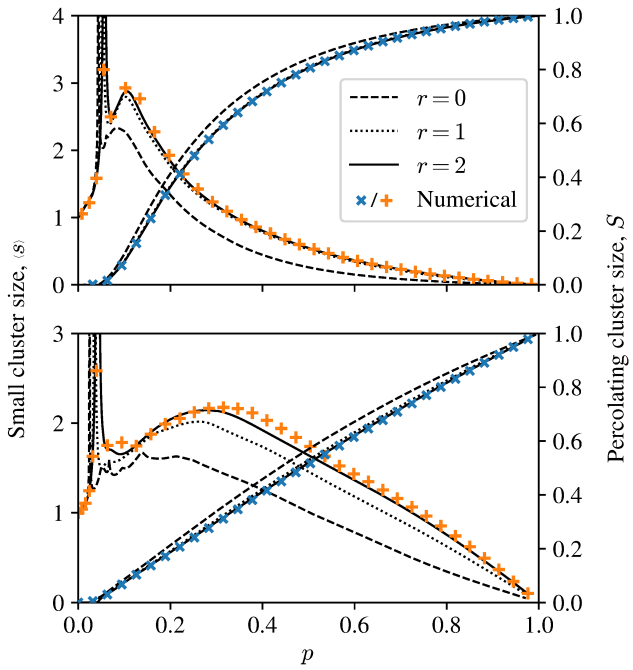

Fig. 2. Percolating cluster size (× symbols) and average cluster size (+ symbols) for two real-world networks. Top: the largest component of a coauthorship network of 13,861 scientists (22). Bottom: a network of 10,680 users of the PGP encryption software (23).

accurate. Third, we show for comparison a direct numerical estimate of the quantities in question made by conventional simulation of the percolation process.

For both networks we see the same pattern. The traditional message passing method fares poorly, as expected, giving estimates that are substantially in disagreement with the simulation results, particularly for the calculations of average cluster size. The $r = 1$ approximation, on the other hand, does significantly better and the $r = 2$ approximation does better still, agreeing closely with the numerical results for all measures on both networks. In these examples at least, it appears that the $r = 2$ method gives accurate results for bond percolation, where standard message passing fails.

The message passing algorithm is relatively fast. For $r \leq 1$ each node receives a message from each neighbor on each iteration, and so on a network with mean degree c there are cn messages passed per iteration. For $r \geq 2$ the number of messages depends on the network structure. On trees the number of messages remains unchanged at cn as r increases but on networks with loops it grows and for large numbers of loops it can grow exponentially. In the common sparse case where the size of the neighborhoods does not grow with n, however, the number of messages is linear in n for fixed r and hence so is the running time for each iteration. It is not known in general how many iterations are needed for message passing methods to reach convergence, but elementary heuristic arguments suggest the number should be on the order of the diameter of the network, which is typically $\mathrm{O}(\log n)$. Thus we expect overall running time to be $\mathrm{O}(n \log n)$ for sparse networks at fixed r.

This makes the algorithm quite efficient, although direct numerical simulations of percolation run comparably fast, so the message passing approach does not offer a speed advantage over traditional approaches. However, the two approaches are calculating different things. Traditional simulations of percolation perform a calculation for one particular realization of bond occupancies. If we want average values over many realizations we must perform the average explicitly, repeating the whole simulation for each realization. The message passing approach, on the other hand, computes the average over realizations in a single calculation and no repetition is necessary, making it potentially the faster method in some situations.

In the next section we demonstrate another example application of our method, to the calculation of the spectrum of a sparse matrix, where traditional and message passing calculations differ substantially in their running time, the message passing approach being much faster, making calculations possible for large systems whose spectra cannot be computed in any reasonable amount of time by traditional means.

Matrix spectra. For our second example application we show how the message passing method can be used to compute the eigenvalue spectrum of a sparse symmetric matrix. Any $n \times n$ symmetric matrix can be thought of as an undirected weighted network on n nodes and we can use this equivalence to apply the message passing method to such matrices.

