PAFit: an R Package for the Non-parametric Estimation of Preferential Attachment and Node Fitness in Temporal Complex Networks

Thong Pham
RIKEN Center for AIP

Paul Sheridan
Hirosaki University

Hidetoshi Shimodaira
Kyoto University
RIKEN Center for AIP

Abstract

Many real-world systems are profitably described as complex networks that grow over time. Preferential attachment and node fitness are two simple growth mechanisms that not only explain certain structural properties commonly observed in real-world systems, but are also tied to a number of applications in modeling and inference. While there are statistical packages for estimating various parametric forms of the preferential attachment function, there is no existing package for a non-parametric estimation, which would allow finer inspections on the famous 'rich-get-richer' phenomenon as well as provide clues to explain non-standard structural properties observed in real-world networks. This paper introduces the R package **PAFit**, which implements statistical methods for estimating the preferential attachment function and node fitness non-parametrically, as well as a number of functions for generating complex networks from these two mechanisms. The main computational part of the package is implemented in C++ with OpenMP to ensure scalability to large-scale networks. In this paper, we first introduce the main functionalities of **PAFit** through simulated examples, and then use the package to analyze a collaboration network between scientists in the field of complex networks. The results indicate the joint existence of 'rich-get-richer' and 'fit-get-richer' phenomena in the collaboration network. The estimated attachment function is almost linear, which means that the probability an author develops a new collaboration is proportional to their current number of collaborators. Furthermore, the estimated fitnesses reveal many familiar names of the complex network field as top fittest scientists.

Keywords: temporal networks, dynamic networks, preferential attachment, fitness, rich-get-richer, fit-get-richer, R, C++, **Rcpp**, OpenMP.

1. Introduction

Since the end of the last century, complex networks have been increasingly used in modeling many temporal relations found in diverse fields (Dorogovtsev and Mendes 2003; Caldarelli 2007; Newman 2010). Some notable examples include collaboration networks between authors in a scientific field (Newman 2001), connection networks between computers on the Internet (Barabási *et al.* 2000), and sexual relation networks between members of a community (Liljeros *et al.* 2001). The primary motivation for using complex networks as a simplified representation of real-world systems is that they shed light on the behaviors of

complex systems through the study of underlying patterns of connections. Although this is an over-simplification for systems depending heavily on domain-specific details, this approach nevertheless offers a first view of a system's topological properties, and can be used to guide subsequent in-depth analyses.

Among the most important real-world network structural properties is degree distribution. Degree distribution lets us understand the proportion of highly and lowly connected nodes in a network. Since most dynamical network processes must travel frequently through highly-connected nodes, this understanding in turn sheds light on the answers of important practical questions, including how to prevent the spreading of rumors (Nekovee *et al.* 2007), how to stop a virus outbreak (Pastor-Satorras and Vespignani 2001), and how to guard against cybernetic attacks (Albert *et al.* 2000).

The degree distributions of many real-world networks have been found to be heavy-tailed (Albert and Barabási 1999). The best-known heavy-tailed distribution in network science is the power-law, which is a distribution where the number of nodes in a network with degree k is proportional to $k^{-\gamma}$ for $2 < \gamma \leq 3$. Besides the power-law, there is emerging evidence that real-world network degree distributions have other heavy-tailed forms, including the log-normal (Redner 2005), exponential (Dunne *et al.* 2002), stretched exponential (Newman *et al.* 2002), and power-law with exponential cut-off (Clauset *et al.* 2009).

All of these heavy-tailed distributions differ from the light-tailed binomial degree distribution, which is characteristic of networks produced by the classical Erdös-Rényi (ER) random graph model (Erdös and Rényi 1959). This prompted the network scientists to search for new modeling ingredients capable of explaining heavy-tailed degree distributions. It turns out that temporal complex network models that incorporate growth mechanisms offer a powerful modeling framework for achieving this end.

Temporal complex network models, or temporal network models for short, are probabilistic generative models of a real-world network that change with time. In its most common form, a temporal network model assumes that a network grows gradually from some initial state by the addition of new nodes and edges over a large number of discrete time-steps. Some well-known basic models in the field of complex networks are the Barabási-Albert (BA) model (Albert and Barabási 1999) and the Bianconi-Barabási (BB) model (Bianconni and Barabási 2001). More complex growth models that are used in the field include exponential random graph models (Ripley *et al.* 2013; Krivitsky and Handcock 2016) and dynamic stochastic block models (Matias and Miele 2016). Growth mechanisms, which govern how a node acquires new edges in the growth process, are the most important elements that distinguish different temporal network models.

This paper focuses on estimating two interpretable growth mechanisms: preferential attachment (PA) and node fitness. In the PA mechanism, the probability P_i a node v_i gets a new edge in the future is proportional to some positive function A_{k_i} of its current degree k_i. This function is called the attachment function. The name 'preferential attachment' stems from the motivation for the mechanism: if A_k is an increasing function on average, a highly connected node will acquire more edges than a lowly-connected node, which is an appealing property in many real-world situations. From now, we will say that PA exists if A_k is an increasing function on average. The opposite of PA is called anti-PA, in which A_k is a decreasing function in average.

Note that, however, the meaning we use here differs from the original meaning of the term

'preferential attachment' used in the BA model, which means only the linear case of $A_k = k$. This linear form in fact has been long known in other fields with various names such as 'rich-get-richer' (Simon 1955) and 'cummulative advantage' (Price 1976). When A_k assumes the log-linear form of k^α, with α called the attachment exponent, we have the generalized BA model (Krapivsky *et al.* 2001).

While P_i depends on the degree of v_i in the PA mechanism, in the fitness mechanism P_i depends only on a positive quantity η_i called the fitness of node v_i. We can interpret η_i as the intrinsic attractiveness of v_i. The fitness mechanism offers a simple way to express the variance in edge-acquiring abilities between nodes with the same degree. For example, two early-career scientists with roughly the same number of collaborators at some point in time may acquire different numbers of collaborators in the future based on their intrinsic fitnesses.

The PA and node fitness mechanisms combine to produce a wide range of degree distributions. In their combined form, the probability P_i is proportional to the product of A_{k_i} and η_i:

$$P_i \propto A_{k_i} \times \eta_i. \tag{1}$$

Based on the functional form of A_k and the distribution of η_i, the model depicted in Eq. (1) can produce networks with various degree distributions (Bianconni and Barabási 2001; Caldarelli *et al.* 2002; Borgs *et al.* 2007; Kong *et al.* 2008). In Section 2, we will discuss the relation of Eq. (1) with existing statistical models.

