

Generation of arbitrary two-point correlated directed networks with given modularity

Jie Zhou^a, Gaoxi Xiao^{a,*}, Limsoon Wong^b, Xiuju Fu^c, Stefan Ma^d, Tee Hiang Cheng^a

^a School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798, Singapore

^b School of Computing & School of Medicine, National University of Singapore, Singapore 117417, Singapore

^c Advanced Computing, Institute of High Performance Computing, Singapore 138623, Singapore

^d Epidemiology & Disease Control Division, Ministry of Health, Singapore 169854, Singapore

ARTICLE INFO

Article history:

Received 24 March 2010

Received in revised form 27 May 2010

Accepted 29 May 2010

Available online 3 June 2010

Communicated by C.R. Doering

Keywords:

Two-point correlation

Modularity

Directed network

ABSTRACT

In this Letter, we introduce measures of correlation in directed networks and develop an efficient algorithm for generating directed networks with arbitrary two-point correlation. Furthermore, a method is proposed for adjusting community structure in directed networks without changing the correlation. Effectiveness of both methods is verified by numerical results.

© 2010 Elsevier B.V. All rights reserved.

1. Introduction

Extracting nontrivial common features from a wide range of different networks and generating network models properly reflecting such features are essential to research on complex networks. The generated models enable in-depth studies on the effects of the extracted common features and their correlations with other properties of networks. Without good understanding of such critical issues, research on different systems such as neural, metabolic, and ecological networks [1–5], World Wide Web, power grids, and transport networks [6–8] may not have spawned the new research area of complex networks [9].

Earlier studies on network generation were mainly for generating network models with requested nodal-degree distributions [10–14]. Extended work includes fulfilling other parameters such as clustering coefficient [15]. It was found that two-point correlation, also known as degree–degree correlation, may profoundly influence the structure of complex networks and the dynamics taking place on them [16–19]. In reality almost all the empirical networks have a nontrivial two-point correlation structure. By adopting a simple rewiring method, R. Xulvi-Brunet et al. proposed an algorithm for adjusting the assortative coefficient, a global parameter describing the two-point correction, to any value without changing the degree distribution [20]. In 2007, S. Weber et al.

proposed an efficient algorithm for generating random networks with an a priori defined two-point-correlated structure on undirected network [21]. In the algorithm, links are inserted between randomly selected nodes subject to the given nodal-degree distribution and a conditional joint degree–degree distribution which captures the two-point correction. Further progress was made by A. Pusch et al., who proposed an algorithm for generating an undirected network with required degree–degree correlation and an adjustable level of clustering defined by the degree-dependent clustering coefficient [22].

Many real-life networks, such as the World Wide Web, neural, metabolic and ecological networks, are directed networks [3,5,23–25]. In such networks, influences between individuals go in one direction but not the opposite. As a result, both static and dynamic properties of directed networks may become significantly different from those of the undirected ones. Since the correlation is an important property of complex networks, it is necessary to develop algorithms for generating directed networks with the specified correlation. Inspired by existing results for undirected networks such as those reported in [21,22], in this Letter we introduce measures of correlation in directed networks and propose an algorithm for constructing directed networks with specified two-point correlation. Necessary theoretical analysis is also developed.

Another important structural feature that we would reflect in the generation of directed networks is network modularity, which measures the existence and distribution of densely connected groups of vertices with sparse connections between them [26–28]. Modularity is considered as one of the main organizing principles

* Corresponding author.

E-mail address: egxxiao@ntu.edu.sg (G. Xiao).

of many real-life networks, e.g., biological networks [29,30] and social networks [5]. In 2008, A. Kreimer et al. calculated the modularity scores (quantified by using Newman's algorithm [28]) of more than 300 bacterial metabolic networks and found that they are generally of quite high values [31]. Recently several methods were proposed for generating networks with predefined community structures, such that the performance of various community detection algorithms can be tested [32,33]. However the issue of constructing a network with a given modularity score yet without a predefined community structure has not received the attention it deserves. In this Letter, based on the generated directed network, we propose a method to further adjust the network modularity score without changing its correlation level. We use artificial and real network data to validate our algorithms. Specifically, correlation values and modularity scores are derived from artificial and real-life networks and then be used to generate networks. It is found that the generated networks coincide with the original ones.

The above two parts of contributions combined together enable an efficient and reliable algorithm for generating complex networks with any assortative coefficients and modularity scores. Such an algorithm may find wide applications in different areas including social sciences, communication engineering and ecology, etc. For example, recently it was found that certain network structures and assortativity/clustering properties may strongly encourage cooperations between individuals [34–37]. Such observations may help explain the wide existences of emerging cooperations under the dilemma situations. A flexible network generator shall be of help to research on such interesting topics. Other examples include studies on the effects of assortativity and modularity on distributed search [38] and stability of ecological systems [39], etc.

The layout of this Letter is as follows. Measures of correlation in directed networks are introduced in Section 2. The relationship between them is also briefly discussed. We propose the algorithm for generating a directed network with any given two-point correlation in Section 3. In Section 4, the algorithm for tuning modularity without changing the two-point correlation is presented. Section 5 concludes the Letter.

2. Correlation in directed networks

In this Letter, we call a link as an *arc* in a directed network, and as an *edge* in an undirected network. Extended from that for undirected networks [21], two-point correlation of a directed network can be statistically described by its *joint degree distribution* $p(j^{in}, j^{out}; k^{in}, k^{out})$, which denotes the probability that a randomly chosen arc going from a node with an in-degree j^{in} and an out-degree j^{out} (hereafter termed as with a degree (j^{in}, j^{out})) to another node with a degree (k^{in}, k^{out}) . Later we shall show that *nodal-degree distribution*, denoting the probability that a randomly selected node has a certain degree of (j^{in}, j^{out}) , can be derived from the joint degree distribution.