The spectral density of a symmetric matrix $\mathbf{A}$ is the quantity

$$\rho(x) = \frac{1}{n} \sum_{k=1}^{n} \delta(x - \lambda_k), \qquad [8]$$

where λ_k is the kth eigenvalue of $\mathbf{A}$, and $\delta(x)$ is the Dirac delta function. Following standard arguments (24), we can show that the spectral density is equal to the imaginary part of the complex function

$$\begin{aligned} \rho(z) &= -\frac{1}{n\pi} \sum_{k=1}^{n} \frac{1}{z - \lambda_k} = -\frac{1}{n\pi} \operatorname{Tr}(z\mathbf{I} - \mathbf{A})^{-1} \\ &= -\frac{1}{n\pi z} \sum_{i=1}^{n} \sum_{s=0}^{\infty} \frac{X_i^s}{z^s}, \end{aligned} \qquad [9]$$

where $X_i^s = [\mathbf{A}^s]_{ii}$ is the ith diagonal element of $\mathbf{A}^s$, and $z = x + i\eta$ and we take the limit as $\eta \to 0$ from above. The imaginary part η acts as a resolution parameter that broadens the delta-function peaks in Eq. (8) by an amount roughly equal to its value.

The quantities $X_i^s = [\mathbf{A}^s]_{ii}$ can be related to sums over closed walks in the equivalent network. If we consider the "weight" of a walk to be the product of the matrix elements on the edges it traverses, then X_i^s is the sum of the weights of all closed walks of length s that start and end at node i.

A closed walk from i need not visit i only at its start and end, however. It can return to i any number of times over the course of the walk. The simplest case, where it returns just once at the end of the walk, we will call an *excursion*. A more general closed walk that returns to node i exactly m times can be thought of as a succession of m excursions. Such a walk will have length s if those m excursions have lengths $s_1 \ldots s_m$ with $\sum_{u=1}^{m} s_u = s$.

With this in mind, let Y_i^s be the sum of the weights of all *excursions* of length s that start and end at node i. Then the sum X_i^s over *closed walks* of length s can be written in terms

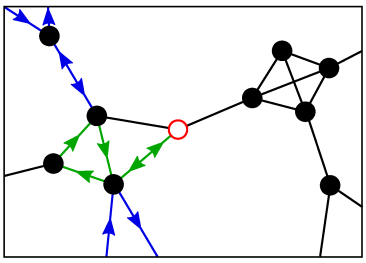

Fig. 3. An example excursion from the central node (open circle). The excursion is equivalent to an excursion inside the neighborhood, shown with green arrows, plus closed walks to regions outside of the neighborhood, shown in blue.

of Y_i^s as

$$X_i^s = \sum_{m=0}^{\infty} \Biggl[\sum_{s_1=1}^{\infty} \cdots \sum_{s_m=1}^{\infty} \delta\bigl(s, \textstyle\sum_{u=1}^m s_u\bigr) \displaystyle\prod_{u=1}^m Y_i^{s_u} \Biggr]. \qquad [10]$$

Using this result, and defining the function

$$H_i(z) = \sum_{s=1}^{\infty} \frac{Y_i^s}{z^{s-1}}, \qquad [11]$$

we find after some algebra that

$$\rho(z) = -\frac{1}{n\pi} \sum_{i=1}^{n} \frac{1}{z - H_i(z)}. \qquad [12]$$

(See the Supplementary Information for a detailed derivation.) Thus, if we can calculate $H_i(z)$ then we can calculate $\rho(z)$. This we do as follows.

Consider the neighborhood $N_i^{(r)}$ around i. If there are no primitive cycles of length longer than $r+2$ in our network then all loops starting at i are already included within the neighborhood, which means that any excursion from i takes the form of an excursion w within the neighborhood plus some number of additional closed walks outside the neighborhood that each start at one of the nodes in w and return some time later to the same node—see Fig. 3. The additional walks must necessarily return to the same node they started at since if they did not they would complete a loop outside the neighborhood, of which by hypothesis there are none.

Let the length of the excursion w be $l+1$, meaning that it visits l nodes $j_1 \ldots j_l$ (not necessarily distinct) within the neighborhood other than the starting node i, and let s_j be the length of the external closed walk (if any) that starts at node j, or zero if there is no such walk. The total length of the complete excursion from i will then be $l + 1 + \sum_{j\in w} s_j$ and the sum of the weights of all excursions of length s with w as their foundation will be

$$|w| \sum_{\{s_j : j\in w\}} \delta\bigl(s, l+1+\textstyle\sum_{j\in w} s_j\bigr) \displaystyle\prod_{j\in w} X_{i\leftarrow j}^{s_j}, \qquad [13]$$

