MethodsX 5 (2018) 773–780

Contents lists available at ScienceDirect

MethodsX

journal homepage: www.elsevier.com/locate/mex

Method Article

A unifying framework for fast randomization of ecological networks with fixed (node) degrees

Corrie Jacobien Carstens[a], Annabell Berger[b], Giovanni Strona[c,*]

[a] University of Amsterdam, KdV Institute for Mathematics, Amsterdam, Netherlands
[b] Martin Luther University Halle-Wittenberg, Institute of Computer Science, Halle(Saale), Germany
[c] European Commission Joint Research Centre, Directorate D – Sustainable Resources, Bio-Economy Unit, Ispra (VA), Italy

A B S T R A C T

The Curveball algorithm is an efficient and unbiased procedure for randomizing bipartite networks (or their matrix counterpart) while preserving node degrees. Here we introduce two extensions of the procedure, making it capable to randomize also unimode directed and undirected networks. We provide formal mathematical proofs that the two extensions, as the original Curveball, are fast and unbiased (i.e. they sample uniformly from the universe of possible network configurations).

- We extend the Curveball algorithm to unimode directed and undirected networks.
- As the original Curveball, extensions are fast and unbiased.
- We provide Python and R code implementing the new procedures.

A R T I C L E I N F O
Method name: Curveball algorithms to randomize networks with fixed node degrees
Keywords: Binary matrices, Co-occurrence, Curveball algorithm, Food web, Null model
Article history: Received 2 May 2018; Accepted 25 June 2018; Available online 5 July 2018

Specification table

Subject Area	*Agricultural and Biological Sciences*
More specific subject area:	*Network science*
Method name:	*Curveball algorithms to randomize networks with fixed node degrees*

* Corresponding author.
E-mail address: giovanni.strona@ec.europa.eu (G. Strona).

https://doi.org/10.1016/j.mex.2018.06.018

Name and reference of original method	*Curveball algorithm: G. Strona, D. Nappo, F. Boccacci, S. Fattorini, J. San-Miguel-Ayanz. A fast and unbiased procedure to randomize ecological binary matrices with fixed row and column totals. Nat. Commun. 5 (2014) 4114.*
Resource availability	*https://github.com/queenBNE/Curveball*

Methods details

Background

A good algorithm to generate random networks with prescribed degree distribution (which is identical to the issue of generating random binary matrices with fixed marginal totals) should have two properties: it should be able to generate any one among all possible networks having a certain node degrees with the same probability, i.e. it should not tend towards the generation of networks having particular structural properties; and it should be able to generate truly random networks fast.

Markov chains, where the randomization takes place in subsequent steps, each involving a small change in the network structure, represent a common solution to this problem. Several network randomization Markov chains have been shown to converge to the uniform distribution on their state space, that is they have been shown able to generate truly random networks with prescribed degree distribution [1–5]. By contrast, most of Markov chains exhibit an important limit, that is it is not clear how many randomization steps they require to ensure that the randomized network is truly random.

The best known Markov chain approach for randomizing network while preserving their degree sequence is the *switching model* (also known as rewiring, switching chain and swapping edges) [6,7,2,8]. It can be applied to different kinds of networks, being able to properly randomize bipartite networks, undirected networks or directed networks with given node degrees, by repeatedly switching the ends of non-adjacent edge pairs (with some additional rules required for the correct randomization of directed networks, [2]). Yet, this method has a fundamental drawback, which is requiring a very large number of switches in order to ensure an unbiased randomization, which grows very rapidly with the size of the network (see, for example, [9]).

A more recent Markov chain approach is the Curveball algorithm [10]. Experimentally, this chain has been shown to mix much faster than the corresponding switching chain [10]. Why the Curveball algorithm mixes faster than the switching model can be understood when thinking of both algorithms as games in which kids trade cards. That is, think of the Curveball algorithm as an algorithm that randomises the binary $n \times m$ bi-adjacency matrix of a bipartite network. Imagine that each row of the adjacency matrix corresponds to a kid, and the 1's in each row correspond to the cards owned by the kid. Then at each step in the Curveball algorithm, two kids are randomly selected, and trade a number of their differing cards. Using this same analogy for the switching model, in each step two cards are randomly selected and traded if firstly they are different and secondly they are owned by different kids (note that various algorithms implementing similar approaches were discovered independently by Verhelst [4]). Intuitively, the Curveball algorithm is clearly a more efficient approach to randomise the card ownership by the kids. More formally, the Curveball algorithm is also based on switches but instead of making one switch, several switches can be made in a single step, which leads to possibly exponentially many networks being reached in a single step, in contrast with the switching model where at most n^4 (the maximum number of possible edge pairs) networks can be reached in a single step.