Equation (1) has a number of applications. Based on the functional forms of A_k and η_i, we can check whether two important social phenomena called 'rich-get-richer' or 'fit-get-richer' exist in a temporal network (Pham *et al.* 2016). The two mechanisms have also been proposed to be the underlying mechanisms of another phenomenon called the 'generalized friendship paradox' (Feld 1991; Eom and Jo 2014; Momeni and Rabbat 2015). They are also used in inference problems in biological networks (Sheridan *et al.* 2010; Guetz and Holmes 2011), the World Wide Web (Kong *et al.* 2008), Internet topology graphs (Bezáková *et al.* 2006), and citation networks (Wang *et al.* 2013; Sinatra *et al.* 2016). Finally, we can classify real-world temporal network data based on the estimated attachment exponent of A_k (Kunegis *et al.* 2013).

While there are existing R packages that estimate PA in a growing network, for example, packages **tergm** (Krivitsky and Handcock 2016) and **RSiena** (Ripley *et al.* 2013), these packages, however, only allow parametric estimation of the A_k function. This means that one has to assume a functional form for A_k, rather than learning it from observed data without constraint. Non-parametric estimation of A_k allows finer inspection on the 'rich-get-richer' phenomenon (Pham *et al.* 2015, 2016) as well as provide clues to explain irregularities observed in real-world degree distributions (Sheridan and Onodera 2018).

This paper introduces the R package **PAFit** (Pham *et al.* 2017), which fills the gap with an implementation of the standard PA and node fitness non-parametric estimation procedures. In particular, we implement Jeong's method (Jeong *et al.* 2003), Newman's method (Newman 2001) and the PAFit method (Pham *et al.* 2015, 2016) in the package. The first two are heuristic methods that are widely used in estimating non-parametrically the attachment function A_k in isolation, while the last one is a statistical method that can non-parametrically estimate either A_k (or η_i) in isolation or simultaneously estimate the two mechanisms. Although using PAFit is advisable in almost every circumstance, Jeong's method and Newman's method are still widely used and might still be appropriate in certain situations. Therefore,

the inclusion of the two heuristic methods in the package is warranted. We discuss their strengths and shortcomings in Section 2 when we provide an overview of the methodology and related statistical models.

The package also implements a variety of functions to simulate temporal networks from the PA and node fitness mechanisms, as well as functions to plot the estimated results and underlying uncertainties. We review **PAFit**'s main functions in Section 3. Before demonstrating their usages on three simulated examples in Section 5, we will discuss how **PAFit** relates with existing network analysis packages in Section 4. We provide a systematic simulation to asses the results of the non-parametric joint estimation in Section 6 before showing a complete end-to-end work-flow analyzing a collaboration network of scientists from the field of complex networks in Section 7. Finally, concluding remarks are given in Section 8.

2. Mathematical background

Here we review the standard methods for estimating the attachment function A_k and node fitnesses η_i in a temporal network. In Section 2.1, we state the network growth model used in the package as well as discuss its relation with exiting statistical models. We review the estimation of A_k in isolation in Section 2.2, then the estimation of η_i in isolation in Section 2.3, and finally the joint estimation of A_k and η_i in Section 2.4.

2.1. Network model

First we describe the General Temporal (GT) model (Pham *et al.* 2016) used in **PAFit**. The model is a generalization of many well-known temporal network models in the complex network field.

The GT is a model for temporal networks at discrete times. Starting from some initial network G_0, at time $t = 1, \cdots, T$, $m(t)$ new edges and $n(t)$ new nodes are added to G_{t-1} to form G_t. The GT model assumes that the parameters govern the distributions of G_0, $m(t)$ and $n(t)$ do not involve A_k and η_i. Under these assumptions, the GT model assumes that the probability that a node v_i with degree $k_i(t)$ receives a new edge at time t is:

$$P_i(t) = A_{k_i(t)} \times \eta_i. \tag{2}$$

Equation (2) encompasses several well-known growing network models based on PA and node fitness as summarized in Table 1. Unlike the BA or BB models, the GT model allows for the emergence of new edges between old nodes and can handle both undirect and directed networks. We refer readers to Supplementary Information Section S2.2 in Pham *et al.* (2016) for the definition of the model in the case of undirected networks.

The GT model can be viewed as a special case of the exponential random graph models used in the R packages **RSiena** and **tergm**. We will defer the details of these models to Section 4 when we compare the two packages with **PAFit**.

Not limiting ourselves to the complex network field, the GT model bears some similarities to the contagious Poisson process (Coleman 1964; Allison 1980) and the conditional frailty model (Kelly and Lim 2000; Box-Steffensmeier and De Boef 2006). In the contagious Poisson process, the initial propensity of each node plays a similar role to that of node fitness and the rate of enforcement represents the PA mechanism. In the conditional frailty model, while

Temporal network model	Attachment function	Node fitness
Growing ER model (Callaway *et al.* 2001)	$A_k = 1$	$\eta_i = 1$
BA model	$A_k = k$	$\eta_i = 1$
Caldarelli model	$A_k = 1$	Free
BB model	$A_k = k$	Free

Table 1: Some well-known temporal network models in the complex network field that are special cases of model defined by Eq. (2).

the frailty of each node describes the heterogeneity among nodes and thus is similar to node fitness, the event-based baseline hazard rate has the same effect as the non-parametric function A_k.

2.2. Attachment function estimation

The methods for estimating the attachment function A_k in isolation assume a simplified version of Eq. (2), in which the η_i are assumed to be 1. Thus the probability $P_i(t)$ in Eq. (2) depends only on A_k. Perhaps the most frequently-encountered parametric version of this model is the log-linear for $A_k = k^\alpha$ with attachment exponent α. Network scientists are particularly interested in estimating α, since the asymptotic degree distribution of the network corresponds to simple regions of α. If α is less than unity (the sub-linear case), then the degree distribution is a stretched exponential, while in the super-linear case of $\alpha > 1$, one node will eventually get all the incoming new edges (Krapivsky *et al.* 2001). It is only the linear case of $\alpha = 1$ that gives rise to a power-law distribution.

