IOP PUBLISHING JOURNAL OF PHYSICS A: MATHEMATICAL AND THEORETICAL

J. Phys. A: Math. Theor. **41** (2008) 224006 (15pp) doi:10.1088/1751-8113/41/22/224006

The generation of random directed networks with prescribed 1-node and 2-node degree correlations

Gorka Zamora-López[1], **Changsong Zhou**[2], **Vinko Zlatić**[3] **and Jürgen Kurths**[1]

[1] Institute of Physics, University of Potsdam, PO Box 601553, 14415 Potsdam, Germany
[2] Department of Physics, Hong Kong Baptist University, Kowloon Tong, Hong Kong
[3] Rudjer Bošković Institute, PO Box 180, HR-10002 Zagreb, Croatia

Received 18 October 2007, in final form 12 December 2007
Published 21 May 2008
Online at stacks.iop.org/JPhysA/41/224006

Abstract

The generation of random networks is a very common problem in complex network research. In this paper, we have studied the correlation nature of several real networks and found that, typically, a large number of links are deterministic, i.e. they cannot be randomized. This finding permits fast generation of ensembles of maximally random networks with prescribed 1-node and 2-node degree correlations. When the introduction of self-loops or multiple-links are not desired, random network generation methods typically reach blocked states. Here, a mechanism is proposed, the 'force-and-drop' method, to overcome such states. Our algorithm can be easily simplified for undirected graphs and reduced to account for any subclass of 2-node degree correlations.

PACS numbers: 87.18.Sn, 89.75.Da, 89.75.Hc, 05.45.Xt

(Some figures in this article are in colour only in the electronic version)

1. Introduction

Random networks play a fundamental role in the development of network theory and its real life applications. Whether it is for significance testing of real data analysis or for numerical corroboration of theoretical results, researchers often need to face the practical problem of generating random networks. The *random graph* is probably the most studied network model ever. It consists of an initially empty network of N unconnected nodes where a total of L links are randomly introduced one-by-one so that all pairs of nodes are connected with equal probability [1–3]. Very often it is desirable to generate networks that, while having specified basic properties, are maximally random. Random networks with prescribed degree sequence are commonly considered in the literature as a manner to test the significance of measured properties. Degree assortativity, a measure of 2-node degree correlations, represents

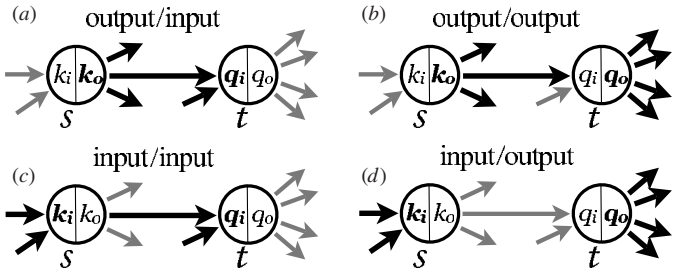

Figure 1. 2-node 2-degree correlations of *neighbouring nodes* in directed networks. Links corresponding to correlated degrees are coloured black.

the tendency of nodes to connect to other nodes of (dis-)similar degree. A network is called *assortative* when nodes of similar degree are connected with each other, and *disassortative* when low degree nodes preferentially link to high degree ones and not among them. As summarized in [3], social networks tend to be assortative while other real network classes tend to be disassortative, e.g. technological and biological networks.

When the network is directed, the input degree $k_i(s)$ and the output degree $k_o(s)$ of an individual node s are not necessarily equal, giving rise to different conditions of 1-node correlations. The 2-node correlations become also splitted into different classes. Imagine a link $s \to t$ pointing from a source node s with input and output degrees (k_i, k_o) to a target node t with degrees (q_i, q_o). When all the four values (k_i, k_o, q_i, q_o) are correlated, both 1-node and 2-node 4-point degree correlations are present. For notation simplicity, we will call to this general case as the $1n2n$ correlation class. If only one of the degrees $\{k_i, k_o\}$ is correlated to one of the $\{q_i, q_o\}$ then different special classes of 2-node 2-point degree correlations may happen as depicted in figure 1.

In our recent paper [4], the expected reciprocity of directed networks under different classes of 1-node and 2-node degree correlations has been studied. The results clearly show that both the 1-node and the 2-node degree correlations influence the outcome of graph measures. In order to numerically corroborate our theoretical results, several algorithms have been presented to randomize real networks while conserving only the desired class of correlations. However, those methods differ from each other and this makes their application confusing. In this paper, we present a general framework for the generation of random directed networks with $1n2n$ degree correlations. The algorithm is very easily simplified to account for any of the 2-node 2-point correlation classes. First, a general algorithm will be presented that allows the introduction of self-loops (links connecting a node to itself) and multiple-links (more than one link between two nodes). In section 3, the correlation nature of real networks is explored searching for hints to improve the method. As the main result, in sections 4 and 5 strategies to avoid self-loops and multiple-links are presented together with the general algorithmic implementation.

2. Generation of random networks with prescribed $1n2n$ correlations

Given a network of N nodes and L links we characterize the 1-node degree correlations by the number of nodes $N(\boldsymbol{k}) = N(k_i, k_o)$ having in-degree k_i and out-degree k_o. Let us characterize the 2-node degree correlations by the number of links $L(\boldsymbol{k} \to \boldsymbol{q})$ pointing from nodes with degrees $\boldsymbol{k} = (k_i, k_o)$ to nodes with degrees $\boldsymbol{q} = (q_i, q_o)$. All these quantities are easy

to calculate from real networks and contain all the necessary information about the degree correlations. For theoretical applications, one can define the desired correlation structure by taking into account that the following conservation rules do necessarily hold:

$$N = \sum_{\boldsymbol{k}'} N(\boldsymbol{k}') \qquad \text{and} \qquad L = \sum_{\boldsymbol{k}',\boldsymbol{q}'} L(\boldsymbol{k}' \to \boldsymbol{q}'), \tag{1}$$

$$k_o N(\boldsymbol{k}) = \sum_{\boldsymbol{q}'} L(\boldsymbol{k} \to \boldsymbol{q}') \qquad \text{and} \qquad \sum_{\boldsymbol{k}'} \mathrm{L}(\boldsymbol{k}' \to \boldsymbol{q}) = \mathrm{q_i N}(\boldsymbol{q}). \tag{2}$$

Equations (1) are simply the conservation of nodes and links, and equations (2) arise from the fact that the set $\mathcal{N}(\boldsymbol{k})$ of nodes with degrees $\boldsymbol{k}$ necessarily project $k_o N(\boldsymbol{k})$ edges. Equivalently, the same set of nodes necessarily receives $k_i N(\boldsymbol{k})$ edges.