Designed to randomize only bipartite networks, the Curveball permits the randomization of both species $\times$ locality matrices, and bipartite ecological networks such as host-parasite and plant-pollinator ones. There is, however, an important reason why such design requires an urgent upgrade. Notably, bipartite ecological networks have often been studied separately from food webs, even though all those networks belong to the same broader ecological class of 'resource-consumer' networks [11].

Food webs belong to a different class of networks, that is directed networks. In such networks, nodes cannot be attributed unambiguously to two different classes, since the same node can be

simultaneously a consumer and a resource (for example, a predator can be eaten by another predator of a higher trophic level) [12]. A third class of networks is that of undirected networks, which has importance in various fields, such as social sciences and epidemiology, with networks of those kinds being well suited, to represent contacts between persons, and that is also becoming increasingly relevant in the ecological context. In fact, there is a growing interest in the study of co-occurrence networks, that is networks obtained by linking species found together more often than random expectation, and hence considered as potentially interacting (see, for example, [13–15]).

Although some attempts has been recently made to provide measures of network structure applicable to different kinds of ecological networks (see, for example, [16,17]), we are still very far from having a unifying analytical toolbox. Here we take a step further in this direction, by showing how the efficient Curveball algorithm can be extended to work also with unimode directed networks, and undirected networks. Besides providing ecologists with a common procedure to analyze different ecological entities, this constitutes an important advance for network science in general, with the potential of bringing benefit to various disciplines.

Extending the Curveball algorithm

We now propose two extensions of the Curveball algorithm: the Directed Curveball algorithm, which samples directed networks and the Undirected Curveball algorithm, which samples networks (Throughout this paper we use the convention that both directed and undirected networks do not contain self-loops or multiple edges). Both are Markov chain algorithms, which randomise networks by repeatedly applying them small changes, in this case trades of elements.

The Directed Curveball algorithm

An efficient way to store a directed network $G = (V, E)$ is by storing its adjacency list, which is also the most natural data structure to run the Directed Curveball algorithm on. The adjacency list of G is a list of sets A_v, one for each node $v \in V$, where the set A_v contains all out-neighbours of v.

The Directed Curveball algorithm randomises a directed network by repeatedly applying *trades* to its adjacency list. A *trade* is defined as follows: (a) select two sets A_i and A_j at random, (b) Let A_{i-j} be all nodes in A_i that are not in A_j and that are *not equal to* j, i.e. $A_{i-j} = A_i \setminus (A_j \cup \{j\})$ and similarly let $A_{j-i} = A_j \setminus (A_i \cup \{i\})$. (c) Create new sets B_i by removing A_{i-j} from A_i and adding the same number of elements randomly chosen from $A_{i-j} \cup A_{j-i}$. Combine A_j A_{j-i} with the remaining elements of $A_{i-j} \cup A_{j-i}$ to form B_j.

Notice that the definition of $A_{i-j} = A_i \setminus (A_j \cup \{j\})$ in step (b) ensures no self-loop can be created at node j, since this would require adding j to A_{j-i}.

Fig. 1 illustrates a trade of the Directed Curveball algorithm. We will refer to the number of elements exchanged as the *size of a trade*, for instance the trade in Fig. 1 is of size two. It is possible for a trade to be of size zero, in this case the current network is repeated and we move on to the next trade. The Lemma below shows that all switches in the switching model for directed equal trades of size one in the Directed Curveball algorithm. But, the Directed Curveball algorithm in addition allows trades of larger size. Intuitively, this could reduce the number of steps needed to obtain a random sample as compared to switching chains, since a trade can introduce more randomness than a switch.