Concerning this model, there are three main estimation methods for A_k: Jeong's method (Jeong *et al.* 2003), Newman's method (Newman 2001), and PAFit (Pham *et al.* 2015). Jeong's method basically makes a histogram of the number of new edges n_k connected to a node with degree k, then divides n_k by the number of nodes with degree k in the system to get A_k (Jeong *et al.* 2003). Jeong's method has the merit of being simple, but estimates obtained using the method are subject to high variance and low accuracy (Pham *et al.* 2015). By contrast, Newman's method combines a series of histograms for lower variance and higher accuracy (Newman 2001). Note that in PAFit we implemented a corrected version of Newman's original method (Pham *et al.* 2015). The main drawback of Newman's method is that the mathematical assumption behind its derivation only holds when $\alpha = 1$, thus the method amounts to an approximation when $\alpha \neq 1$ (Pham *et al.* 2015).

The final method is PAFit (Pham *et al.* 2015). It iteratively maximizes an objective function that is a combination of the log-likelihood of the model with a regularization term for A_k by a Minorization-Maximization (MM) algorithm (Hunter and Lange 2000). There is a hyper-parameter, called r, in the method that controls the strength of the regularization. PAFit chooses r automatically by cross-validation (Pham *et al.* 2016). We defer the details to Section 2.4. The method is not only able to recover A_k accurately, but also can estimate confidence intervals of the estimated A_k for each k (Pham *et al.* 2015). Its main drawback is that it might be slow, since it is an iterative algorithm.

2.3. Node fitness estimation

When we consider only node fitnesses, there are two generative models in the literature with different assumptions on the functional form of A_k in Eq. (2). While the Caldarelli model (Caldarelli *et al.* 2002) assumes that A_k is 1 for all k, the BB model (Bianconni and Barabási 2001) assumes that $A_k = k$. Both models have been shown to generate networks with various heavy-tailed distributions (Borgs *et al.* 2007; Kong *et al.* 2008).

Node fitnesses in both models can be estimated by variants of the PAFit method proposed in Pham *et al.* (2016), by either setting $A_k = k$ for the BB model or $A_k = 1$ for the Caldarelli model. These estimation methods use MM algorithms that maximize the corresponding log-likelihood functions with a regularization term that regularizes the distribution of η_i. Specifically, the inverse variance of this distribution is controlled by a hyper-parameter, called s, which is chosen automatically by cross-validation. We defer a more detail discussion to the next section. We note that node fitnesses in the BB model can also be estimated by the method in Kong *et al.* (2008). But since PAFit has been shown to outperform this method (Pham *et al.* 2016), we did not include it in the package.

2.4. Joint estimation of the attachment function and node fitnesses

Finally, by using the full model in Eq. (1) the method PAFit in Pham *et al.* (2016) can jointly estimate A_k and η_i. We note this full model includes all the temporal network models shown in Table 1. For a more complete table, see Table 1 in Pham *et al.* (2016).

The objective function of PAFit is a combination of the log-likelihood of the full model defined by Eq. (2) and two regularization terms: one for A_k and one for η_i. While we refer readers to Supplementary Information Section S2.3 in Pham *et al.* (2016) for a complete presentation, we will sketch here the log-likelihood function for the case of directed networks. Assume the set of observed snapshots is $\{G_t\}_{t=0}^T$. Let $\mathbf{A} = [A_0\ A_1 \cdots A_{K-1}]^\top$ be the vector of the PA function and $\boldsymbol{\eta} = [\eta_1\ \eta_2 \cdots \eta_N]$ be the vector of node fitnesses. Here K is the maximum degree appearing in the growth process and N is the total number of nodes at the end of the process. Let $z_i(t)$ be the number of new edges connected to node v_i at time-step t. Eq. (2) implies that $\{z_i(t)\}_{i=1}^N$ follows a multinomial distribution with parameters $\{\pi_i(t)\}_{i=1}^N$ where

$$\pi_i(t) = \frac{A_{k_i(t)}\eta_i}{\sum_{j=1}^N A_{k_j(t)\eta_j}}. \tag{3}$$

Here we use the convention $k_j(t) = -1$ for a node that did not exist at time-step t and $A_{-1} = 0$. Using Eq. (3), one can write the likelihood of each snapshot $G_1, \cdots, G_T$. The log-likelihood function of the temporal network $\{G_t\}_{t=0}^T$ is then the sum of the log-likelihood of each snapshot and is equivalent to:

$$l(\mathbf{A}, \boldsymbol{\eta}) = \sum_{t=1}^{T}\sum_{i=1}^{N} z_i(t) \log A_{k_i(t)} + \sum_{t=1}^{T}\sum_{i=1}^{N} z_i(t) \log \eta_i - \sum_{t=1}^{T}\sum_{i=1}^{N} z_i(t) \log \sum_{j=1}^{N} A_{k_j(t)}\eta_j + C, \tag{4}$$

with C being a constant that depends neither on $\mathbf{A}$ or $\boldsymbol{\eta}$.

The regularization term for A_k is defined by

$$reg_A = -r \sum_{k=1}^{K-2} w_k \left(\log A_{k+1} + \log A_{k-1} - 2\log A_k\right)^2, \tag{5}$$

with $w_k = \sum_{t=1}^{T} m_k(t)$ and $m_k(t)$ the number of edges that connect to a degree k node at time-step t. This term controls the shape of A_k. When r is large, A_k becomes more linear in log-scale. In the limiting case when r is large, we effectively assume that $A_k = k^\alpha$ (Pham *et al.* 2016). Thus this covers the case of $\alpha = 1$ in the BA or the BB models and the case of $\alpha = 0$ in the growing ER or the Caldarelli model.

The regularization term for the node fitnesses is defined by

$$reg_F = \sum_{i=1}^{N}((s-1)\log\eta_i - s\eta_i). \tag{6}$$

This term is the sum of the logarithms of Gamma distribution densities with mean 1 and variance $1/s$. The regularization is equivalent to placing such Gamma distribution as prior independently for each fitness η_i. The larger the value of s, the more tightly concentrated the values of η_i become. If s is infinitely large, then all η_i will take the same value. This is equivalent to estimating the attachment function in isolation.

To conclude: joint estimation with the above regularization terms also compasses the two cases of estimating either A_k or η_i in isolation. In particular, we maximize the following objective function:

$$J(\mathbf{A}, \boldsymbol{\eta}) = l(\mathbf{A}, \boldsymbol{\eta}) + reg_A + reg_F,$$

with a MM algorithm. At each iteration, the algorithm replaces the objective function with an easier-to-maximize surrogate function and this surrogate function is maximized instead. The surrogate function is chosen in such a way that the objective function value is guaranteed to be non-decreasing over iterations. We refer the readers to Hunter and Lange (2004) for the definition of surrogate functions and techniques to derive them. In the surrogate function, the variables are often seperable and thus the maximization at each iteration can be parallerized. While we refer readers to Supplementary Information Section S2.4 of Pham *et al.* (2016) for a detailed discussion, the essence of the MM algorithms in **PAFit** is to linearize the term $\log\sum_{j=1}^{N} A_{k_j(t)}\eta_j$ in Eq. (4) and to apply Jensen's inequality to make the variables in Eq. (5) separable.