We remind now that a network with prescribed $1n2n$ degree correlations is considered as *maximally* random when *any of the nodes s in the set* $\mathcal{N}(\boldsymbol{k})$ *is equally likely connected to any of the nodes t in the set* $\mathcal{N}(\boldsymbol{q})$. This probability is expressed by,

$$p(\boldsymbol{k} \to \boldsymbol{q}) = \frac{L(\boldsymbol{k} \to \boldsymbol{q})}{N(\boldsymbol{k})N(\boldsymbol{q})}, \tag{3}$$

where $N(\boldsymbol{k})N(\boldsymbol{q})$ is the total number of possible links from nodes with $\boldsymbol{k}$ to nodes with $\boldsymbol{q}$. When the network is directed, $L(\boldsymbol{k} \leftarrow \boldsymbol{q})$ does not necessarily equal $L(\boldsymbol{k} \to \boldsymbol{q})$. We refer the reader to appendix A for computational tips on implementing $N(\boldsymbol{k})$ and $L(\boldsymbol{k} \to \boldsymbol{q})$.

2.1. Basic algorithm

Once $N(\boldsymbol{k})$ and $L(\boldsymbol{k} \to \boldsymbol{q})$ have been calculated out of a real network or arbitrarily specified, an empty network of N nodes is created. Following $N(\boldsymbol{k})$, all the nodes are assigned their final degrees $\boldsymbol{k}$. Then L links are introduced in the following manner:

(1) One node is chosen at random to act as the source s. As the final degrees have been previously assigned, we know that s will have degrees $\boldsymbol{k}'$.
(2) A list of possible target nodes, *tlist*, is constructed containing all the nodes in the sets $\mathcal{N}(\boldsymbol{q}')$ such that $L(\boldsymbol{k}' \to \boldsymbol{q}') > 0$.
(3) From *tlist* one target node is chosen at random, t', and the connection $s' \to t'$ is created.

The algorithm needs to keep track of the nodes that cannot accept more connections and of those $(\boldsymbol{k}, \boldsymbol{q})$ combinations for which all the $\boldsymbol{k} \to \boldsymbol{q}$ links have already been introduced. Therefore, we define the *free* in-degrees fk_i as the remaining number of incoming links that a node can still receive and the *free* out-degrees fk_o as the number of its remaining outgoing links[4]. Equivalently, $fL(\boldsymbol{k} \to \boldsymbol{q})$ counts the number of $\boldsymbol{k} \to \boldsymbol{q}$ links that are yet to be introduced. With these quantities in hand, s is selected in step-1 out of those nodes with $fk_o(s) > 0$. In step-2, only those $\boldsymbol{q}'$ are considered for which $fL(\boldsymbol{k}' \to \boldsymbol{q}') > 0$, and from $\mathcal{N}(\boldsymbol{q}')$, only the nodes with $fk_i(t') > 0$ are included into *tlist*.

2.2. Performance and optimization of the basic algorithm

So far, an algorithm implemented following these rules generates random networks with the prescribed $1n2n$ correlations in L iterations. Each iteration, however, comprises of three steps and step-2 is the slowest of them. In order to construct *tlist* the whole structure

[4] We use the term *free* because fk_i and fk_o are the number of remaining 'free stabs' in analogy to the configuration model.

$fL(\boldsymbol{k} \to \boldsymbol{q})$, the largest structure of all, needs to be fully iterated searching for those $\boldsymbol{q}'$ for which $fL(\boldsymbol{k}' \to \boldsymbol{q}') > 0$. Being $\boldsymbol{k}'$ the degrees of the chosen node s, one would typically code the step-2 as:

```
tlist = empty
for each (k'', q'') combination in fL(k → q) :
    if k'' == k' :
        if fL(k' → q'') > 0 :
            for node in N(q'') :
                if fk_i(node) > 0 :
                    include node in tlist.
```

As we will later discuss, the size of $L(\boldsymbol{k} \to \boldsymbol{q})$ largely depends on the precise 2-node correlation structure of the real network. Hence, the complexity of each iteration is distinct for different networks. In the worst possible situation, when $L(\boldsymbol{k} \to \boldsymbol{q}) = 1$ for all $(\boldsymbol{k}, \boldsymbol{q})$ combinations, then its size equals L and the generation process, described as it is, has a complexity of the order $\mathcal{O}(L^2)$. The introduction of the following additional structures largely helps improving the performance of step-2:

$$fqlist = \{\boldsymbol{k}\colon \{\boldsymbol{q}' \text{ such that } fL(\boldsymbol{k} \to \boldsymbol{q}' > 0)\}\}$$

For each of the degrees $\boldsymbol{k}$ present in the network, $fqlist(\boldsymbol{k})$ contains the list of degrees $\boldsymbol{q}'$ for which the random process still needs to introduce links of the type $\boldsymbol{k} \to \boldsymbol{q}'$.

$$\boldsymbol{f}\mathcal{N}(\boldsymbol{k}) = \{\boldsymbol{k}\colon \{t' \in \mathcal{N}(\boldsymbol{k}) \text{ such that } fk_i(t') > 0\}\}$$

contains the updated sets $\mathcal{N}(\boldsymbol{k})$ of nodes with degrees $\boldsymbol{k}$ that still have free place for incoming links.

The introduction of these look-up tables allows us to simplify the step-2 and replace the highly expensive iteration through $fL(\boldsymbol{k} \to \boldsymbol{q})$ by few memory access operations:

```
tlist=empty
for q' in fqlist (k') :
    for node in f N(q') :
        include node in tlist.
```

Furthermore, if these look-up tables are properly updated, they are smaller after each iteration and step-2 becomes faster as the process advances.