where $|w|$ is the weight of w itself and $X_{i\leftarrow j}^s$ is the sum of weights of length-s walks from node j if the neighborhood $N_i^{(r)}$ is removed from the network. By a similar argument to the one that led to Eq. (10), we can express $X_{i\leftarrow j}^s$ in terms of the sum $Y_{i\leftarrow j}^s$ of excursions from j thus:

$$X_{i\leftarrow j}^s = \sum_{m=0}^{\infty} \Biggl[\sum_{s_1=1}^{\infty} \cdots \sum_{s_m=1}^{\infty} \delta\bigl(s, \textstyle\sum_{u=1}^m s_u\bigr) \displaystyle\prod_{u=1}^m Y_{i\leftarrow j}^{s_u} \Biggr]. \qquad [14]$$

And the quantity Y_i^s appearing in Eq. (11) can be calculated by summing Eq. (13) first over the set of excursions of length $l+1$ in the neighborhood of i and then over l. This allows us to write Eq. (11) as

$$H_i(z) = \sum_{w\in W_i} |w| \prod_{j\in w} \frac{1}{z - H_{i\leftarrow j}(z)}, \qquad [15]$$

where W_i is the complete set of excursions of all lengths in the neighborhood of i and we have defined

$$H_{i\leftarrow j}(z) = \sum_{s=1}^{\infty} \frac{Y_{i\leftarrow j}^s}{z^{s-1}}. \qquad [16]$$

Following an analogous line of argument for this function we can show similarly that

$$H_{i\leftarrow j}(z) = \sum_{w\in W_{j\setminus i}} |w| \prod_{k\in w} \frac{1}{z - H_{j\leftarrow k}(z)}. \qquad [17]$$

Equation (17) defines our message passing equations for the spectral density. By iterating these equations to convergence from suitable starting values we can solve for the values of the messages $H_{i\leftarrow j}(z)$, then substitute into Eqs. (12) and (15) and to get the spectral density itself.

As with our percolation example, the utility of this approach relies on our having an efficient method for evaluating the sum in Eq. (17). Fortunately there is such a method, as follows. Let $\mathbf{v}_{i\leftarrow j}$ be the vector with elements $v_{i\leftarrow j,k} = A_{jk}$ if nodes j and k are directly connected in $N_{j\setminus i}^{(r)}$ and 0 otherwise. Further, let $\mathbf{A}^{i\leftarrow j}$ be the matrix of the neighborhood of j with the neighborhood of i removed, such that

$$A_{kl}^{i\leftarrow j} = \begin{cases} A_{kl} & \text{for } k,l \neq j \text{ and edge } (k,l) \in N_{j\setminus i}^{(r)}, \\ 0 & \text{otherwise,} \end{cases} \qquad [18]$$

and let $\mathbf{D}^{i\leftarrow j}(z)$ be the diagonal matrix with entries $D_{kk}^{i\leftarrow j} = z - H_{j\leftarrow k}(z)$. As shown in the Supplementary Information, Eq. (17) can then be written

$$H_{i\leftarrow j}(z) = A_{jj} + \mathbf{v}_{i\leftarrow j}^T \bigl(\mathbf{D}^{i\leftarrow j} - \mathbf{A}^{i\leftarrow j}\bigr)^{-1} \mathbf{v}_{i\leftarrow j}. \qquad [19]$$

Since the matrices in this equation are the size of the neighborhood, each message update requires us to invert only a small matrix, which gives us a linear-time algorithm for each iteration of the message passing equations and an overall running time of $O(n \log n)$ for sparse networks with fixed neighborhood sizes, or for the equivalent sparse matrices.

As an example of this method, we show in Fig. 4 spectra for the same two real-world networks that we used in Fig. 2. To demonstrate the flexibility of the method we calculate different spectra in the two cases: for the coauthorship network we calculate the spectrum of the graph Laplacian; for the PGP network we calculate the spectrum of the adjacency matrix. For each network the black curve in the figure shows the spectral density calculated using the message passing method with $r = 1$. We also calculate the full set of eigenvalues of each network directly using traditional numerical methods and substitute the results into Eq. (9) to compute the spectral density, shown as the shaded areas in the figure. As we can see, the agreement between the two methods is excellent for both networks. There are a few regions where small differences are visible but in general they agree closely. Extending the