As mentioned in the two previous sections, the values of r and s are automatically selected by cross-validation. In particular, the dataset is divided into a learning part and a testing part in which the ratio of the number of new edges in the learning part to that of the whole dataset is set at the default value $p = 0.75$. For each combination of r and s, we use the learning data to get the estimated value of $\mathbf{A}$ and $\boldsymbol{\eta}$ and plug these estimated values into Eq. (4) to calculate the log-likelihood of the testing data. The combination of r and s that maximize this log-likelihood is then chosen. The method then re-estimates $\mathbf{A}$ and $\boldsymbol{\eta}$ using the whole dataset with the chosen combination of r and s.

3. Package overview

The **PAFit** package provides functions to simulate various temporal network models, gather essential network statistics from raw input data, and use these summarized statistics in the estimation of A_k and η_i. The heavy computational parts of the package are implemented in C++ through the use of the **Rcpp** package (Eddelbuettel and François 2011; Eddelbuettel 2013). Furthermore, users with a multi-core machine can enjoy a hassle-free speed up through

OpenMP parallelization mechanisms implemented in the code. Apart from the main functions, the package also includes a real-world collaboration network dataset between scientists in the field of complex networks. Table 2 summarizes the main functions in the package. In what follows, we will review the main package functions one by one.

Firstly, most well-known temporal network models based on PA and node fitness mechanisms can be easily simulated using the package. **PAFit** implements `generate_BA` for the BA model, `generate_ER` for the growing ER model, `generate_BB` for the BB model, and `generate_fit_only` for the Caldarelli model. These functions have many customizable options. For example, the number of new edges at each time-step is a tunable stochastic variable; see Table 3 for descriptions of the parameters. They are actually wrappers of the more powerful `generate_net` function, which simulates networks with more flexible attachment function and node fitness settings.

In any case, the output of these functions is a `PAFit_net` object, which is a list with four fields: `type`, `fitness`, `PA`, and `graph`. The `type` field is a string indicating the type of network: `"directed"` or `"undirected"`. This field is `"directed"` for the networks generated by the simulation functions. The `fitness` and `PA` fields contain the true node fitnesses and PA function, respectively.

The `graph` field contains the generated temporal network in a three-column matrix format. Each row of this matrix is of the form `(id_1 id_2 time_stamp)`. While `id_1` and `id_2` are IDs of the source node and the destination node, respectively, `time_stamp` is the birth time of the edge. This is the so-called edgelist format in which raw temporal networks are stored in many online repositories (Kunegis 2013; Leskovec and Krevl 2014). We will discuss how to use functions provided by **PAFit** to convert this edgelist format to formats used in other network analysis packages in the next section. One can apply the function `plot` directly to a `PAFit_net` object to visualize its contents.

The second functionality of **PAFit** is implemented in `get_statistics`. With its core part implemented in C++, this function efficiently collects all temporal network summary statistics that are needed in the subsequent estimation of PA and node fitnesses. The input network is assumed to be stored in a `PAFit_net` object. One can use the function `graph_from_file` to read an edgelist graph from a text file into a `PAFit_net` object, or convert an edgelist matrix to this class by the function `as.PAFit_net`.

The edgelist matrix is assumed to be in the same format as simulated graphs from **PAFit**, i.e., each row is of the form `(id_1 id_2 time_stamp)`. The node IDs are required to be integers greater than -1, but need not to be contiguous. Note that `(id -1 t)` describes a node `id` that appeared at time t without any edge. There are no assumptions on the values or data types of `time_stamp`, other than that their chronological order is the same as what the R function `order` returns. Examples of time-stamps that satisfy this requirement are the integer vector `1:T`, the format 'yyyy-mm-dd', and the POSIX time.

The `get_statistics` function automatically handles both directed and undirected networks. It returns a list containing many statistics that can be used to characterize the network growth process. Notable fields are `m_tk` containing the number of new edges that connect to a degree-k node at time-step t, and `node_degree` containing the degree sequence, i.e., the degree of each node at each time-step.

The most important functionality of **PAFit** relates to the estimation of the attachment function and node fitnesses of a temporal network. This is implemented through various methods.

Function	Main Input	Output
`generate_ER`	network parameters	network from the growing ER model
`generate_BA`	network parameters	network from the generalized directed BA model
`generate_BB`	network parameters	network from the BB model
`generate_fit_only`	network parameters	network from the Caldarelli model
`generate_net`	network parameters	network from the GT model
`get_statistics`	`PAFit_net` object containing the network	`PAFit_data` object containing summary statistics
`Jeong`	`PAFit_net` object and `PAFit_data` object	estimated PA function by Jeong's method
`Newman`	`PAFit_net` object and `PAFit_data` object	estimated PA function by Newman's method
`only_A_estimate`	`PAFit_net` object and `PAFit_data` object	estimated PA function by PAFit
`only_F_estimate`	`PAFit_net` object and `PAFit_data` object	estimated node fitnesses by PAFit
`joint_estimate`	`PAFit_net` object and `PAFit_data` object	estimated PA function and node fitnesses by PAFit
`to_networkDynamic`	`PAFit_net` object	`networkDynamic` object
`from_networkDynamic`	`networkDynamic` object	`PAFit_net` object
`to_igraph`	`PAFit_net` object	`igraph` object
`from_igraph`	`igraph` object	`PAFit_net` object
`graph_to_file`	`PAFit_net` object	text file in either edgelist format or `gml`
`graph_from_file`	text file in edgelist format or `gml`	a `PAFit_net` object
`as.PAFit_net`	edgelist matrix	`PAFit_net` object
`test_linear_PA`	degree vector	`Linear_PA_test_result` object

Table 2: Summary of the main functions in the **PAFit** package.

Parameter (default value)	Description
`N` (1000)	total number of nodes in the network
`num_seed` (2)	initial graph is a circle with `num_seed` nodes
`multiple_node` (1)	number of new nodes added at each time-step
`m` (1)	number of edges of a new node
`alpha` (1)	attachment exponent α when we assume $A_k = k^\alpha$
`mode_f` ("gamma")	distribution of node fitnesses: gamma, log-normal or power-law
`s` (10)	distribution of node fitnesses has mean 1 and variance $1/s$

Table 3: Main parameters in network-generating functions in the **PAFit** package.