Generating an ensemble of random networks requires an initial preparation that needs to be performed only once. Given a network in adjacency list form, the input and output degrees can be obtained in $\mathcal{O}(N)$, and the 2-node degree correlations $L(\boldsymbol{k} \to \boldsymbol{q})$ are obtained in $\mathcal{O}(L)$, see appendix A. Creation of the look-up tables *fqlist* and $f\mathcal{N}(k)$ requires also $\mathcal{O}(L)$, see appendix B.

The basic algorithm presented here is trivially reduced to generate undirected networks with prescribed 2-node degree correlations. In such a case $k_i(s) = k_o(s)$ for all node s and the 1-node degree correlations disappear. The quantity $N(\boldsymbol{k})$ becomes the typical degree distribution $N(k)$. The only drawback is that, described as it is, the current algorithm allows for the introduction of self-loops and multiple-links. For many applications this is not a problem at all, but very often one needs to avoid such types of connections. In section 5, we will introduce an extended algorithm that also avoids the formation of self-loops and multiple-links. Before that, let us first turn our attention to the correlation structure of real networks.

Table 1. The real networks analysed in this paper are very different in sizes N, link densities ρ and degree correlation structures. The number of embedded deterministic links L_{det} is also shown for comparison.

Network	N	L	ρ	L_{det}	L_{det}/L
Cortical networks					
Cat [5]	53	826	0.300	654	0.792
Macaque [6]	70	747	0.155	569	0.762
Food webs [7]					
St. Martin Isl.	45	224	0.1131	139	0.621
St. Marks sea.	49	223	0.0948	146	0.655
Grassland	88	137	0.0179	9	0.0657
Ythan estuary	135	597	0.0330	267	0.447
Silwood Park	154	365	0.0155	33	0.315
Little Rock lake	183	2476	0.0743	2149	0.868
World-trade-webs [8]					
Year 1948	82	2539	0.382	2433	0.958
Year 2000	190	20 105	0.560	19 138	0.952
Wikipedia Website [9]					
Chinese	18 089	332 434	0.0010	96 611	0.291
Portuguese	30 374	373 215	0.0004	78 152	0.209
Spanish	39 562	655 615	0.0004	166 073	0.253

3. Exploring the correlation nature of real networks

In this section, we will study the correlations of several real networks summarized in table 1. They represent a wide range of network classes: there are both small networks with large density and with low density of links. The density of links $\rho = \frac{L}{N(N-1)}$ is the fraction between L and the total number of possible links, $N(N-1)$. The three Wikipedia networks display the typical characteristics of many large networks: very low density and scale-free degree distribution.

We have measured $N(\boldsymbol{k})$ of all these real networks (appendix A). In general, it is observed that most of the nodes possess a unique degree $\boldsymbol{k}' = (k_i', k_o')$ that no other node has. This means that for most of the (k_i, k_o) combinations $N(\boldsymbol{k}) = 1$ as observed in the histograms of figure 2 (left column). From figure 2(*a*) we see that in both cortical networks over 70% of the nodes have a unique combination of degrees (k_i', k_o') and only very few nodes share the same combination with another node, i.e., $N(\boldsymbol{k}') = 2$. This observation has a very large impact on the 2-node correlation structure of the networks. As $N(\boldsymbol{k})N(\boldsymbol{q})$ is the total number of possible $\boldsymbol{k} \to \boldsymbol{q}$ links, when any of the following conditions hold,

$$L(\boldsymbol{k} \to \boldsymbol{q}) = N(\boldsymbol{k})N(\boldsymbol{q}) \qquad \text{if} \quad \boldsymbol{k} \neq \boldsymbol{q}, \tag{4}$$

$$L(\boldsymbol{k} \to \boldsymbol{q}) = N(\boldsymbol{k})[N(\boldsymbol{k}) - 1] \qquad \text{if} \quad \boldsymbol{k} = \boldsymbol{q}, \tag{5}$$

then all nodes $s \in \mathcal{N}(\boldsymbol{k})$ are connected to all nodes $t \in \mathcal{N}(\boldsymbol{q})$. We refer to this local all-to-all connections as *deterministic links*. When for most of the degrees $\boldsymbol{k}$, $N(\boldsymbol{k}) = 1$, it is clear that many links connect $(\boldsymbol{k}, \boldsymbol{q})$ combinations that are unique giving rise to deterministic links. See, for example, figure 2(*b*) for the cortical networks. Note that when all nodes have unique degrees, i.e. $N(\boldsymbol{k}) = 1$ for all $\boldsymbol{k}$, then all links are necessarily deterministic. In general, the lower the $N(\boldsymbol{k})$ are, the higher is the chance for deterministic links. In all the real networks

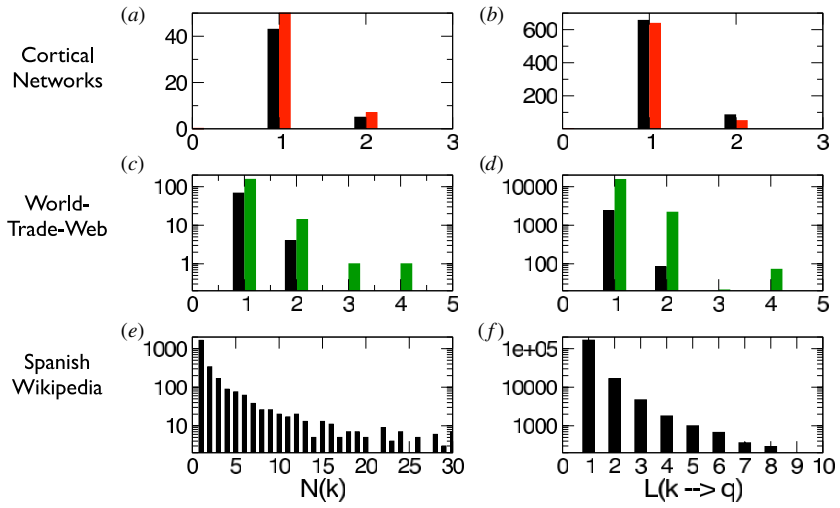

Figure 2. (left column) Number of times that any combination of in- and out-degrees $\boldsymbol{k} = (k_i, k_o)$ is used by 1 node, 2 nodes, etc. (right column) the number of times that any given class of links $\boldsymbol{k} \to \boldsymbol{q}$ is represented by 1 link, 2 links, etc.