The joint degree distribution $p(j^{in}, j^{out}; k^{in}, k^{out})$ is generally asymmetric, i.e.,

$$p(j^{in}, j^{out}; k^{in}, k^{out}) \neq p(k^{in}, k^{out}; j^{in}, j^{out}).$$

Obviously, $p(j^{in}, j^{out}; k^{in}, k^{out}) = 0$ if $j^{out} = 0$ or $k^{in} = 0$. However, when $j^{in} = 0$ or $k^{out} = 0$, the joint degree distribution may be not zero. By summing over all the k^{in} and k^{out} , one obtains that

$$p_l(j^{in}, j^{out}) = \sum_{k^{in}, k^{out}} p(j^{in}, j^{out}; k^{in}, k^{out}), \quad (1)$$

which denotes the probability that a randomly chosen arc is emanated from a node with a degree (j^{in}, j^{out}) . Similarly, we have

$$p_r(k^{in}, k^{out}) = \sum_{j^{in}, j^{out}} p(j^{in}, j^{out}; k^{in}, k^{out}), \quad (2)$$

which denotes the probability that a randomly chosen arc ends at a node with a degree (k^{in}, k^{out}) . The average nodal-degree of the network $\langle k \rangle$, defined as the ratio between the number of arcs M and the number of nodes N , and the nodal-degree distribution $p_n(j^{in}, j^{out})$ can be calculated by using p_l and p_r . Specifically, when $j^{out} \neq 0$, we have

$$p_n(j^{in}, j^{out}) = \langle k \rangle \frac{p_l(j^{in}, j^{out})}{j^{out}}. \quad (3)$$

Similarly, when $k^{in} \neq 0$, we have

$$p_n(k^{in}, k^{out}) = \langle k \rangle \frac{p_r(k^{in}, k^{out})}{k^{in}}. \quad (4)$$

As Eq. (4) contains the case of $j^{out} = 0$ and Eq. (3) contains the case of $k^{in} = 0$, they combined together cover all the different cases for calculating $p_n(j^{in}, j^{out})$. Specifically, from the conservation condition

$$\sum_{j^{in}, j^{out}} p_n(j^{in}, j^{out}) = 1$$

and Eqs. (3) and (4), we have

$$p_n(j^{in}, j^{out}) = \begin{cases} \langle k \rangle p_l(j^{in}, j^{out}) / j^{out} & \text{if } j^{out} \geq 1, \\ \langle k \rangle p_r(j^{in}, 0) / j^{in} & \text{if } j^{out} = 0 \end{cases} \quad (5)$$

and

$$\langle k \rangle = \frac{1}{\left(\sum_{j^{in} \geq 0, j^{out} \geq 1} \frac{p_l(j^{in}, j^{out})}{j^{out}} + \sum_{j^{in} \geq 1} \frac{p_r(j^{in}, 0)}{j^{in}} \right)}. \quad (6)$$

With the nodal-degree distribution, we can further calculate *out-degree distribution* $p^{out}(j^{out})$ and *in-degree distribution* $p^{in}(j^{in})$, defined as the probabilities that a randomly chosen node has an out-degree j^{out} or an in-degree j^{in} respectively:

$$p^{in}(j^{in}) = \sum_{j^{out}} p_n(j^{in}, j^{out}),$$

$$p^{out}(j^{out}) = \sum_{j^{in}} p_n(j^{in}, j^{out}). \quad (7)$$

The above degree distributions enable the calculations of *single-point correlation*, defined as the correlation of the in-degrees and out-degrees of network node quantified by the Pearson correlation:

$$r_n = \frac{1}{\sigma_n^2} \sum_{j^{in}, j^{out}} j^{in} j^{out} [p_n(j^{in}, j^{out}) - p^{out}(j^{out}) p^{in}(j^{in})]. \quad (8)$$

The factor $\frac{1}{\sigma_n^2}$ is to ensure that r_n fall into the range of $[-1, 1]$; $\sigma_n^2 = \sigma^{out} \cdot \sigma^{in}$, where

$$\begin{aligned} (\sigma^{out})^2 &= \sum_{j^{out}} (j^{out})^2 \cdot p^{out}(j^{out}) - \left(\sum_{j^{out}} j^{out} \cdot p^{out}(j^{out}) \right)^2, \\ (\sigma^{in})^2 &= \sum_{j^{in}} (j^{in})^2 \cdot p^{in}(j^{in}) - \left(\sum_{j^{in}} j^{in} \cdot p^{in}(j^{in}) \right)^2. \end{aligned} \quad (9)$$

When $r_n > 0$ ($r_n < 0$), network nodes tend to have similar (different) in- and out-degrees. For the special case where $r_n = 0$, there is no correlation between in- and out-degrees of each node. Hence $p_n(j^{in}, j^{out}) = p^{in}(j^{in}) p^{out}(j^{out})$.

By using the Pearson correlation, we can also define the *arc correlation* as follows:

$$r_e = \sum_{j^{in}, k^{out}} \left\{ \frac{1}{\sigma_e^2} \sum_{j^{out}, k^{in}} j^{out} k^{in} [p(j^{in}, j^{out}; k^{in}, k^{out}) - p_l(j^{in}, j^{out}) p_r(k^{in}, k^{out})] \right\}. \quad (10)$$

The normalizing factor $\sigma_e^2 = \sigma_e^{in} \cdot \sigma_e^{out}$ has

$$\begin{aligned} (\sigma_e^{out})^2 &= \sum_{j^{out}} (j^{out})^2 \cdot p_l(j^{in}, j^{out}) \\ &\quad - \left(\sum_{j^{out}} (j^{out}) \cdot p_l(j^{in}, j^{out}) \right)^2, \\ (\sigma_e^{in})^2 &= \sum_{k^{in}} (k^{in})^2 \cdot p_r(k^{in}, k^{out}) \\ &\quad - \left(\sum_{k^{in}} (k^{in}) \cdot p_r(k^{in}, k^{out}) \right)^2. \end{aligned} \quad (11)$$