There are three usages: estimation of the attachment function in isolation, estimation of node fitnesses in isolation, and the joint estimation of the attachment function and node fitnesses.

The functions for estimating the attachment function in isolation are: `Jeong` for Jeong's method, `Newman` for Newman's method, and `only_A_estimate` for the PAFit method in Pham *et al.* (2015). For estimation of node fitnesses in isolation, `only_F_estimate` implements a variant of the PAFit method in Pham *et al.* (2016). For the joint estimation of the attachment function and node fitnesses, we implement the full version of the PAFit method (Pham *et al.* 2016) in `joint_estimate`. The input of these functions is the output object of the function `get_statistics`. The output objects of these functions contain the estimation results as well as some additional information pertaining to the estimation process.

In Table 4, we show the input parameters of `joint_estimate`, the most important function in **PAFit**. This function takes the temporal network `net_object` and the summarized statistics `net_stat` as the main inputs. There are three parameters that control the estimation process: `p`, `stop_cond`, and `mode_reg_A`. The parameter `p` specifies the ratio of the number of new edges in the learning data to that of the full data in the cross-validation step. Following Pham *et al.* (2016), its default value is set at 0.75. The parameter `stop_cond` specifies the threshold ϵ: the iterative algorithm will continue until the relative difference of the objective function $J(\mathbf{A}, \boldsymbol{\eta})$ between two successive iterations falls below this threshold (Pham *et al.* 2016; Zhou *et al.* 2011). The default value $\epsilon = 10^{-8}$ is set following Pham *et al.* (2016). The parameter `mode_reg_A` specifies the regularization term for A_k. The default value `mode_reg_A = 0` corresponds to the regularization term in Eq. (5) (Pham *et al.* 2016). When `mode_reg_A = 1`, the following regularization term is used:

$$-r \sum_{k=2}^{K-1} w_k \left\{ \frac{\log A_{k+1} - \log A_k}{\log(k+1) - \log k} - \frac{\log A_k - \log A_{k-1}}{\log k - \log(k-1)} \right\}^2. \tag{7}$$

Although this regularization term will enforce exactly the form $A_k = k^\alpha$, it is significantly slower to optimize this regularization term while the improvement over Eq. (5) is little.

Finally, although one can roughly assess whether PA exists in the network by visual inspection of the estimated PA function, Handcock and Jones (2004) provide a method to test whether the linear PA-only case, i.e., $A_k = k$ and $\eta_i = 1$, is consistent with a given degree vector. We implemented this method in the function `test_linear_PA`. This function chooses the best fitted distribution to a given degree vector among a set of distributions by comparing the Akaike Information Criterion (Akaike 1974) or the Bayesian Information Criterion (Raftery 1995). The set of distributions are Yule, Waring, Poisson, geometric, and negative binomial.

Parameter	Default value
`net_object`	no default value
`net_stat`	`get_statistics(net_object)`
`p`	0.75
`stop_cond`	10^{-8}
`mode_reg_A`	0

Table 4: Parameters of the `joint_estimate` function and their default values.

The linear PA-only case corresponds to Yule or Waring (Yule 1925; Irwin 1963).

4. Related network packages

Since network analysis has been an important field for a long time, various aspects of it have been implemented in a large number of software packages. To our best effort, we have confirmed that the non-parametric joint estimation of PA and fitness mechanisms in a growing network is not implemented elsewhere. Restricting the discussion to packages in R, there are some notable implementations of related statistical network models. For example, stochastic block models in packages **igraph** (Csardi and Nepusz 2006), **sna** (Butts 2016), **blockmodels** (INRA and Leger 2015) and **dynsbm** (Matias and Miele 2016, 2018); exponential random graph models in packages **ergm** (Hunter *et al.* 2008; Handcock *et al.* 2017), **tergm** (Krivitsky and Handcock 2016), **hergm** (Schweinberger *et al.* 2018), **btergm** (Leifeld *et al.* 2018), and **RSiena** (Ripley *et al.* 2013, 2018); and latent space models in the package **latentnet** (Krivitsky and Handcock 2008, 2017).

The **dynsbm** package estimates a dynamic stochastic block model in which nodes are assumed to belong to some latent groups which can be varying with time, and the edge weight between two nodes at any time follows some parametric distribution. The package can deal with both discrete and continuous weighted edges.

The **igraph** package contains the functions `sample_pa` and `sample_growing` which are the equivalents of `generate_BA` and `generate_ER` in **PAFit**, respectively. Although **igraph** also generates networks from many other mechanisms, it does not contain any function for estimating the PA function and/or node fitnesses. It does contain many functionalities for dealing with stochastic block models and various other network models.

Some of the above packages are included in the extensive meta-package **statnet** (Handcock *et al.* 2008, 2016). In **statnet**, packages that deal with temporal networks are: **networkDynamic** (Butts *et al.* 2016), **tsna** (Bender-deMoll and Morris 2016), and **tergm**. The **networkDynamic** package provides the `networkDynamic` class to store dynamic networks and various functions to manipulate them. The **tsna** package calculates many temporal statistics of a dynamic network stored in a `networkDynamic` object.

The closest packages to **PAFit** that estimate PA in a temporal network are **tergm** and **RSiena**. The two packages both fit exponential random graph models based on an extensive and customizable list of network statistics or user-defined node covariates. Ignoring technical

details, **tergm** and **RSiena** essentially employ a model in which

$$P_i(t) \propto \exp\left(\sum_{j=1}^{p} \beta_j s_{j,i}(t)\right), \tag{8}$$

with $s_{j,i}(t)$ the j-th effect calculated at node v_i at time t. Each effect is a fixed transformation of some network statistic or node covariates. The goal is to estimate the coefficients $\{\beta_j\}_{j=1}^{p}$. Recall that **PAFit** essentially explains the increase in the dependent variable, which is the degree of node v_i, by two independent effects: node fitness η_i and PA value $A_{k_i(t)}$. The advantages of **tergm** and **RSiena** over **PAFit** are: (1) the independent effects $s_{j,i}(t)$ can include either node covariates or a wide range of network statistics that are not limited to degree, and (2) one can change $P_i(t)$ in Eq. (8) to the probability of some event that depends on both network statistics and node covariate.