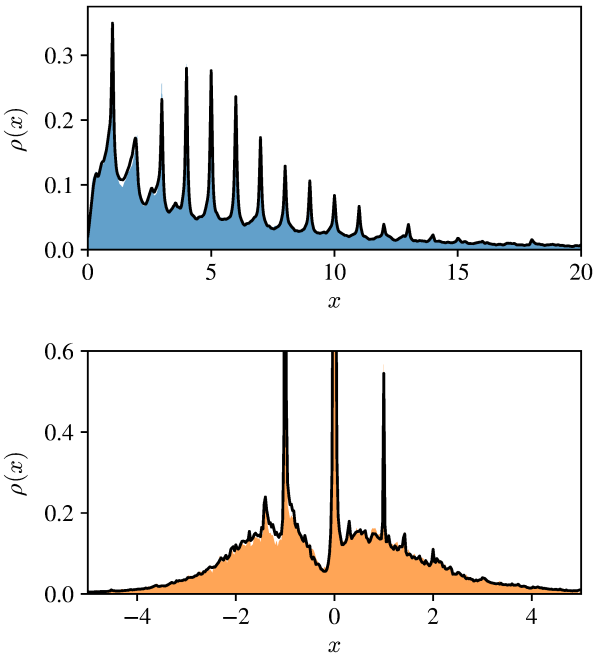

Fig. 4. Matrix spectra for the same two networks that were used in Fig. 2. Top: the spectrum of the graph Laplacian of the coauthorship network. Bottom: the spectrum of the adjacency matrix of the PGP network. The shaded areas show the spectral density calculated by direct numerical diagonalization. The black lines show the $r = 1$ message-passing approximation. The broadening parameter η was set to 0.05 in the top panel and 0.01 in the bottom panel.

calculation to the next ($r = 2$) approximation gives a modest further improvement in the results.

The $O(n \log n)$ running time of the message passing algorithm significantly outstrips that of traditional numerical diagonalization. Complete spectra are normally calculated using the QR algorithm, which runs in time $O(n^3)$ and is consequently much slower as system size becomes large. The Lanczos algorithm is faster, but typically gives only a few leading eigenvalues and not a complete spectrum—it takes time $O(rn)$ to compute r eigenvalues of a sparse matrix. The kernel polynomial method (25) is capable of computing complete spectra for sparse matrices, but requires Monte Carlo evaluation of the traces of large matrix powers which has slow convergence and is always only approximate, even in cases where our method gives exact results.

This opens up the possibility of using our approach to calculate the spectral density of networks and matrices significantly larger than those that can be tackled by traditional means. As an example, we have used the message passing method to compute the spectral density of one network with 317 080 nodes. This is significantly larger than the largest systems that can be diagonalized using the QR algorithm, which on current (non-parallel) commodity hardware is limited to a few tens of thousands of nodes in practical running times.

Conclusions

In this paper we have described a new class of message passing methods for performing calculations on networks that contain short loops, a situation in which traditional message passing often gives poor results or may fail to converge entirely. We derive message passing equations that account for the effects of loops up to a fixed length that we choose, so that calculations are exact on networks with no loops longer than this. In practice we achieve excellent results on real-world networks by accounting for loops up to length three or four only, even if longer loops are present.

We have demonstrated our approach with two example applications, one to the calculation of bond percolation properties of networks and the other to the calculation of the spectra of sparse matrices. In the first case we develop message passing equations for the size of the percolating cluster and the average size of small clusters and find that these give good results, even on networks with an extremely high density of short loops. For the calculation of matrix spectra, we develop a message passing algorithm for the spectral density that gives results in good agreement with traditional numerical diagonalization but in much shorter running times. Where traditional methods are limited to matrices with at most a few tens of thousands of rows and columns, our method can be applied to cases with hundreds of thousands at least.

There are a number of possible directions for future work on this topic. Chief among them is the application of the method to other classes of problems, such as epidemiological calculations, graph coloring, or spin models. Many extensions of the calculations in this paper are also possible, including the inclusion of longer primitive cycles in the message passing equations, development of more efficient algorithms for very large systems, and applications to individual examples of interest such as the computation of spectra for very large graphs. Finally, while our example applications are to real-world networks, the same methods could in principle be applied to model networks, and in particular to ensembles of random graphs, which opens up the possibility of new analytic results about such models. These possibilities, however, we leave for future research.

ACKNOWLEDGMENTS. The authors thank Alec Kirkley, Cristopher Moore, Jean-Gabriel Young, and Robert Ziff for useful conversations. This work was funded in part by the US National Science Foundation under grant DMS–1710848.