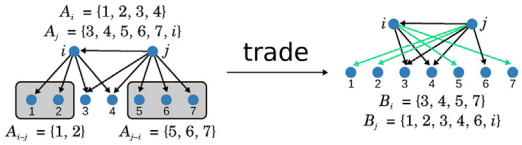

Fig. 1. Illustration of a trade in the Directed Curveball algorithm. Vertex i and j have several common out-neighbours and i is an out-neighbour of j. Those are removed to obtain the set {1, 2, 5, 6, 7} of nodes that can be traded. The nodes 5 and 7 are selected as new neighbours for i, the trade results in the network on the right.

Lemma 1. *Any switch in a directed network is a trade of size one of the Directed Curveball algorithm.*

Proof. Let (x, y) and (u, v) be edges in a directed network G that are allowed to be switched. Then $x \neq v$ and $u \neq y$ since otherwise this switch would introduce a self-loop. Furthermore $v \notin A_x$ and $y \notin A_u$ since otherwise the resulting directed network would have multiple edges. In particular this implies $y \in A_{x-u}$ and $v \in A_{u-x}$. Now if row x and row u are selected for a trade, then $B_x = (A_x \{y\}) \cup \{v\}$ and $B_u = (A_u \{v\}) \cup y$ are possible sets in step (c) that lead to exactly the two new edges (x, v) and (u, y).

In the Appendix we show that the Directed Curveball algorithm converges to the uniform distribution whenever the switching chain for directed networks does. That is, eventually after a large number of trades, we obtain a network sampled from the uniform distribution. Note that, for certain node degrees, not all networks realisations (i.e. networks with the given node degrees -) can be obtained by applying switches to a given network.

The simplest example is the oriented triangle (1, 2), (2, 3), (3, 1). No swap or trade can be applied, and hence we can never generate the triangle (1, 3), (3, 2), (2, 1), which is also a possible network. Adding a second procedure, i.e. reorienting a randomly chosen directed triangles ('hexagonal move') solves this problem completely [2]. However, further work showed that not all of these triangles need to be re-oriented [18]. Theorem B3 in Appendix can be used to create a fast algorithm to recognize the triangles which need to be re-oriented in a network, and to choose one random orientation for each of them. Using the Curveball chain after this step delivers a uniform randomly sampled network. Furthermore, depending on the purpose of the randomization, this step might be even not necessary.

Berger et al. [18] proved that (in cases where only network topology matters, i.e. where the information about the identity of individual nodes can be discarded) the sole use of the switch chain permits to sample uniformly at random from the set of all possible networks (which is a subset of the full set of networks including those that could be generated by triangle re-orientation). This would be enough to compare the frequency of network structural patterns such as motifs, nestedness, or C-score between empirical and randomized networks, since their proportion is not affected by the re-orientation step. Conversely, the re-orientation step discussed in Appendix would be necessary to assess the significance of patterns affected by the identity of nodes (for example, the frequency of a specific directed edge (i, j) in all networks).

The Undirected Curveball algorithm

The Undirected Curveball algorithm samples networks with fixed node degrees. The adjacency list representation of a network $G = (V, E)$ is a list of sets A_i. The set A_i contains the indices of the neighbours of vertex i. For undirected networks each edge $\{i, j\}$ is represented twice in the adjacency list, since $i \in A_j$ and $j \in A_i$. Furthermore, $i \notin A_i$ for all i, since networks do not contain self-loops.

A trade in the Undirected Curveball algorithm is defined by the following steps. (a) Randomly select two sets A_i and A_j. (b) Let A_{i-j} be the set of elements in A_i not in A_j and not equal to j, i.e. $A_{i-j} := A_i \ (A_j \cup \{j\})$. Analogously define $A_{j-i} := A_j \ (A_i \cup \{i\})$. (c) Create a new set B_i by removing A_{i-j} from A_i and adding the same number of elements randomly chosen from $A_{i-j} \cup A_{j-i}$. Combine $A_j \ A_{j-i}$ with the remaining elements of $A_{i-j} \cup A_{j-i}$ to form B_j. (d) For each node $k \in B_i \ A_i$, replace j by i in B_k, similarly for each $l \in B_j \ A_j$, replace i by j in B_l.

Step (b) ensures no self-loops are introduced and step (d) ensures that B represents a network (i.e. that $i \in B_j$ implies $j \in A_i$). Fig. 2 illustrates a trade in the Undirected Curveball algorithm. Note that the same trade can be made in the opposite direction, i.e. B network can be transformed into A in a single trade. A proof for this (which is a necessary condition for an unbiased Markov chain) is provided in Appendix.