analysed here there exists a clear tendency for low values of $N(\boldsymbol{k})$, although to different extent. Figures 2(*c*) and (*d*) show the histograms for both world-trade-webs. Most of the nodes have also a unique combination of degrees. There are only 10 degree combinations $\boldsymbol{k} = (k_i, k_o)$ that two nodes have, i.e. $N(\boldsymbol{k}) = 2$. It occurs only once that three or four nodes have the same degrees, i.e., $N(\boldsymbol{k}) = 3$ or $N(\boldsymbol{k}) = 4$. Figures 2(*e*) and (*f*) show the histograms for the Spanish Wikipedia network. The distributions are much broader in this case, but still, the low values of $N(\boldsymbol{k})$ and $L(\boldsymbol{k} \to \boldsymbol{q})$ dominate.

3.1. Embedded deterministic networks

These findings are fundamental for our goal of obtaining an efficient random network generator algorithm because it can simply obviate the deterministic links. For all the real networks in table 1 we have extracted their embedded *deterministic networks*, say, the subnetwork composed uniquely of those links that are deterministic after conditions (4) and (5). Let L_{det} be the number of deterministic links and N_{det} the size of the deterministic network, i.e., the number of nodes that send or receive at least one deterministic link. In figure 3(*a*) the relative sizes of the embedded networks are plotted for comparison. In some cases a rather small part of the original connections is deterministic, e.g. 20%–30% in the Wikipedias. In other cases the fraction is much larger. In both cortical networks near 70% of the links are deterministic while, in the world-trade-webs they even account for 95% of the connectivity!

It is not trivial to predict the number of deterministic links L_{det}. It seems to depend on the in- and out-degree distributions, the 1-node and the 2-node degree correlations. Let us try gaining some intuitive understanding. In a network of size N a node can maximally connect to other $N-1$ nodes. If $N(k_i)$ and $N(k_o)$ are the input and output degree distributions then $N = \sum_{k_i} N(k_i) = \sum_{k_o} N(k_o)$ must hold. Equivalently $N = \sum_{k_i,k_o} N(k_i, k_o) = N(\boldsymbol{k})$. This conservation rule implies that, the N nodes are distributed among the space of all possible (k_i, k_o) combinations. When the degree distributions are very broad, $k_{\min} \to 0$ and $k_{\max} \to N-1$, then the space of possible (k_i, k_o) combinations is very large and it is very

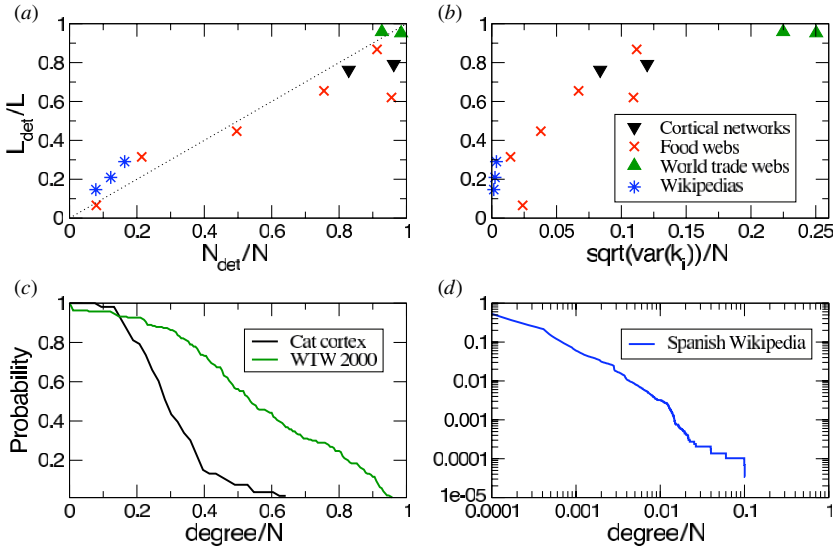

Figure 3. Deterministic Networks. (*a*) Relative size of the embedded deterministic networks for several real networks. The dotted line is only a reference. (*b*) Dependence of the number of deterministic links $L_{\rm det}$ on the variance of the in-degrees. (*c*) and (*d*) Cumulative in-degree distributions.

unlikely that two nodes share the same degree combination. Therefore, the average number of nodes with a given degree tends to be very low, $\langle N(\boldsymbol{k})\rangle \to 1$. Under such a situation, the average number of links between nodes with $\boldsymbol{k}$ and nodes with $\boldsymbol{q}$ is also very small and $\langle L(\boldsymbol{k} \to \boldsymbol{q})\rangle \to 1$. In the limit, when all nodes have unique degrees, all links are necessarily deterministic. On the other hand, if the degree distribution is narrow, $|k_{\rm max} - k_{\rm min}| \ll N-1$, the space of possible (k_i, k_o) combinations is much smaller and there is a higher chance that two or more nodes have the same degrees. Then, the number of possible $\boldsymbol{k} \to \boldsymbol{q}$ links $N(\boldsymbol{k})N(\boldsymbol{q})$ grows significantly and there is a lower chance for deterministic links to occur.

To test this argument, the variance of the input degree has been calculated for all the real networks. Its relationship to $L_{\rm det}$ is presented in figure 3(*b*). The variance has been normalized by the network size N for comparative reasons. A clear trend exists that, the larger the variance, the larger is the embedded deterministic network. This indicates the validity of the argument above, however, the relation is neither perfect nor linear and further factors do exist.

In figures 3(*c*) and (*d*) the *cumulative* input degree distributions of three real networks are shown. The output degree distributions display similar behaviour in all cases. The world-trade-web of the year 2000 possesses an extremely wide input degree distribution, some nodes having degrees up to $k_i = 183$ ($N = 190$). The almost constant decay denotes a very uniform distribution. These two factors imply that $\langle N(\boldsymbol{k})\rangle \to 1$, resulting in a very large number of deterministic links, indeed $L_{\rm det}/L \approx 0.95$. The cortical network of the cat has also a very broad degree distribution with nodes receiving connections from up to 66% of other nodes ($k_{i,\rm max} = 35$, $N = 53$). However, its distribution decays much faster than that of the year 2000 world-trade-web. Indeed, it exhibits a long tail similar to that of scale-free-like networks. The range $k_i/N > 0.4$ in figure 3(*c*) shows that only very few of all the nodes have high degree. This condensation of nodes towards low degrees results in an increase of $\langle N(\boldsymbol{k})\rangle$ and thus, a lower chance for deterministic links.