When $r_e > 0$ ($r_e < 0$), a node with a certain out-degree trends to connect to a node with a similar (different) in-degree. For the special case where $r_e = 0$, there is no arc correlation. Therefore,

$$\begin{aligned} p(j^{in}, j^{out}; k^{in}, k^{out}) &= p_l(j^{in}, j^{out}) p_r(k^{in}, k^{out}) \\ &= p_n(j^{in}, j^{out}) \frac{j^{out}}{\langle k \rangle} p_n(k^{in}, k^{out}) \frac{k^{in}}{\langle k \rangle}. \end{aligned} \quad (12)$$

When both r_n and r_e equal to 0, denoting the distribution $p(j^{in}, j^{out}; k^{in}, k^{out})$ for this special uncorrelated case as $p_{uc}(j^{in}, j^{out}; k^{in}, k^{out})$, we have

$$\begin{aligned} p_{uc}(j^{in}, j^{out}; k^{in}, k^{out}) \\ &= p^{in}(j^{in}) p^{out}(j^{out}) \frac{j^{out} k^{in}}{\langle k \rangle^2} p^{in}(k^{in}) p^{out}(k^{out}). \end{aligned} \quad (13)$$

As pointed out in [21], to avoid neglecting a very small $p(j^{in}, j^{out}; k^{in}, k^{out})$ in numerical computations, we may adopt in calculations the relative joint degree distribution $f(j^{in}, j^{out}; k^{in}, k^{out})$ that

$$f(j^{in}, j^{out}; k^{in}, k^{out}) = \frac{p(j^{in}, j^{out}; k^{in}, k^{out})}{p_{uc}(j^{in}, j^{out}; k^{in}, k^{out})}. \quad (14)$$

3. Algorithm for generating a directed network with requested two-point correlation

Most existing methods for generating networks adopt the following framework: firstly a number of nodes with stubs are setup. Then two stubs are chosen to be connected with an edge between them at a certain probability (e.g., depending on the degrees of the two nodes [12,13]). We call such a framework as adopting a *node dominant* procedure.

In this section, by adopting the same framework, we propose a new algorithm for constructing directed networks with given joint degree distribution. The main idea of the algorithm is as follows.

Since

$$p(j^{in}, j^{out}; k^{in}, k^{out}) = p_l(j^{in}, j^{out}) p(k^{in}, k^{out} | j^{in}, j^{out}), \quad (15)$$

where $p_l(j^{in}, j^{out})$ is obtained from Eq. (1) and

$$p(k^{in}, k^{out} | j^{in}, j^{out}) = p(j^{in}, j^{out}; k^{in}, k^{out}) / p_l(j^{in}, j^{out}),$$

to generate a directed network with the requested joint degree distribution $p(j^{in}, j^{out}; k^{in}, k^{out})$, we select two nodes with nodal-degrees (j^{in}, j^{out}) and (k^{in}, k^{out}) at probabilities of $p_l(j^{in}, j^{out})$ and

$p(k^{in}, k^{out} | j^{in}, j^{out})$, respectively. Then an arc is inserted to connect these two nodes.

Note that for any finite-size network with given nodal-degrees, the joint degree distributions that can be exactly realized (hereafter termed as realizable distributions) form into a finite set. Other distributions however may be approximately reached. Since finding the realizable joint degree distribution closest to an arbitrarily requested one is difficult, we propose to find the realizable $p_n(j^{in}, j^{out})$ (hereafter denoted as $p_n^{(d)}(j^{in}, j^{out})$) close to the requested one and then calculate approximate, realizable $p_l^{(d)}(j^{in}, j^{out})$ and $p_r^{(d)}(j^{in}, j^{out})$, respectively. Below we present the construction algorithm in detail.

Assume that the number of nodes N and the joint degree distribution are given.

(1) Use Eqs. (1) and (2) and then Eqs. (5) and (6) to calculate $p_n(j^{in}, j^{out})$ and the number of the arcs M which equals to $N\langle k \rangle$. Calculate $p_n^{(d)}(j^{in}, j^{out})$. A simple method is to adjust each $p_n(j^{in}, j^{out})$ to the nearest or the second nearest integer multiples of $1/N$ subject to the conservation condition. Different algorithms can be developed for the adjustments. In our experiences, however, different algorithms make minor differences when N is large. Assign each node an in-degree and an out-degree simultaneously according to $p_n^{(d)}(j^{in}, j^{out})$.

(2) Calculate $p_l^{(d)}(j^{in}, j^{out})$ and $p_r^{(d)}(j^{in}, j^{out})$ by using Eqs. (3) and (4). Specifically, we have

$$p_l^{(d)}(j^{in}, j^{out}) = p_n^{(d)}(j^{in}, j^{out}) j^{out} / \langle k \rangle$$

and

$$p_r^{(d)}(j^{in}, j^{out}) = p_n^{(d)}(j^{in}, j^{out}) j^{in} / \langle k \rangle.$$

From Eq. (15), we have

$$\begin{aligned} p(k^{in}, k^{out} | j^{in}, j^{out}) &= \frac{p(j^{in}, j^{out}; k^{in}, k^{out})}{p_l(j^{in}, j^{out})} \\ &= f(j^{in}, j^{out}; k^{in}, k^{out}) p_r(k^{in}, k^{out}). \end{aligned}$$