PAFit, however, has advantages over **tergm** and **RSiena** when one focuses on using the non-parametric PA function and node fitnesses to explain the changes in the degree vector. Regarding PA, the main focus of **tergm** and **RSiena** is parametric estimation of A_k. All pre-implemented functions of $s_{j,i}(t)$ in Eq. (8) correspond to some parametric forms of A_k. Popular choices, for example, are $s_{j,i}(t) = k_i(t)$, $1/k_i(t)$, $\log k_i(t)$ and $\sqrt{k_i(t)}$, which correspond respectively to the following parametric forms of the PA function: $A_k = \exp \beta k$, $\exp(\beta/k)$, k^β and $\exp \beta\sqrt{k}$.

Although it is theoretically possible to describe a non-parametric A_k function by using indicator functions $s_{j,i}(t) = 1_{k_i(t)=j}$, **tergm** and **RSiena** contain no regularization terms for the joint estimation of the non-parametric PA function and node fitnesses. Joint estimation without regularization terms would be intractable, since the number of parameters is too high. On the other hand, **PAFit** is specifically designed for estimating A_k non-parametrically with node fitnesses, since it has two regularization terms in Eqs. (5) and (6), together with the cross-validation step used to choose suitable regularization parameters.

We also note one minor advantage of **PAFit** over **RSiena** is that **RSiena** assumes the node set is fixed in time and thus cannot handle the addition of new nodes without introducing some approximations (for example, assuming that all the nodes appear at the initial time), while **PAFit** does not assume so.

PAFit provides functionalities to communicate with existing network analysis packages. Using `to_networkDynamic` and `from_networkDynamic`, one can convert a `PAFit_net` object to a **networkDynamic**'s `networkDynamic` object and vice versa. The functions `to_igraph` and `from_igraph` do the same for **igraph**'s `igraph` objects. The extensive functions of **statnet** and **igraph** packages can then be used. One can also output the graph stored in a `PAFit_net` object to the universal `gml` format by the function `graph_to_file`, or read from a `gml` file by the function `graph_from_file`.

5. Package usage

Here we show three usages of **PAFit**: the estimation of the attachment function A_k in isolation in Section 5.1, the estimation of node fitnesses η_i in isolation in Section 5.2, and the joint estimation of A_k and the η_i values in Section 5.3.

5.1. Attachment function estimation

First we generate a network from a directed version of the BA model, called Price's model (Price 1976). The network consists of $N = 1000$ nodes with $m = 5$ new edges added at each time-step.

```
R> set.seed(1)
R> library(PAFit)
R> sim_net_1 <- generate_BA(N = 1000, m = 5)
```

Recall that A_k is linear in the BA model, i.e., the attachment exponent α is equal to 1, and the node fitnesses are uniform.

One can observe the emergence of hubs in this network by visualizing the generated graph at various time-steps by the function `plot`. The following script plots the network snapshot at time $t = 1$ in Figure 1a and its corresponding degree distribution in Figure 1d:

```
R> plot(sim_net_1, slice = 1)
R> plot(sim_net_1, slice = 1, plot = "degree")
```

Note that if the network is directed, as it is in this example, the option `plot = "degree"` will plot the in-degree distribution. We can plot in the same way the network snapshots at time $t = 10$ and $t = 100$ in Figures 1b and 1c and their corresponding degree distributions in Figures 1e and 1f.

The next step is to use the function `get_statistics` to get the summary statistics for the temporal network:

```
R> stats_1 <- get_statistics(sim_net_1)
```

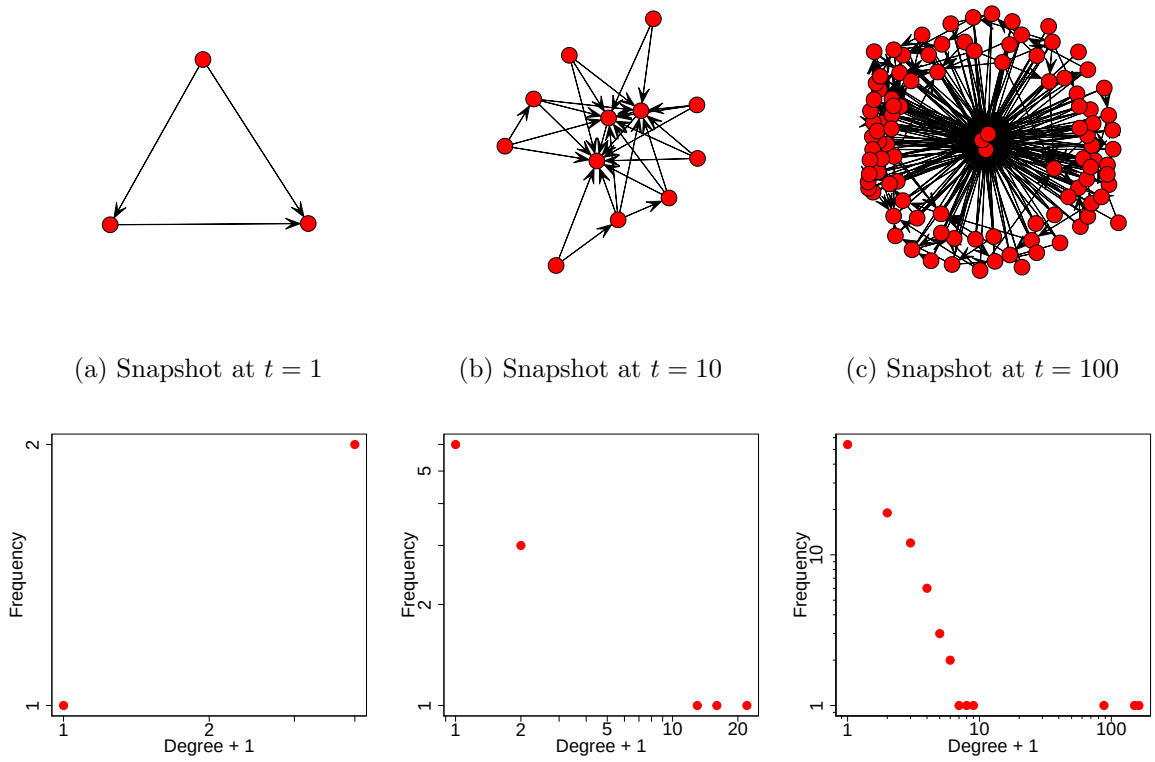
With `stats_1` containing all the needed summary statistics, we then apply the three methods of estimating the attachment function in isolation:

```
R> result_Jeong   <- Jeong(sim_net_1, stats_1)
R> result_Newman  <- Newman(sim_net_1, stats_1)
R> result_PA_only <- only_A_estimate(sim_net_1, stats_1)
```