The Spanish Wikipedia possesses a scale-free-like degree distribution with an exponent $\gamma \approx 2.2$, meaning that there is an extreme condensation of nodes towards low degrees, figure 3(*d*). Up to 98% of the nodes have $k_i < 100$ and only the rest 2% of the nodes have larger degrees. Besides, as $N = \sum_{k_i} N(k_i)$, the maximal degree is necessarily very low ($k_{\max} \ll N$). The node with the largest input degree has $k_i = 2987$, that is only 10% of the maximum possible ($N = 29\,196$). These observations indicate that $\langle N(\boldsymbol{k})\rangle \gg 1$ and then conditions (4) and (5) rarely hold, illustrating why the Wikipedias are the networks with smallest fraction of L_{det} from all networks studied here. Anyway, where do the 20% of deterministic links in the Spanish Wikipedia come from? For degrees $k_i > 100$ and $k_o > 100$, $N(\boldsymbol{k})$ is very often 1 or 2 giving rise to deterministic links. From all the deterministic links, 65% of them correspond to links $\boldsymbol{k} \to \boldsymbol{q}$ where all $k_i, k_o, q_i, q_o > 100$. Contrary to what happens in the world-trade-webs, the deterministic links in the Wikipedias arise mainly from the long tail due to the limited size effects of the scale-free distribution.

3.2. Computational implications

So far, the observations above demonstrate that a large number of links are deterministic once all the 1-node and 2-node correlations have been assigned. The random generation process can safely ignore them and it only needs to introduce the remaining $L_{\text{free}} = L - L_{\text{det}}$ links. In order to generate an ensemble of random networks, if each realization is started from the deterministic network, there is a very relevant reduction of computational time. For the real networks analysed, the worst case corresponds to the Wikipedias whose 20% of links are deterministic. Even in this case, the computational benefit is very important.

4. Avoiding self-loops and multiple-edges

For different reasons, self-loops and multiple-links are often not desired. If the method in section 2 is directly modified for that purpose, the process will most likely end up blocked in a state where the only possibility to continue is to either introduce a self-loop or a multiple-link. Depending on the specific prescribed properties and the past history of the generation process, there exist many possible blocked states. Let us illustrate only a few of them.

Consider the situation in figure 4(*a*). Imagine that the process runs for several iterations without problems. At some point the node s_3 is chosen as source, that has been assigned final degrees $\boldsymbol{k}'$. For all nodes $s \in \mathcal{N}(\boldsymbol{k}')$, $fk_o(s) = fk_i(s) = 0$ except for s_3. Imagine also that from all possible $\boldsymbol{k}' \to \boldsymbol{q}$ links, the only remaining link to be introduced is of the type $\boldsymbol{q} = \boldsymbol{k}'$:

$$fL(\boldsymbol{k}' \to \boldsymbol{q}) = \begin{cases} 1 & \text{if} \quad \boldsymbol{q} = \boldsymbol{k}' \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

Under these circumstances, the only possible option for the random process to continue is to introduce the self-loop $s_3 \to s_3$.

Another possible blocked situation is depicted in figure 4(*b*). Assume that during the random process the links $s_1 \to t_1$, $s_3 \to t_2$ and $s_3 \to t_3$ are introduced. Some iterations later s_3 is again selected. Then we have the following scenario: the node s_2 has done all its possible links, $fk_o(s_2) = 0$. There are only three remaining links to be done of the type $\boldsymbol{k}' \to \boldsymbol{q}$:

$$fL(\boldsymbol{k}' \to \boldsymbol{q}) = \begin{cases} 2 & \text{if} \quad \boldsymbol{q} = \boldsymbol{q}_1 \\ 1 & \text{if} \quad \boldsymbol{q} = \boldsymbol{q}_2 \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

Nodes t_1, t_4 and t_5 cannot accept more incoming connections $fk_i(t_1) = fk_i(t_4) = fk_i(t_5) = 0$. Nodes t_2 and t_3 have $fk_i > 0$ and are therefore still free to accept incomming connections,

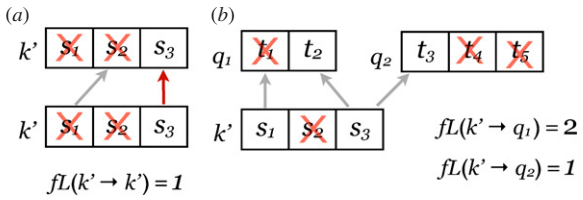

Figure 4. Blocked states of the random generation process. (*a*) The only remaining choice to connect s_3 is to introduce a self-loop $s_3 \to s_3$. (*b*) s_3 can only connect again introducing a multiple-link $s_3 \to t_2$ or $s_3 \to t_3$.

but both $s_3 \to t_2$ and $s_3 \to t_3$ already exist. The only possibility to connect s_3 again is to introduce a multiple-link with t_2 or to t_3. Hence, the process is blocked.

So, what to do whenever the process closes itself into a state where no more configurations are possible? We need to 're-open' in a random manner the space of possible configurations to which it can move ahead. The process will then continue to accomplish its purpose: the generation of a network that belongs to the space of all possible maximally random networks with the desired properties. Furthermore, all members of that space should come out with equal probability. We can image two different strategies to let the process escape from blocked configurations.

4.1. The 'fair-trade' method

When a node s' has no choices to form new connections other than make a self-loop or a multiple-link, it could try 'trading' a new link with any of its companions in $\mathcal{N}(\boldsymbol{k}')$. We look for another $s'' \in \mathcal{N}(\boldsymbol{k}')$ that has no connections yet to at least one of the free remaining target nodes, $\{t_{\text{free}}\}$. Then, one of those targets is chosen at random and the link $s'' \to t_{\text{free}}$ is created. Usually s'' ends up with more links than allowed, $fk_o(s'') = -1$. In exchange, s'' gives one of its own links to s'.

$$s'' \to t_{\text{free}}, \qquad s'' \nrightarrow t', \quad s' \to t'.$$

The trade done by s' consists in taking one of the links from s'' in order to escape the blocked state, and in exchange, we look for a new link for s''. Note that this is not a switching step because one of the links, $s'' \to t_{\text{free}}$, does not previously exist.