Hence the approximate value of the conditional probability $p(k^{in}, k^{out} | j^{in}, j^{out})$, denoted as $p^{(d)}(k^{in}, k^{out} | j^{in}, j^{out})$, can be calculated as follows:

$$\begin{aligned} p^{(d)}(k^{in}, k^{out} | j^{in}, j^{out}) \\ &= \frac{f(j^{in}, j^{out}; k^{in}, k^{out}) p_r^{(d)}(k^{in}, k^{out})}{\sum_{k^{in}, k^{out}} f(j^{in}, j^{out}; k^{in}, k^{out}) p_r^{(d)}(k^{in}, k^{out})}. \end{aligned} \quad (16)$$

(3) Randomly choose a node with a degree (j^{in}, j^{out}) at a probability of $p_l^{(d)}(j^{in}, j^{out})$. Denote the chosen node as A and its actual degree as (j_A^{in}, j_A^{out}) . Then choose another node with degree (k^{in}, k^{out}) at a probability of $p^{(d)}(k^{in}, k^{out} | j_A^{in}, j_A^{out})$. Denote the actually chosen node as B . Insert an arc from A to B if the constraint of no self-loop and no multi-connection is fulfilled; otherwise, repeat Step (3).

(4) Once an arc is created, the joint degree distribution and the distributions of $p_l^{(d)}(j^{in}, j^{out})$ and $p^{(d)}(k^{in}, k^{out} | j^{in}, j^{out})$ for the arcs to be further inserted are revised accordingly. Specifically, denote $M' p_l^{(d)}(j^{in}, j^{out})$, $M' p_r^{(d)}(k^{in}, k^{out})$ and $M' p^{(d)}(k^{in}, k^{out} | j^{in}, j^{out})$ as the corresponding distributions of $p_l^{(d)}(j^{in}, j^{out})$, $p_r^{(d)}(k^{in}, k^{out})$ and $p^{(d)}(k^{in}, k^{out} | j^{in}, j^{out})$ when there are M' links remained to be inserted. The equations for updating the first two distributions when one more arc has been inserted are as follows:

$$M' p_l^{(d)}(j_A^{in}, j_A^{out}) = \frac{M' + 1}{M'} \left[M' p_l^{(d)}(j_A^{in}, j_A^{out}) - \frac{1}{M' + 1} \right],$$

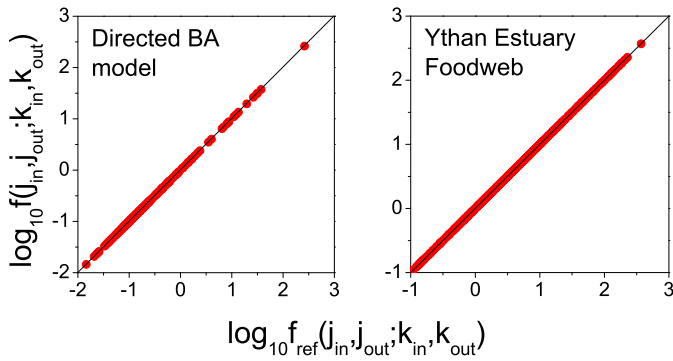


Fig. 1. (Color online.) The plot of the correlation function $f_{ref}(j_{in}^{in}, j_{out}^{in}; k_{in}^{in}, k_{out}^{in})$ of the empirical network versus the correlation function $f(j_{in}^{in}, j_{out}^{in}; k_{in}^{in}, k_{out}^{in})$ of the corresponding random network as generated by the algorithm for all indices j_{in}^{in} , j_{out}^{in} and k_{in}^{in} , k_{out}^{in} . The line $y = x$ is for reference.

$$M' p_r^{(d)}(k_B^{in}, k_B^{out}) = \frac{M'+1}{M'} \left[M' p_r^{(d)}(k_B^{in}, k_B^{out}) - \frac{1}{M'+1} \right]. \quad (17)$$

The coefficient $\frac{M'+1}{M'}$ is to keep the normalization of the distributions. With $M' p_r^{(d)}(k_{in}^{in}, k_{out}^{in})$, the distribution $M' p^{(d)}(k_{in}^{in}, k_{out}^{in} | j_{in}^{in}, j_{out}^{in})$ then can be recalculated by Eq. (16).

Repeat Steps (3) and (4) until the requested number of links have been inserted; or in other words, $M' = 0$.

We test the proposed algorithm in two example networks: (i) a *directed BA model*, which is generated in nearly the same way as that for the classic BA model by growth and preferential attachment [40]. The only difference is that coming with network growth, directed arcs rather than undirected edges are added pointing from newly added nodes to the existing ones. In such a network, the out-degree of the initial nodes is 0 while the out-degree of the rest nodes is m , where m denotes the number of arcs attached to each newly added node. Without loss of generality, we assume that m is also the number of initial nodes. The in-degree distribution still obeys the power law with an exponent of -3 . In our example, the network size $N = 1000$ and $m = 3$, thus the number of arcs $M \simeq 3000$; (ii) Ythan Estuary food-web network, the data of the network is downloaded from [41]. After omitting all the multiple connections and self-loops, it contains 135 nodes and 601 edges. We measure the joint degree distribution of each network and use it as input for the construction algorithm. The directed networks generated by the algorithm are expected to display the similar distributions as those of the original ones.