The Undirected Curveball algorithm includes trades of size zero which correspond to repeating the current network. Furthermore, Lemma 2 shows that any switch in the switching model for corresponds to a trade of size one in the Undirected Curveball algorithm. In fact, Fig. 3 shows that for each switch in the switching model, there are two different trades of size one in the Undirected Curveball algorithm.

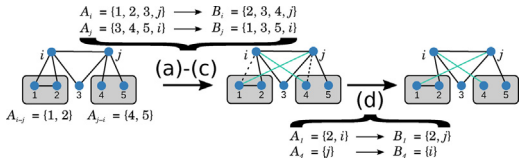

Fig. 2. Illustration of a trade in the Undirected Curveball algorithm. Nodes i and j are neighbours and have a single common neighbour. The nodes that are available for trading are nodes 1, 2, 4 and 5. A trade of size one is performed, exchanging nodes 1 and 4, hence in step (d) the sets A_1 and A_4 are updated.

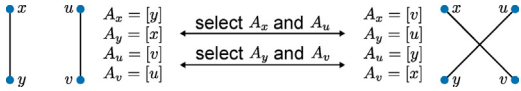

Fig. 3. A switch corresponds to a trade of size one in the Undirected Curveball algorithm. Notice that each switch can be realized by two distinct trades: the switch from $\{x, y\}$ and $\{u, v\}$ to $\{x, v\}$ and $\{u, y\}$ can be realized by selecting A_x and A_u and trading y for v *or* by selecting A_y and A_v and trading x for u.

Lemma 2. *Let G, G' be that differ by a switch. There are two trades of size one in the Undirected Curveball algorithm from G to G'.*

Proof. Without loss of generality we may assume that $G = (V, E)$ and $G' = (V, E')$ differ by a switch from $\{x, y\}$ and $\{u, v\}$ to $\{x, v\}$ and $\{u, y\}$. Let $\{A_1, \ldots, A_n\}$ be the adjacency set representation of G, then $y \in A_{x-u}$ since the edge $\{x, y\}$ is an edge of G, the edge $\{u, y\}$ is not and y can not be equal to u since $\{u, y\} \in E'$ and G' has no self-loops. Similarly we find that $v \in A_{u-x}$ and hence the trade that swaps y and v between rows x and u results in the network G'. Similarly, there is a second trade which generates G', namely the trade that exchanges x and u between sets A_y and A_v. □

Analogous to the other versions of the Curveball algorithm, the Undirected Curveball algorithm in addition allows trades of larger size, corresponding to making several switches at once. In the Appendix we prove that the Undirected Curveball algorithm samples uniformly at random after applying a large number of trades.

Mixing time and experimental stopping times

There are some obvious major differences between the typical implementation of switch-based algorithm and that of Curveball algorithms that stem from the use of the adjacency list representation of a network used by the latter, compared to the edge-list representation used by the first. For example, the most common procedure used for fixed-degree network randomization [19] consists, at each step, in sampling two pair of linked nodes, a–b and c–d and rewire them in the form a–d and c–b if and only if neither a–d nor c–b exist already in the network. It is clear that this procedure has two important limitations in terms of performance, one deriving from the probability of performing a successful swap, and the other one deriving from the need to check that performing a given swap will not result in the generation of multiple edges. While the first limitation is strongly dependent on network structure (and we may also imagine situations where the probability of performing a successful swap is identical to the probability of performing a successful Curveball trade), the additional check for multiple edges is a serious bottleneck limiting the efficiency of swap-based algorithms (and one narrowing with network size). As a consequence, performing a performance comparison (for example in terms of CPU time needed to properly randomize a network) between typical swap implementations and their Curveball counterparts would be a trivial exercise. However, the most important question for practitioners as well as theoreticians is how many steps the Curveball algorithms have to run from an initial probability distribution (where an initial state is taken from) to sample from a probability distribution which is close to the uniform

distribution. This number is defined as the *total mixing time*, i.e. the number N of reiteration steps in the Curveball algorithms.