Let us explain `result_PA_only` in more detail. Information on the estimated results as well as the estimation process can be viewed by invoking `summary`:

```
R> summary(result_PA_only)

Estimation results by the PAFit method.
Mode: Only the attachment function was estimated.
Estimated r parameter:  0.1
Estimated attachment exponent:  1.001139
Two-sigma confidence interval of the attachment exponent: ( 0.9908913 , 1.011387 )
-----------------------------------------
Additional information:
Number of bins:  50
Number of iterations:  63
Stopping condition: 1e-08
```



(a) Snapshot at $t = 1$ (b) Snapshot at $t = 10$ (c) Snapshot at $t = 100$

(d) Degree distribution at $t = 1$ (e) Degree distribution at $t = 10$ (f) Degree distribution at $t = 100$

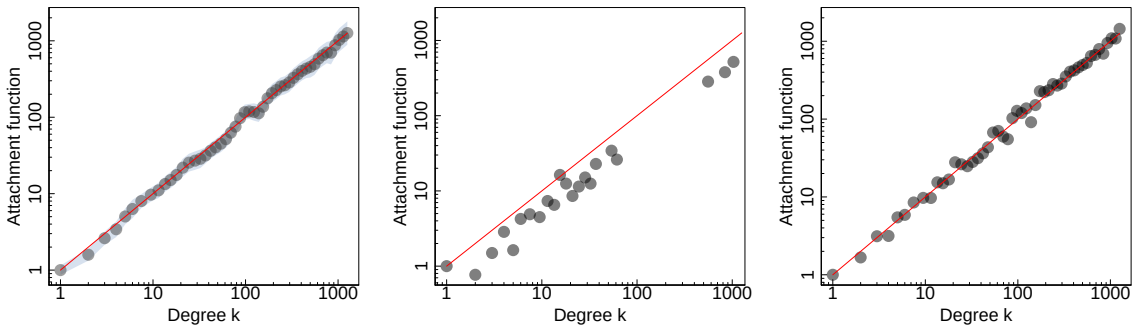
Figure 1: Network snapshots and their corresponding in-degree distributions at time-steps $t = 1$, 10 and 100.

As stated in Section 2, the PAFit method first finds the r parameter, which regularizes the PA function, by cross-validation, and then estimates A_k using the chosen r. The estimated function can be accessed via `$estimate_result$k` and `$estimate_result$A` of `result_PA_only`. From this estimated function, the attachment exponent α (when we assume $A_k = k^\alpha$) and its two-sigma confidence interval are also estimated. Here $\hat{\alpha}$ is 1.001 ± 0.01 as we can see from the output of `summary`. These values can be accessed via `$estimate_result$alpha` and `$estimate_result$ci`.

The output also reveals that by default **PAFit** applies binning with 50 bins. In this procedure, we divide the range of k into bins consisting of consecutive degrees, and assume that all k in a bin have the same value of A_k. Binning is an important regularization technique that significantly stabilizes the estimation of the attachment function (Pham *et al.* 2015). In this example, the center of each bin is stored in the field `$center_k` of `stats_1`.

Since the center of a bin is also the PA value corresponding to that bin in the linear PA case, we can plot the estimated attachment function together with the true attachment function using the following script, which produces Figure 2a. The options `min_A` and `max_A` specify the minimum and maximum values in the vertical axis of the plot, respectively.

```
R> plot(result_PA_only, stats_1, min_A = 1, max_A = 2000)
```



(a) PAFit ($\hat{\alpha} = 1.001 \pm 0.01$)

(b) Jeong's method ($\hat{\alpha} = 0.96 \pm 0.07$)

(c) Newman's method ($\hat{\alpha} = 1.01 \pm 0.02$)

Figure 2: Estimating the attachment function in isolation. The true attachment function is $A_k = k^\alpha$ with attachment exponent $\alpha = 1$. We also show the estimated α and its two-sigma confidence interval provided by each method.

```
R> lines(stats_1$center_k, stats_1$center_k, col = "red")
```

The estimation results of Jeong's method and Newman's method can be plotted in a similar way, and are shown in Figures 2b and 2c, respectively.

Overall, Newman's method and PAFit estimate the attachment function A_k about equally well, while Jeong's method is found to underestimate the function and also exhibits high variance. This can also be observed in the estimated attachment exponent of the three methods: Newman's method and PAFit recover the true α, while Jeong's method underestimates it. Note that in PAFit we also have the two-sigma confidence intervals (lightblue region in Figure 2a) of the estimated A_k, which are unavailable in the other two methods. This is a significant advantage of PAFit over the other two methods since it allows the user to quantify uncertainties in the result.

5.2. Node fitnesses estimation

Here we estimate node fitnesses from a BB model generated network with the assumption that $A_k = k$. To demonstrate the functionality of the package, we generate a BB network with a nonstandard setting:

```
R> sim_net_2 <- generate_BB(N = 1000, num_seed = 100, multiple_node = 100,
                            m = 15, s = 10)
```

This network grows from a seed network with $N_0 = 100$ nodes where the nodes form a line graph. The value of N_0 can be specified by `num_seed`. At each time-step we add $n = 100$ new nodes where each node has $m = 15$ new edges. The values of n and m can be specified via `multiple_node` and `m`, respectively. The total number of nodes in the final network is $N = 1000$. Finally, the distribution from which we generate node fitnesses is the Gamma distribution with mean 1 and inverse variance $s = 10$.

Next we get the network summary statistics and then apply the estimation function:

```
R> stats_2         <- get_statistics(sim_net_2)
R> result_fit_only <- only_F_estimate(sim_net_2, stats_2)
R> plot(result_fit_only, stats_2, plot = "f")
```

The final line of the snippet generates the distribution of estimated node fitnesses in Figure 3a.