Let us illustrate this with the example in figure 4(*b*). Node s_3 is blocked. It is already connected to t_2 and t_3, the only remaining nodes in $\mathcal{N}(\boldsymbol{q}_1)$ and $\mathcal{N}(\boldsymbol{q}_2)$ that are free to receive connections. However, nodes s_1 and s_2 are not connected to any of them. One of the $\{s_1, s_2\}$ nodes is taken at random, say s_2. One of the free targets $\{t_2, t_3\}$ is also chosen at random, say t_3, and the link $s_2 \to t_3$ is created. Now, as s_2 has more connections than it should, $fk_o(s_2) = -1$, it will give one of its links to s_3. From the set of nodes $s_2 \to \{t\}$, a t' is selected at random for which $s_3 \to t'$ does not previously exist, otherwise a multiple-link would be introduced. The link $s_2 \nrightarrow t'$ is removed and $s_3 \to t'$ created.

If necessary, a 'fair-trade' can also happen among the target nodes. If for example $N(\boldsymbol{k}') = 1$, the only node $s' \in \mathcal{N}(\boldsymbol{k}')$ has no companions. Then, the target nodes could exchange their sources in an equivalent manner. We have largely explored this sort of strategies and constructed very satisfactory algorithms that would generate the desired random networks in only $L_{\text{free}} = L - L_{\text{det}}$ iteration steps (starting from the deterministic network). However, the outcome of a successful realization is not always guaranteed. The blocked situations shown in figure 4 are only two examples. In reality there exist a myriad of possible blocked states and,

unfortunately, finding a trade is not always guaranteed. In the following section, we present an alternative strategy that works in all situations.

4.2. The 'force-and-drop' method

Let us consider again the blocked situation in figure 4(*b*) with node s_3 as the chosen source. From the remaining free target degrees $\{q_1, q_2\}$ one is chosen at random, say q_1. There exist $N(\boldsymbol{k}')N(\boldsymbol{q}_1) = 6$ possible links of the type $\boldsymbol{k}' \to \boldsymbol{q}_1$. Links $s_1 \to t_1$ and $s_3 \to t_2$ are already present in the random network. From the remaining four possible links: $\{(s_1 \to t_2), (s_2 \to t_1), (s_2 \to t_2)$ and $(s_3 \to t_1)\}$ one is chosen at random, e.g. $s_2 \to t_2$. *A priori*, this link is not possible because $fk_o(s_2) = 0$. However, the link is 'forced' into the network. Now, s_2 has more outgoing links than it should, so one of its old links is 'dropped' at random. The removal of this link might open new accessible configurations for the random process. This is not always true, but trying enough times, the generation process will always find a path to finish.

Due to the randomness of the method, it is not possible to know with certainty the number of iterations necessary. Some dropped links might not open the space of possible configurations while others will. When a link $s' \to t'$ is forced, the current iteration might have a gain of +1 if both s' and t' finish with degrees $fk_o(s') \geqslant 0$ and $fk_i(t') \geqslant 0$. If after the forcing either $fk_o(s') = -1$ or $fk_i(t') = -1$, then the iteration has null gain, i.e., one link is introduced and another one is dropped. If both s' and t' end up with $fk_o(s') = fk_i(t') = -1$, then the iteration has a gain of -1 because one link is introduced but two are dropped. Let us now adequately describe the complete algorithm.

5. Algorithm for the generation of random *directed* networks with prescribed $1n2n$ correlations, without self-loops or multiple-links

When the introduction of self-loops or multiple-links is not a problem, the algorithm described in section 2 is the algorithm of choice. If several realizations are wanted, only remember to calculate first the deterministic network (appendix C), update the fk_o, fk_i and $fL(\boldsymbol{k} \to \boldsymbol{q})$ quantities, and proceed to introduce the remaining $L_{\text{free}} = L - L_{\text{det}}$ links. In order to avoid self-loops and multiple-links, the same method is applied, only that every time the process gets blocked then the 'force-and-drop' method is activated. This permits the process to continue and finish all realizations satisfactorily. Additionally, the structures *fqlist* and $f\mathcal{N}(k)$ can be implemented to speed up the generation process (section 2.2). For clarity reasons, we will omit them in the description [10].

Once all necessary data structures are ready the deterministic network is taken as the starting point. The remaining number of $L_{\text{free}} = L - L_{\text{det}}$ links are introduced in the following manner:

(1) From all the nodes with $fk_o(s) > 0$ one is chosen at random to act as the source, s'. As degrees have been previously assigned, we know that s' needs to have final degrees $\boldsymbol{k}'$.
(2) A list is constructed, *tlist*, that contains all potential target nodes for node s'. This includes all nodes t with $fk_i(t) > 0$ that belong to any of the degree classes $\mathcal{N}(\boldsymbol{q})$ for which $fL(\boldsymbol{k}' \to \boldsymbol{q}) > 0$. Importantly, as no self-loops nor multiple-links are desired, *tlist* cannot include s' itself nor any of the nodes for which a link $s' \to t$ already exist.

When the process arrives at a blocked state, *tlist* remains empty. This is the crucial sign that allows us to decide whether the process can normally continue or, otherwise, the 'force-and-drop' sequence is to be activated.

(3a) `if tlist not empty:` a target node t' is chosen at random from *tlist* and the connection $s' \to t'$ is done. The data structures are updated: $fk_i(t') -= 1$, $fk_o(s') -= 1$ and $fL(\boldsymbol{k}' \to \boldsymbol{q}') -= 1$, where $\boldsymbol{q}'$ are the degrees of t'.
The iteration has satisfactorily finished. `Return to step (1).`

(3b) `if tlist is empty:` the 'force-and-drop' method needs to be activated:

(3b.1) From all degrees $\boldsymbol{q}$ for which $fL(\boldsymbol{k}' \to \boldsymbol{q}) > 0$, one is chosen at random, $\boldsymbol{q}'$.