The Curveball algorithm has experimentally been shown to run much faster than the switching algorithm [10]. Intuitively, this property should extend also to the directed and undirected versions of the Curveball. We tested this experimentally, by comparing the total mixing time of the Curveball algorithm with that of the switching model in: two sets of random (i.e. Erdős-Rényi) networks including, respectively, 100 directed and 100 undirected networks having a number of nodes (V) extracted at random between 100 and 1000, and a number of edges equal to $E \times V$, with edges varying randomly between 5 and 50; two sets of power law (i.e. Albert Barabási) networks including, respectively, 100 directed and 100 undirected networks having a number of nodes (V) extracted at random between 100 and 1000; and two empirical networks, namely a directed food-web (listing all trophic interactions recorded in Little Rock Lake, Wisconsin in the United States of America, [20], having 183 nodes and 2476 edges), and an undirected co-occurrence network (representing ecological interactions between bacteria [21], having 316 nodes and 1086 edges).

In order to compare the asymptotic mixing times of these algorithms, that is to assess the increased performance stemming from trading multiple elements at once compared to performing individual swaps, while excluding the above mentioned limitations emerging from the different algorithms' design, we implemented a swap algorithm in the same form of the Curveball but with the additional constrain of permitting only trades of size 1 between adjacency lists.

To track the two algorithms' convergence towards the uniform distribution, we used, as a proxy, the degree of network perturbation, that we measured as the fraction of edges in the target network differing from those in its randomized counterpart [10]). For both the Curveball and the (modified) swap procedure, we recorded network perturbation every 100 step (i.e. trades and swaps), performing a total of 25,000 steps.

In Fig. 4 we show how the Curveball algorithms converge much faster than the switching chain in both the random (Erdős-Rényi) and real world networks. The improvement of the Curveball algorithms against the switching chain was less pronounced for the power law (Albert Barabási) networks. This can be explained by the fact that all vertices in this network have low out-degree (1, 2 or 3), and hence the number of elements traded by the Curveball at each step is often one (i.e. equivalent to a step in the swap procedure). Furthermore, due to the power-law degree distribution, many of the edges will have the same target further limiting the size of trades.

All algorithms were implemented in the R and Python programming language. All the code using in our analyses is publicly available [22]. We also provide user friendly functions implementing the new algorithms in both Python and R programming language as Supplementary material.

Concluding remarks

The increasing understanding of natural systems' complexity is making clear how the current separation between food-web science, and mutualistic and antagonistic bipartite ecological networks may hide important patterns and processes possibly responsible for the emergence and maintaining of diversity [23,24]. New analytical tools are needed to overcome this issue, and there is still long way towards a truly organic theory of ecological interactions. By providing a unifying framework for the randomization of all kinds of networks relevant to the ecological field, we hope that this work may represent a further step in that direction.

Finally, we have focussed here mainly on ecological networks as this was the main motivation behind the development of the original Curveball, and behind our interest in extending its application beyond bipartite networks. Nevertheless, the randomization of large directed and undirected networks is a compelling problem in several fields other than ecology. For example, investigating the structure of social networks is becoming more and more important to improve our understanding of complex societal mechanisms [25], and disease spread dynamics [26]. We are confident that our new methods will be useful in those fields too.