1. M. Newman, *Networks*. Oxford University Press, Oxford, 2nd edition (2018).
2. H. A. Bethe, Statistical theory of superlattices. *Proc. R. Soc. London A* **150**, 552–575 (1935).
3. J. Pearl, Reverend Bayes on inference engines: A distributed hierarchical approach. In *Proceedings of the 2nd National Conference on Artificial Intelligence*, pp. 133–136, AAAI Press, Palo Alto, CA (1982).
4. M. Mézard and A. Montanari, *Information, Physics, and Computation*. Oxford University Press, Oxford (2009).
5. M. Mézard, G. Parisi and R. Zecchina, Analytic and algorithmic solution of random satisfiability problems. *Science* **297**, 812–815 (2002).
6. S. Yoon, A. V. Goltsev, S. N. Dorogovtsev, and J. F. F. Mendes, Belief-propagation algorithm and the Ising model on networks with arbitrary distributions of motifs. *Phys. Rev. E* **84**, 041144 (2011).
7. B. Karrer, M. E. J. Newman, and L. Zdeborová, Percolation on sparse networks. *Phys. Rev. Lett.* **113**, 208702 (2014).
8. B. Karrer and M. E. J. Newman, A message passing approach for general epidemic models. *Phys. Rev. E* **82**, 016101 (2010).
9. A. Decelle, F. Krzakala, C. Moore, and L. Zdeborová, Inference and phase transitions in the detection of modules in sparse networks. *Phys. Rev. Lett.* **107**, 065701 (2011).
10. R. G. Gallager, *Low-Density Parity-Check Codes*. MIT Press, Cambridge, MA (1963).
11. B. J. Frey and D. J. C. MacKay, A revolution: Belief propagation in graphs with cycles. In M. I. Jordan, M. J. Kearns, and S. A. Solla (eds.), *Proceedings of the 1997 Conference on Neural Information Processing Systems*, pp. 479–485, MIT Press, Cambridge, MA (1998).
12. M. Mézard, G. Parisi, and M. A. Viasoro, *Spin Glass Theory and Beyond*. World Scientific, Singaport (1987).
13. M. Chertkov and V. Y. Chernyak, Loop calculus in statistical physics and information science. *Phys. Rev. E* **73**, 065102 (2006).
14. M. E. J. Newman, Random graphs with clustering. *Phys. Rev. Lett.* **103**, 058701 (2009).
15. J. C. Miller, Percolation and epidemics in random clustered networks. *Phys. Rev. E* **80**, 020901 (2009).
16. B. Karrer and M. E. J. Newman, Random graphs containing arbitrary distributions of subgraphs. *Phys. Rev. E* **82**, 066118 (2010).

17. M. E. J. Newman, Spectra of networks containing short loops. *Phys. Rev. E* **100**, 012314 (2019).
18. J. S. Yedidia, W. T. Freeman, and Y. Weiss, Generalized belief propagation. In T. G. Dietterich, S. Becker, and Z. Ghahramani (eds.), *Proceedings of the 14th Annual Conference on Neural Information Processing Systems*, pp. 689–695, MIT Press, Cambridge, MA (2001).
19. J. S. Yedidia, W. T. Freeman, and Y. Weiss, Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory* **51**, 2282–2312 (2005).
20. H. L. Frisch and J. M. Hammersley, Percolation processes and related topics. *J. SIAM* **11**, 894–918 (1963).
21. D. Stauffer and A. Aharony, *Introduction to Percolation Theory*. Taylor and Francis, London, 2nd edition (1992).
22. M. E. J. Newman, The structure of scientific collaboration networks. *Proc. Natl. Acad. Sci. USA* **98**, 404–409 (2001).
23. M. Boguñá, R. Pastor-Satorras, A. Díaz-Guilera, and A. Arenas, Models of social networks based on social distance attachment. *Phys. Rev. E* **70**, 056122 (2004).
24. R. R. Nadakuditi and M. E. J. Newman, Spectra of random graphs with arbitrary expected degrees. *Phys. Rev. E* **87**, 012803 (2013).
25. A. Weiße, G. Wellein, A. Alvermann, and H. Fehske, The kernel polynomial method. *Rev. Mod. Phys.* **78**, 275–306 (2006).
26. M. E. J. Newman and R. M. Ziff, Efficient Monte Carlo algorithm and high-precision results for percolation. *Phys. Rev. Lett.* **85**, 4104–4107 (2000).