A proper test of the simulation results is to compare the relative joint degree distributions $f(j_{in}^{in}, j_{out}^{in}; k_{in}^{in}, k_{out}^{in})$ of the original networks and of the generated networks respectively. Denote $f(j_{in}^{in}, j_{out}^{in}; k_{in}^{in}, k_{out}^{in})$ in the original network as $f_{ref}(j_{in}^{in}, j_{out}^{in}; k_{in}^{in}, k_{out}^{in})$, and in the generated network as $f(j_{in}^{in}, j_{out}^{in}; k_{in}^{in}, k_{out}^{in})$. To facilitate comparisons, we introduce a parameter γ where

$$\gamma = \frac{\sum_{j_{in}^{in}, j_{out}^{in}, k_{in}^{in}, k_{out}^{in}} f_{ref} \cdot f}{(\sum_{j_{in}^{in}, j_{out}^{in}, k_{in}^{in}, k_{out}^{in}} f_{ref}^2 \sum_{j_{in}^{in}, j_{out}^{in}, k_{in}^{in}, k_{out}^{in}} f^2)^{1/2}}. \quad (18)$$

Having a value of γ closer to 1 generally denotes a better match between the original and the generated network. In our simulations, γ remains to be equal to 1 for all the example cases, which reveals a perfect agreement. A density plot of the reference relative joint degree distribution f_{ref} versus the resulting f is shown in Fig. 1, which verifies the satisfactory agreement between them.

4. Algorithm for tuning modularity without changing the two-point correlation

In this section, we introduce an algorithm for tuning the modularity without changing the two-point correlation. The main idea

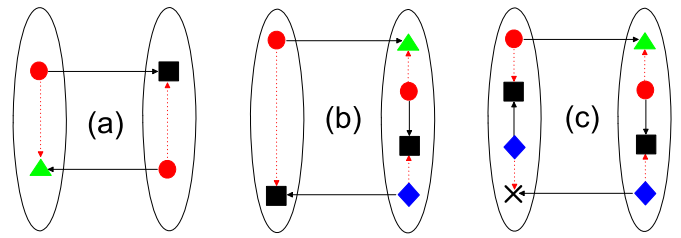


Fig. 2. Schemes of the algorithm of tuning modularity without changing the two-point correlation. Sub-figures (a), (b) and (c) shown three different cases. The same symbols represent the nodes with the same degree. Nodes in the same oval box means they are in the same community. To increase network modularity score, the black solid links are going to be replaced by the red dotted links, and vice versa. (For interpretation of colors in this figure, the reader is referred to the web version of this Letter.)

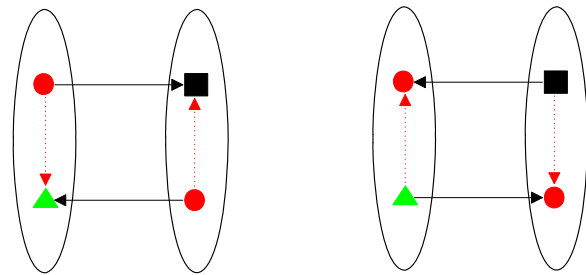


Fig. 3. (Color online.) The rewiring operations where directions of all the arcs are reversed in Fig. 2(a).

is as follows: detect communities and calculate the corresponding modularity score. Compare the calculated score to the target value. If the score needs to be increased, increase the connections within communities while reducing the connections between communities by rewiring some arcs. The operations are reversed if the modularity score is to be lowered. Repeat the above procedure until the target score is achieved or until no further feasible rewiring can be found though the target score is not achieved yet (in which case the algorithm fails). Without loss of generality, we let the modularity score be calculated by using the community detection method introduced in [42].

Fig. 2 illustrates a few simple rewiring operations we have adopted. Specifically, for any given two communities, let the same symbol represent the nodes with the same degree. To increase modularity score, we shall try to find a few arcs shown as solid lines and replace them by dotted lines; to decrease modularity score, the dotted lines are replaced by the solid lines. Apparently such rewiring operations do not change the two-point correlation of the network. For convenience, we term the approaches shown in Figs. 2(a), 2(b) and 2(c) as type-I, type-II and type-III rewiring respectively. Though more complicated rewiring operations certainly can be further introduced, in our practice the three types of rewiring can already ensure achieve quite high or low modularity scores in most network models. Therefore they are sufficient for most real-life applications.

Since the rewiring is in directed networks, the directions of the arcs are of important concern. Fig. 3 shows an example of rewiring where directions of all the arcs in Fig. 2(a) are reversed.

Below we present the algorithm in detail. To simplify the calculations, we adopt the simple rule that type-II rewiring is not adopted unless arc pairs for type-I rewiring cannot be found, and type-III rewiring is not adopted unless type-II rewiring has been exhausted.

(1) First separate the nodes into G groups (either randomly or by using the algorithm in [42]). Label the nodes from 1 to N , and the outgoing (incoming) links of each node i from 1 to k_i^{out} (k_i^{in}),

where k_i^{out} (k_i^{in}) denotes the number of outgoing (incoming) links of node i , $i = 1, 2, \dots, N$.

(2) Carry out a rewiring operation. For type-I rewiring, the procedure is as follows: Randomly choose two groups G_A and G_B . Denote the number of nodes in them as N_{G_A} and N_{G_B} , respectively. Randomly choose an arc j_A sourced from a certain node i_A in G_A and an arc j_B sourced from a certain node i_B in G_B . Starting from these two arcs, we search through all the arc pairs belonging to two different groups (e.g., in a round-robin manner) until a feasible solution for type-I rewiring is found.

One possible searching sequence is as follows: First fix j_A , then test through all the links sourced from i_B in the sequence of $j_B, \dots, k_{i_B}^{out}, 1, \dots, j_B - 1$. If not successful (i.e., no type-I rewiring can be carried out), move on to the next node of G_B and repeat the above procedure, until the search is successful or all the nodes in G_B has been tested in the sequence of $i_B, \dots, N_{G_B}, 1, \dots, i_B - 1$. The above procedure is repeated by firstly changing the selection of j_A in the sequence of $j_A, \dots, k_{i_A}^{out}, 1, \dots, j_A - 1$, and then changing the selection of i_A in the sequence of $i_A, \dots, N_{G_A}, 1, \dots, i_A - 1$. In this way, all the arc pairs in G_A and G_B are examined. Finally, the selection of G_B is changed in the sequence of $G_B, \dots, G, 1, \dots, G_B - 1$ and then G_A in the sequence of $G_A, \dots, G, 1, \dots, G_A - 1$ subject to the condition that $G_A \neq G_B$. All the arc pairs eligible for type-I rewiring are therefore exhaustively searched.