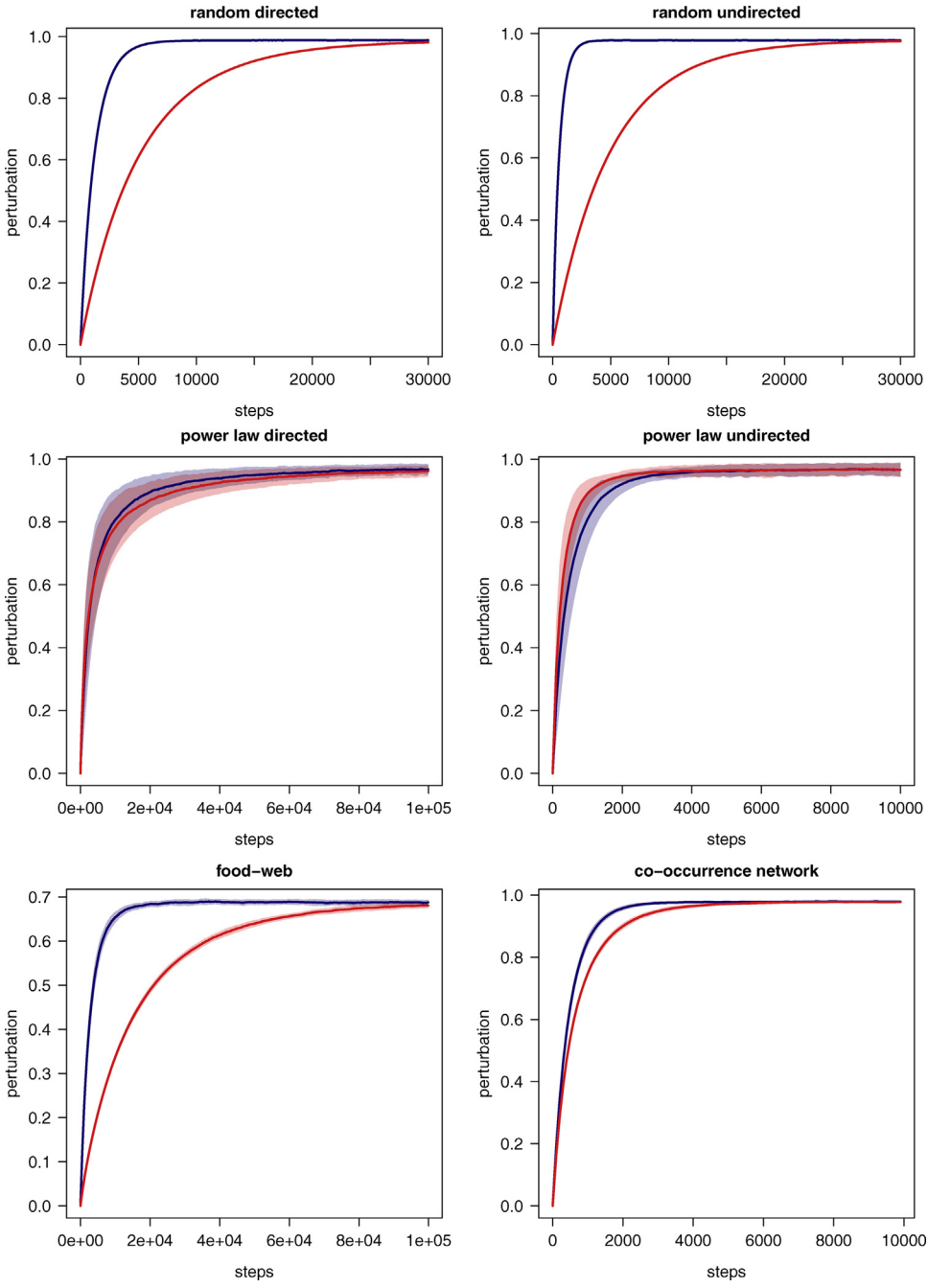

Fig. 4. The degree of network perturbation (measured as the fraction of edges in a randomized network differing from those of the original network) for an increasing number of steps in the Markov chains of the modified swap procedures (red), and of the Curveball algorithms (blue) while randomising: a set of one hundred random (Erdős-Rényi) directed; a set of one hundred random (Erdős-Rényi) undirected; a food-web; and a co-occurrence network. The randomization of the food-web and of the co-occurrence network was replicated one hundred times. Solid lines indicate the average values over the replicates, while shaded areas indicate standard deviation.

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at https://doi.org/10.1016/j.mex.2018.06.018.