(3b.2) A list is constructed, $posslinks$, that contains all possible links from nodes $s \in \mathcal{N}(\boldsymbol{k}')$ to nodes $t \in \mathcal{N}(\boldsymbol{q}')$. In principle there exist $N(\boldsymbol{k}')N(\boldsymbol{q}')$ such connections, but links that already exist in the random network are ignored in order to avoid multiple-links. If $\boldsymbol{k}' = \boldsymbol{q}'$, then potential self-loops $s \to s$ are neither included into $posslinks$.

(3b.3) *Force a link.* One link in $posslinks$ is chosen at random $s'' \to t''$ and introduced. Update the data structures $fk_i(t'') -= 1$, $fk_o(s'') -= 1$ and $fL(\boldsymbol{k}' \to \boldsymbol{q}') -= 1$.

(3b.4) *Drop links* (only if necessary):

- `if` $fk_i(t'') = -1$: from all the incoming links $(\{s\} \to t'')$ choose one at random and drop it, $s_d \nrightarrow t''$. Update $fk_o(s_d)$+=1 and $fk_i(t'') = 0$.
- `if` $fk_o(s'') = -1$: from all the outgoing links $(s'' \to \{t\})$ choose one at random and drop it, $s'' \nrightarrow t_d$. Update $fk_o(s'') = 0$ and $fk_i(t_d)$+=1.

The iteration is finished. `Return to step (1).`

5.1. Performance of the algorithm

Estimating the precise complexity of this algorithm is very difficult. Its performance depends both on the correlation structure and on the stochastic nature of the 'force-and-drop' method. As discussed in section 2.2, preparation of the data structures requires a computational cost of the order $\mathcal{O}(N + L)$. Creating the deterministic network includes an additional cost of $\mathcal{O}(L)$. However, all these structures need to be created only once.

If self-loops and multiple-links are allowed, starting from the deterministic network, our algorithm generates a maximally random network of size N and L links in $L_{\text{free}} = L - L_{\text{det}}$ iterations, less than L itself! As shown in section 3, L_{free} depends nontrivially on the correlation structure, therefore, an accurate estimation of the complexity is not possible. Each iteration consists of three steps whose complexity is neither possible to be estimated for the same reason. Nevertheless, in section 2.2 we explained how to largely reduce the computational cost of each iteration by the introduction of look-up tables. Thanks to the presence of the deterministic links, the temporal requirements to generate an ensemble of m random networks of size N, L links and prescribed $1n2n$ degree correlations scales as $\mathcal{O}(mL_{\text{free}})$, not as $\mathcal{O}(mL)$. According to our observations in section 3, L_{free} is typically much smaller than L what results in a great computational benefit for the generation of the random network ensembles.

When self-loops and multiple-links are to be avoided, it is also very difficult to estimate how often the 'force-and-drop' sequence needs to be activated. Furthermore, a 'dropped' link will not always re-open the space of reachable configurations permitting the random process to advance. In order to explore the influence of the stochastic part in the performance of the algorithm, we have generated ensembles of 100 random realizations using the correlation structure of each of the real networks in table 1. The exception are the Wikipedias for which 30 realizations have been generated. The number of iterations needed to accomplish each realization has been counted. In general, a realization would rarely finish in only L_{free} iterations, i.e. the process finished without using the 'force-and-drop' sequence. For some network types, none of the realizations did. However, as shown in figure 5(*a*) the average number of iterations required to finish all realizations is only slightly larger than L_{free}, meaning that, indeed, the 'force-and-drop' sequence is activated in only few occasions.

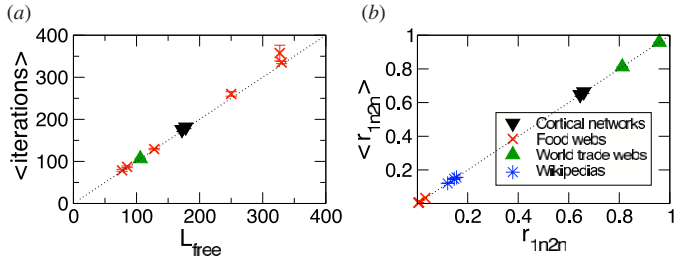

Figure 5. Performance of the algorithm. (*a*) Average number of iterations necessary to accomplish a random realization. Averages of 100 realizations. (*b*) Expected reciprocity of networks due to all $1n2n$ degree correlations. Theoretical expectation r_{1n2n} and average of the measured reciprocity in the random networks $\langle r_{1n2n}\rangle$. Error bars are very small in both figures.

5.2. Testing the statistical properties

In our recent paper [4], we have presented theoretical results for the expected reciprocity of directed networks under different conditions of degree correlations. Our most general result predicts that the expected reciprocity in a maximally random network with all $1n2n$ correlations prescribed is

$$r_{1n2n} = \frac{1}{L}\sum_{\boldsymbol{k},\boldsymbol{q}} \frac{L(\boldsymbol{k}\to\boldsymbol{q})L(\boldsymbol{k}\leftarrow\boldsymbol{q})}{N(\boldsymbol{k})N(\boldsymbol{q})}. \tag{8}$$

The ensemble average reciprocities $\langle r_{1n2n}\rangle$ of the generated networks has been calculated and compared to the theoretical expectations r_{1n2n}. The excellent agreement between theory and the experimental observation reflected in figure 5(*b*) emphasizes the validity of the algorithm proposed here.

6. Conclusions and outlook

In this paper, we have presented a method to generate maximally random *directed* networks with prescribed 1-node and 2-node degree correlations. For this purpose we have analysed the correlation nature of real networks and found that a large number of links are not subject to be randomized. Links that obey the conditions (4) and (5) are referred as *deterministic links*. The precise number of such links, $L_{\rm det}$, depends on several factors and it is difficult to estimate. It has been argued that in a network of N nodes, the wider the range of input and output degrees ($|k_{\rm max} - k_{\rm min}| \to N$), the higher is the chance for deterministic links to happen. Several observations indicate the validity of this hypothesis, although further work is necessary.