Once a feasible solution is found, the rewiring operation is carried out accordingly. If all arc pairs have been exhaustively searched yet no feasible solution has been found, the algorithm will proceed to carry out type-II rewiring, and later if necessary, type-III rewiring as well.

Type-II and type-III rewiring can also apply round-robin exhaustive search. Detailed descriptions of them are very lengthy and therefore omitted.

(3) Calculate the temporary modularity score Q_{temp} of the network. The modularity score function is defined as [42]

$$Q_{temp} = \frac{1}{M} \sum_{i,j} \left[a_{ij} - \frac{k_i^{in} k_j^{out}}{M} \right] \delta_{g_i, g_j}, \quad (19)$$

where a_{ij} equals to one if there is an arc from node j to node i and zero otherwise; M denotes the number of arcs in the network; δ_{ij} denotes the Kronecker delta; and g_i the label of group to which node i is assigned. A higher value of modularity score corresponds to a stronger community structure. Compare Q_{temp} with the requested modularity level, denoted as Q_{obj} . If they match each other within a predefined precision range, go to Step (4); otherwise, go to Step (2). The algorithm however should be terminated if all the three types of rewiring have been exhausted.

(4) Use the community detection method in [42] to detect communities and obtain the corresponding modularity score, denoted as Q . If the requested modularity is achieved, i.e., $|Q - Q_{obj}| \leq \epsilon$ where ϵ is the requested precision, then stop; otherwise, take the newly detected communities as the starting groups and go to Step (2).

The iterative approach of Steps (2)–(4) makes sure that the communities for modularity score calculation are detected by using a well-accepted method rather than arbitrarily defined.

As mentioned earlier, more complicated rewiring operations can be designed if such is needed. In our practice, however, even before we need to adopt type-III rewiring, type-I and type-II rewiring usually can already drive the network to be with a rather high or low modularity score, e.g., 0.65 in the directed BA model (see below). Considering that the modularity scores of real-life networks do not often go extremely large or small [43], the proposed algorithm is expected to have a wide applicable range.

Another concern in the algorithm design is that, since in each iteration the modularity score is recalculated based on reseeded

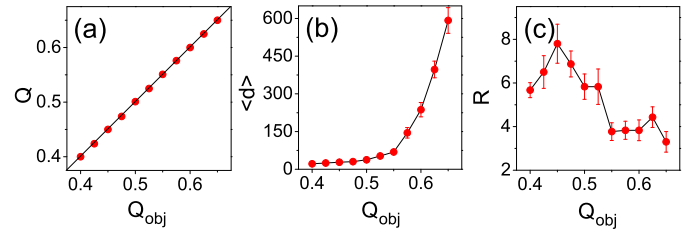


Fig. 4. (Color online.) Results of modularity adjustment of directed BA model. (a) Q_{obj} is the objective modularity score. Q is the measurement of the networks generated by our algorithm. The sizes of the error bar are smaller than those of the symbols. (b) Average number of tested arcs during rewiring process for each Q_{obj} . (c) Number of iterations R versus Q_{obj} . The results are averaged over 30 realizations with the precision $\epsilon = 0.003$. Lines are just guide for the eyes.

communities, there may be fluctuations in the modularity scores preventing the algorithm from converging to the objective value. Theoretically speaking, the problem can be fixed by limiting to a small number of rewirings in each iteration, though at a cost of longer computational time. In our practice, such a strategy has never been implemented: the algorithm always converges quickly. More details are presented in Fig. 4.

To validate the algorithm, we firstly test it in the directed BA model. An objective modularity value is set, e.g., $Q_{obj} = 0.4, \dots, 0.65$. Then the proposed algorithm is adopted to rewire the arcs until Q_{obj} is achieved. In this Letter, unless otherwise specified, we start the calculations by randomly separating all the network nodes into four communities with equal or nearly the equal sizes. Fig. 4(a) shows the comparison between Q_{obj} and the modularity score Q obtained from the proposed method. The results come from 30 independent realizations. We can see they are in good agreement.

Now we briefly discuss the complexity of the proposed algorithm. From Eq. (19), it can be seen that the maximum number of rewirings needed is in the order of $O(M)$. To find a set of rewirable arcs, in the worst case we may need to search through all the arcs. In average this number however is much lower. Use the type-I rewiring as an example: when there are l pairs of rewirable arcs, approximately M^2/l arcs need to be tested before a pair of rewirable arcs are found. Denote the average number of arcs tested for each rewiring as $\langle d \rangle$. The average complexity of arc rewiring operations is then $O(M\langle d \rangle)$. For most cases, $\langle d \rangle$ is much smaller than M . To demonstrate, we show in Fig. 4(b) the average number of arcs tested for finding each set of rewirable arcs during the rewiring process in the directed BA model. We observe that $\langle d \rangle$ increases with a larger value of Q_{obj} , which can be easily understood: achieving a high modularity level tends to exhaust rewirable arcs and consequently makes the later-stage search more difficult. When $\langle d \rangle \ll M$, the main computational time in fact is for executing the community detection method with a moderate complexity of $O((M + N)N \log(N))$, essentially identical to that of the corresponding algorithm for undirected networks [28]. Apparently, the number we have to run the community detection algorithm equals to the number of iterations. Fig. 4(c) shows the relation between the number of iterations, denoted as R , and Q_{obj} . We see that the average value of R peaks at $Q_{obj} = 0.475$ and goes below 5 when $Q_{obj} > 0.525$. Such observations can be understood: when Q_{obj} is relatively small, in each iteration the obtained community structure in Step (3) is less distinct, which tends to induce larger fluctuations in community detections in Step (4) and consequently affects the calculation of modularity scores. When Q_{obj} is large, on the other hand, the dense connections within most communities make them be easily detectable by the community detection algorithm. The calculations therefore converge quickly. Note that even in most difficult cases the propose algorithm converges quickly in an average of no more than 10 iterations.