References

[1] R. Kannan, P. Tetali, S. Vempala, Simple Markov-Chain Algorithms for Generating Bipartite Graphs and Tournaments (Extended Abstract), (1997) .
[2] A.R. Rao, R. Jana, S. Bandyopadhyay, A Markov chain Monte Carlo method for generating random (0, 1)-matrices with given marginals, Sankhya 58 (1996) 225–242.
[3] Y. Artzy-Randrup, L. Stone, Generating uniformly distributed random networks, Phys. Rev. E 72 (2005) 056708, doi:http://dx.doi.org/10.1103/PhysRevE.72.056708.
[4] N.D. Verhelst, An efficient MCMC algorithm to sample binary matrices with fixed marginals, Psychometrika 73 (4) (2008) 705–728.
[5] C.J. Carstens, Proof of uniform sampling of binary matrices with fixed row sums and column sums for the fast curveball algorithm, Phys. Rev. E 91 (2015) 042812.
[6] H.J. Ryser, Combinatorial properties of matrices of zeros and ones, Can. J. Math. 9 (1957) 371–377.
[7] R. Taylor, Constrained switchings in graphs, Combinatorial Mathematics VIII: Proceedings of the Eighth Australian Conference on Combinatoria Mathematics Held at Deakin University, Geelong, Australia, August 25–29, 1980, Springer Berlin Heidelberg, Berlin, Heidelberg, 1981, pp. 314–336.
[8] S. Maslov, K. Sneppen, Specificity and stability in topology of protein networks, Science 296 (2002) 910–913.
[9] T.M. Fayle, A. Manica, Reducing over-reporting of deterministic co-occurrence patterns in biotic communities, Ecol. Model. 221 (19) (2010) 2237–2242.
[10] G. Strona, D. Nappo, F. Boccacci, S. Fattorini, J. San-Miguel-Ayanz, A fast and unbiased procedure to randomize ecological binary matrices with fixed row and column totals, Nat. Commun. 5 (2014) 4114.
[11] K.D. Lafferty, G. DeLeo, C.J. Briggs, A.P. Dobson, T. Gross, A.M. Kuris, A general consumer-resource population model, Science 349 (6250) (2015) 854–857.
[12] T.C. Ings, J.M. Montoya, J. Bascompte, N. Blüthgen, L. Brown, C.F. Dormann, F. Edwards, D. Figueroa, U. Jacob, J.I. Jones, et al., Ecological networks-beyond food webs, J. Anim. Ecol. 78 (1) (2009) 253–269.
[13] M.B. Araújo, A. Rozenfeld, C. Rahbek, P.A. Marquet, Using species co-occurrence networks to assess the impacts of climate change, Ecography 34 (6) (2011) 897–908.
[14] A. Barberán, S.T. Bates, E.O. Casamayor, N. Fierer, Using network analysis to explore co-occurrence patterns in soil microbial communities, ISME J. 6 (2) (2012) 343.
[15] D. Berry, S. Widder, Deciphering microbial interactions and detecting keystone species with co-occurrence networks, Front. Microbiol. 5 (2014) 219.
[16] S. Jonhson, V. Domínguez-García, M.A. Muñoz, Factors determining nestedness in complex networks, PLoS ONE 8 (9) (2013) e74025.
[17] G. Strona, J.A. Veech, A new measure of ecological network structure based on node overlap and segregation, Methods Ecol. Evol. 6 (8) (2015) 907–915.
[18] A. Berger, M. Müller-Hannemann, Uniform sampling of digraphs with a fixed degree sequence, International Workshop on Graph-Theoretic Concepts in Computer Science, Springer, 2010, pp. 220–231.
[19] S. Maslov, K. Sneppen, Specificity and stability in topology of protein networks, Science 296 (5569) (2002) 910–913.
[20] N.D. Martinez, J.J. Magnuson, T. Kratz, M. Sierszen, Artifacts or attributes? Effects of resolution on the Little Rock Lake food web, Ecol. Monogr. 61 (1991) 367–392.
[21] S. Freilich, A. Kreimer, I. Meilijson, U. Gophna, R. Sharan, E. Ruppin, The large-scale organization of the bacterial network of ecological co-occurrence interactions, Nucleic Acids Res. 38 (12) (2010) 3857–3868.
[22] https://github.com/queenBNE/Curveball.
[23] K.D. Lafferty, S. Allesina, M. Arim, C.J. Briggs, G. De Leo, A.P. Dobson, J.A. Dunne, P.T. Johnson, A.M. Kuris, D.J. Marcogliese, et al., Parasites in food webs: the ultimate missing links, Ecol. Lett. 11 (6) (2008) 533–546.
[24] S. Allesina, S. Tang, Stability criteria for complex ecosystems, Nature 483 (7388) (2012) 205.
[25] S. Aral, D. Walker, Identifying influential and susceptible members of social networks, Science (2012) 1215842.
[26] S. Eubank, H. Guclu, V.A. Kumar, M.V. Marathe, A. Srinivasan, Z. Toroczkai, N. Wang, Modelling disease outbreaks in realistic urban social networks, Nature 429 (6988) (2004) 180.