The embedded *determinist network* has been defined as the subnetwork containing uniquely deterministic links. Taking it as the starting point of the random network generation process, maximally random networks of size N and L links with desired 1-node and 2-node correlations are obtained in a number of iterations $L_{\rm free} = L - L_{\rm det}$, what is typically much smaller than L. This results in a large computational time benefit when generating ensembles of random networks. If self-loops and multiple-links are allowed, the method described in section 2 is sufficient.

If self-loops and multiple-links are not desired, random network generation processes are known to fall into blocked states. This happens when the only chance for the process to continue is either to introduce a self-loop or a multiple-link. The widely known *configuration model* [11, 12] describes a method to obtain random networks with desired degree sequences.

If the process reaches a blocked state, it is typically recommended to discard the entire realization and start a new one [12, 13]. In this paper, we have discussed different techniques to overcome blocked states and permit all realizations to return a valid random network. Due to its stability, the 'force-and-drop' is proposed as the preferred mechanism. Finally, a complete algorithm has been described in section 5.

The algorithms proposed in this paper can be very easily simplified for the case of undirected graphs. They are also very easily reduced to generate maximally random networks with any of the desired combination of 2-node 2-point correlations depicted in figure 1. As shown in [4], the distinct classes of degree correlations influence the outcome of graph measures at different extent. Therefore, the methods presented here are useful tools for the significance testing of network measures. Certainly, theoretical developments will also benefit the capacity to generate random networks with the degree correlations of choice.

Acknowledgments

This work has been supported by the Helmholtz Institute for Supercomputational Physics (GZ-L), by GOFORSYS (BMBF) (CZ and JK) and by the Croatian Ministry of Science, Education and Sports under project number 098-0352828-2863 (VZ)

Appendix A. Data structures to handle the correlations

The two-dimensional degree distributions could be stored into a matrix whose elements (k_i, k_o) contain the value $N(\boldsymbol{k})$. However, this matrix will be very sparse and a dictionary-like (hash table) structure is recommended:

$$Nk = \{(k_i, k_o) : N(k_i, k_o)\},$$

where for each (k_i, k_o) element, $\mathtt{Nk}(k_i, k_o)$ stores the scalar value $N(\boldsymbol{k})$.
Given a real network G, $N(\boldsymbol{k})$ can be extracted with complexity $\mathcal{O}(N)$:

```
Nk = empty
for s in G:
    k = (k_i(s), k_o(s))
    if k in Nk:  then Nk(k) = Nk(k) + 1.
    else:  Nk(k) = 1.
```

The 2-node degree correlations form a four-dimensional space. They can also be easily stored in a dictionary-like structure. $Lkq = \{(\boldsymbol{k}, \boldsymbol{q}) : L(\boldsymbol{k} \to \boldsymbol{q})\}$, where for each combination of degrees $(\boldsymbol{k}, \boldsymbol{q}) = ((k_i, k_o), (q_i, q_o))$, the structure $\mathtt{Lkq}(\boldsymbol{k}, \boldsymbol{q})$ contains the scalar value $L(\boldsymbol{k} \to \boldsymbol{q})$.
Being G the adjacency list of a real network where $G(s)$ contains all target nodes t' for which a link $s \to t'$ exists, $\mathtt{Lkq}(\boldsymbol{k}, \boldsymbol{q})$ can be extracted in $\mathcal{O}(L)$:

```
Lkq = empty
for s in G:
    k = (k_i(s), k_o(s))
    for t' in G(s):
        q = (k_i(t'), k_o(t'))
        kq = (k, q)
```

```
if kq in Lkq:  then Lkq(kq) = Lkq(kq) + 1
else:  Lkq(kq) = 1.
```

Appendix B. Look-up tables to help improve performance

Being G the adjacency list of the real network, both look-up tables $f\mathcal{N}(\boldsymbol{k})$ and $fqlist(\boldsymbol{k})$ (see section 2.2) can be simultaneously created as dictionary-like structures in complexity $\mathcal{O}(L)$:

```
fN = empty
fqlist = empty
for s in G:
    k = (k_i(s), k_o(s))
    if k in fN: then include s in fN(k)
    else:  fN(k) = {s}
    for t' in G(s):
        q = (k_i(t'), k_o(t'))
        if q not in fqlist(k):  then include q in fqlist(k).
```

Appendix C. Extracting the deterministic network

Once Nk and Lkq have been calculated, the deterministic network $G^{\det}$ can be extracted with complexity $\mathcal{O}(L)$ from a real network G in the following manner:

```
G^det = empty
for s in G:
    k = (k_i(s), k_o(s))
    for t' in G(s):
        q = (k_i(t'), k_o(t'))
        kq = (k, q)
        if k == q and Lkq(kq) == Nk(k) * (Nk(k) - 1):
            include (s → t') in G^det.
        if k ≠ q and Lkq(kq) == Nk(k) * Nk(q):
            include (s → t') in G^det.
```

References

[1] Solomonoff R and Rapoport A 1951 *Bull. Math. Biophys.* **13** 107–117
[2] Erdös P and Rényi A 1959 *Publ. Math. Debrecen* **6** 290–297
[3] Newman M E J 2003 *SIAM rev.* **45** 167
[4] Zamora-López G, Zlatic V and Zhou C 2008 Reciprocity of networks with prescribed degree correlations and arbitrary degree sequences *Phys. Rev.* E at press (*Preprint* arXiv:0706.3372)
[5] Scannel J W *et al* 1999 *Cereb. Cortex* **9** 277
[6] Sporns O and Zwi J D 2004 *Neuroinf.* **2** 145
[7] http://www.cosin.org/extra/data/foodwebs/
[8] Gleditsch K S 2002 *J. Confl. Resolution* **46** 712

[9] Zlatić V *et al* 2006 *Phys. Rev.* E **74** 016115
[10] A complete and commented code, written in Python language, is available at http://www.agnld.uni-potsdam.de/~gorka
[11] Molloy M and Reed B 1995 *Random Struct. Algorithms* **6** 161–79
[12] Newman M E J, Strogatz S H and Watts D J 2001 *Phys. Rev.* E **64** 026118
[13] Milo R and Kashtan N 2004 *Preprint* cond-mat/0312028 v2