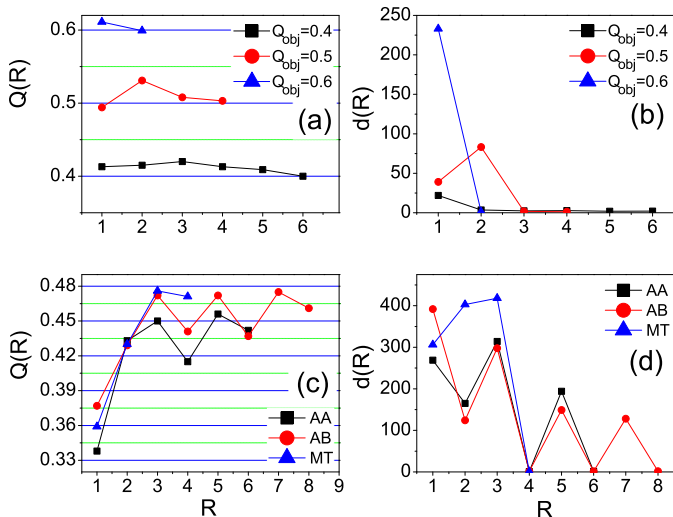


Fig. 5. (Color online.) Temporary results of modularity adjustment in a single realization. In the R -th iteration, $Q(R)$ is the modularity score detected in Step (4) and $d(R)$ is the average number of arcs tested for each rewiring. (a) and (b) show the results obtained in directed BA model with $Q_{obj} = 0.4, 0.5$ and 0.6 . (c) and (d) show the results obtained from three metabolic networks with code names AA, AB and MT respectively. Lines are just guide for the eyes.

Fig. 5 shows in more detail the temporary results of each iteration in the directed BA model as well as a few metabolic networks [3]. The metabolic network data is downloaded from [44]. For each metabolic network, we firstly calculate its two-point correlation. Then we use the algorithm in Section 3 to generate a new network with the same correlation. Finally we measure the modularity of the original network and then tune the modularity of the new network to be the same.

Fig. 5(a) shows the modularity scores detected in Step (4) in the R -th iteration, denoted as $Q(R)$, in the directed BA model where $Q_{obj} = 0.4, 0.5$ and 0.6 respectively. We see that $Q(R)$ gets close to Q_{obj} in the first iteration, and after a few iterations with small fluctuations, quickly converges to the objective value. **Fig. 5(b)** shows in the same network and for the same values of Q_{obj} , the average numbers of arcs tested for each rewiring in different iterations, denoted as $d(R)$. We see that $d(R)$ tends to be larger in the first several iterations. This can be explained: in the first several iterations, a large number of arcs need to be rewired before the objective value can be reached. As discussed earlier, having a larger number of rewiring operations tends to exhaust rewirable arcs and consequently makes the later-stage search more difficult.

Figs. 5(c) and 5(d) show the simulation results in the metabolic networks. Simulation results for three different networks with code names AA, AB and MT respectively are illustrated. We see that basically all the conclusions above still hold, though fluctuations in simulation results are larger, especially in network AB. More simulation results for a larger number of metabolic networks are summarized in **Table 1**, which shows a few important parameters and calculation results. Good matching has been consistently achieved.

Finally, Ythan Estuary food web is also simulated. Again we calculate its modularity score and then adjust the modularity of the corresponding network generated in Section 3. The modularity score of the resulting network is 0.362, which matches well with the original value of 0.361.

5. Conclusion

In this Letter, we presented an algorithm for generating directed networks with given two-point correlation defined by joint degree distribution. Furthermore, an algorithm was developed to

Table 1

Summaries of the original and generated networks respectively: number of nodes N and arcs M , modularity value Q_r of original networks and Q_g of the generated networks, and the value of the parameter γ .

Organism code	N	M	Q_r	Q_g	γ
AA	414	1911	0.445	0.442	1.000
AB	620	2516	0.459	0.461	0.998
AG	653	2754	0.487	0.487	0.999
AT	348	1424	0.460	0.460	1.000
BS	1048	4680	0.476	0.477	0.998
CA	734	3137	0.468	0.470	0.997
CE	618	2659	0.469	0.469	0.999
CJ	612	2468	0.482	0.481	0.998
CT	563	2302	0.483	0.483	0.999
CY	801	3414	0.464	0.465	0.998
DR	1086	4815	0.477	0.478	0.997
EF	601	2589	0.466	0.468	0.999
EN	377	1704	0.436	0.434	1.000
HI	511	2421	0.433	0.431	0.998
MB	421	1894	0.441	0.441	1.000
ML	417	1904	0.437	0.437	1.000
MT	580	2738	0.469	0.471	1.000
NM	375	1798	0.418	0.420	0.998
OS	289	1218	0.470	0.472	0.999
PF	313	1384	0.450	0.452	0.999
PG	417	1835	0.447	0.449	0.994
PH	320	1401	0.456	0.459	1.000
PN	409	1962	0.419	0.422	0.996
RC	664	3139	0.451	0.450	0.999
RP	206	824	0.485	0.487	1.000
SC	552	2789	0.439	0.442	1.000
ST	395	1915	0.421	0.420	0.996
TH	427	2022	0.451	0.452	1.000
TM	333	1543	0.454	0.451	1.000
TP	204	864	0.454	0.457	0.997

tune the modularity without changing the two-point correlation. Artificial and real-life networks have been adopted to test the proposed algorithms. It is manifested that the correlation and modularity of the generated networks coincide with those of the original ones. As degree-degree correlation and modularity may affect many dynamic processes in complex systems, our algorithms are expected to provide a useful tool for in-depth studies on such effects.

Acknowledgement

This work is supported in part by the Singapore A*STAR grant BMRC 06/1/21/19/457.

References

- [1] L.F. Lago-Fernandez, R. Huerta, F. Corbacho, J.A. Siguenza, Phys. Rev. Lett. 84 (2000) 2758.
- [2] J.W. Bohland, A.A. Minai, Neurocomputing 38 (2001) 489.
- [3] H. Jeong, B. Tombor, R. Albert, Z.N. Oltvai, A.-L. Barabási, Nature (London) 407 (2000) 651.
- [4] R.V. Kulkarni, E. Almaas, D. Stroud, Phys. Rev. E 61 (2000) 4268.
- [5] M. Girvan, M.E.J. Newman, Proc. Natl. Acad. Sci. USA 99 (2002) 7821.
- [6] R. Albert, H. Jeong, A.L. Barabási, Nature (London) (1999) 130.
- [7] Réka Albert, István Albert, Gary L. Nakarado, Phys. Rev. E 69 (2004) 025103(R).
- [8] Hyejin Youn, Michael T. Gastner, Hawoong Jeong, Phys. Rev. Lett. 101 (2008) 128701; G. Li, S.D.S. Reis, A.A. Moreira, S. Havlin, H.E. Stanley, J.S. Andrade Jr., Phys. Rev. Lett. 104 (2010) 018701.
- [9] R. Albert, A.-L. Barabási, Rev. Mod. Phys. 74 (2002) 47.
- [10] E.A. Bender, E.R. Canfield, J. Combin. Theory Ser. A 24 (1978) 296.
- [11] B. Bollobas, Eur. J. Comb. 1 (1980) 311.
- [12] M. Molloy, B. Reed, Random Struct. Algorithms 6 (1995) 161.
- [13] M. Molloy, B. Reed, Combin. Probab. Comput. 7 (1998) 295.
- [14] M. Catanzaro, M. Boguna, R. Pastor-Satorras, Phys. Rev. E 71 (2005) 027103.
- [15] M.A. Serrano, M. Boguna, Phys. Rev. E 72 (2005) 036133.
- [16] M.E.J. Newman, Phys. Rev. Lett. 89 (2002) 208701.
- [17] A. Vázquez, R. Pastor-Satorras, A. Vespignani, Phys. Rev. E 65 (2002) 066130.
- [18] M.E.J. Newman, SIAM Rev. 45 (2003) 167.

- [19] M.E.J. Newman, Phys. Rev. E 67 (2003) 026126.
- [20] R. Xulvi-Brunet, I.M. Sokolov, Phys. Rev. E 70 (2004) 066102.
- [21] S. Weber, M. Porto, Phys. Rev. E 76 (2007) 046111.
- [22] A. Pusch, S. Weber, M. Porto, Phys. Rev. E 77 (2008) 017101.
- [23] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, J. Wiener, Comput. Netw. 33 (2000) 309.
- [24] B. Tadic, Physica A 293 (2001) 273.
- [25] O. Sporns, G. Tononi, G.M. Edelman, Neural Networks 13 (2000) 909.
- [26] M.E.J. Newman, Eur. Phys. J. B 38 (2004) 321.
- [27] L. Danon, J. Duch, A. Diaz-Guilera, A. Arenas, J. Stat. Mech. (2005) 09008.
- [28] M.E.J. Newman, Proc. Natl. Acad. Sci. USA 103 (2006) 8577.
- [29] L.H. Hartwell, J.J. Hopfield, S. Leibler, A.W. Murray, Nature (London) 402 (1999) C47.
- [30] D.M. Wolf, A.P. Arkin, Curr. Opin. Microbiol. 6 (2003) 125.
- [31] A. Kreimer, E. Borenstein, U. Gophna, E. Ruppin, Proc. Natl. Acad. Sci. USA 105 (2008) 9676.
- [32] M. Girvan, M.E.J. Newman, Proc. Natl. Acad. Sci. USA 99 (2002) 7821.
- [33] A. Lancichinetti, S. Fortunato, F. Radicchi, Phys. Rev. E 78 (2008) 046110;
- A. Lancichinetti, S. Fortunato, Phys. Rev. E 80 (2009) 016118;
- A. Lancichinetti, S. Fortunato, Phys. Rev. E 80 (2009) 056117.
- [34] J. Gomez-Gardenes, M. Campillo, L.M. Floria, T. Moreno, Phys. Rev. Lett. 95 (2005) 098104.
- [35] J. Tanimoto, Phys. Rev. E 76 (2007) 021126.
- [36] A. Pusch, S. Weber, M. Porto, Phys. Rev. E 77 (2008) 036120.
- [37] J. Tanimoto, Physica A 388 (2009) 953.
- [38] J. Kleinberg, Complex networks and decentralized search algorithms, in: Proceedings of the International Congress of Mathematicians (ICM), vol. 3, 2006, pp. 1019–1044.
- [39] Mercedes Pascual, Jennifer A. Dunne (Eds.), Ecological Networks: Linking Structure to Dynamics in Food Webs, Oxford University Press, 2006.
- [40] A.-L. Barabási, R. Albert, Science 286 (1999) 509;
- A.-L. Barabási, R. Albert, H. Jeong, Physica A 272 (1999) 173.
- [41] <http://www.cosinproject.org>.
- [42] E.A. Leicht, M.E.J. Newman, Phys. Rev. Lett. 100 (2008) 118703.
- [43] M.E.J. Newman, M. Girvan, Phys. Rev. E 69 (2004) 026113.
- [44] <http://www.nd.edu/networks>.