In its default setting, the function `only_F_estimate` estimates node fitnesses under the assumption that $A_k = k$. But one also can estimate node fitnesses in the Caldarelli model, i.e., assuming $A_k = 1$ for all k, with the option `model_A = "Constant"`. The function `only_F_estimate` works by first finding the estimated value $\hat{s}$ of s by cross-validation, and then using $\hat{s}$ in the subsequent estimation of node fitnesses. The summary information of the estimation result can be viewed by invoking `summary`:

```
R> summary(result_fit_only)

Estimation results by the PAFit method.
Mode: Only node fitnesses were estimated.
Estimated s parameter: 8
-----------------------------------------
Additional information:
Number of bins:  50
Number of iterations:  19
Stopping condition: 0.00000001
```

The method slightly under-estimated s. We can check whether the node fitnesses were estimated well by plotting the estimated fitnesses versus the true fitnesses by running the following command:

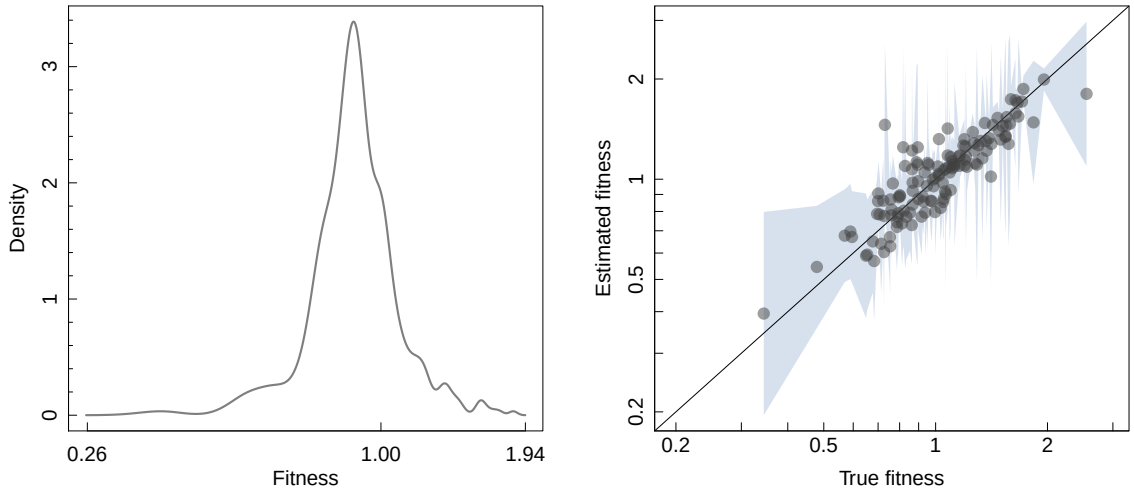
```
R> plot(result_fit_only, stats_2, true_f = sim_net_2$fitness, plot = "true_f")
```

This will produce the plot of Figure 3b. It turns out that the estimated node fitnesses agree pretty well with the true node fitnesses. We note that the light blue band around the $\hat{\eta}_i$ values depicts the two-sigma confidence intervals of the estimated values. The upper and lower values can be accessed via `$estimate_result$upper_f` and `$estimate_result$lower_f` of `result_fit_only`, respectively.

5.3. Joint estimation of the attachment function and node fitnesses

Here we show how to estimate the attachment function and node fitnesses simultaneously. We need to assume in Section 5.1 the equality of all η_i for the estimation of A_k in isolation, and in Section 5.2 a specific functional form of A_k for the estimation of η_i in isolation. Such assumptions become unnecessary when we perform joint estimation, since the appropriate functional forms will be automatically enforced through the regularization parameters r and s, which will be chosen by cross-validation. We recommend the joint estimation procedure as the standard estimation procedure in this package, unless there is a specific reason to justify the one or the other of these assumptions.

This time, we generate a network in which the attachment function is $A_k = k^\alpha$ with $\alpha = 0.5$ and the Gamma distribution of node fitnesses has mean 1 and variance $1/s$ with $s = 10$:



(a) Distribution of estimated fitnesses. (b) Estimated fitnesses versus true fitnesses.

Figure 3: Estimating node fitnesses in isolation. The attachment function is fixed at $A_k = k$. In panel b, we only plot nodes for which the number of acquired new edges is at least 5.

```
R> sim_net_3   <- generate_net(N = 1000, num_seed = 100, multiple_node = 100,
                               m = 15, s = 10, alpha = 0.5)
```

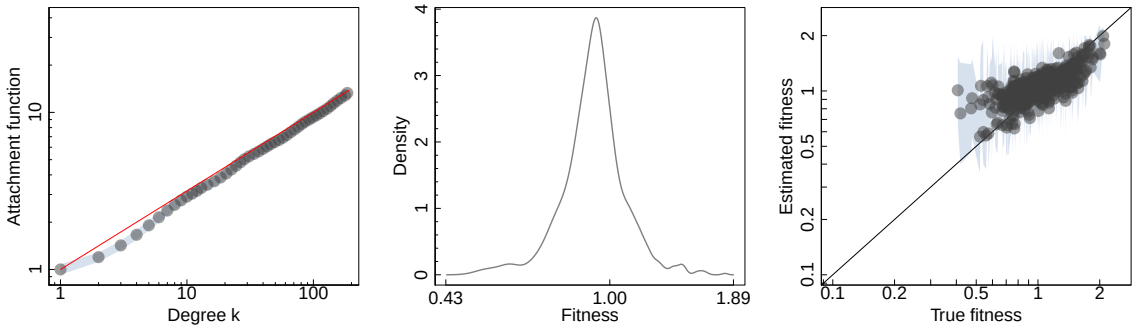
We then apply `joint_estimation`:

```
R> stats_3      <- get_statistics(sim_net_3)
R> result_PAFit <- joint_estimate(sim_net_3, stats_3)
R> summary(result_PAFit)

Estimation results by the PAFit method.
Mode: Both the attachment function and node fitness were estimated.
Estimated r parameter:  10
Estimated s parameter: 18.75
Estimated attachment exponent:  0.5168941
Two-sigma confidence interval of the attachment exponent: ( 0.5097277 , 0.5240605 )
-----------------------------------------
Additional information:
Number of bins:  50
Number of iterations:  596
Stopping condition: 0.00000001
```

We can plot the estimated attachment function as in Figure 4a, and the distribution of the $\hat{\eta}_i$'s as in Figure 4b with the following code:

```
R> plot(result_PAFit, stats_3)
R> lines(stats_3$center_k, stats_3$center_k^0.5, col = "red")
R> plot(result_PAFit, stats_3, plot = "f")
```



(a) Attachment function ($\hat{\alpha} = 0.52 \pm 0.01$)

(b) Distribution of estimated fitnesses

(c) Estimated and true η_i

Figure 4: Joint estimation of the attachment function and node fitnesses. The red line in panel a is the true attachment function $A_k = k^{0.5}$.

Concerning the estimated values, while s is slightly over-estimated by $\hat{s} = 18.75$, $\hat{\alpha} = 0.52 \pm 0.01$ is a good estimate of α. We can also plot the estimated fitnesses versus the true fitnesses as in Figure 4c with the following code:

```
R> plot(result_PAFit, stats_3, true_f = sim_net_3$fitness, plot = "true_f")
```

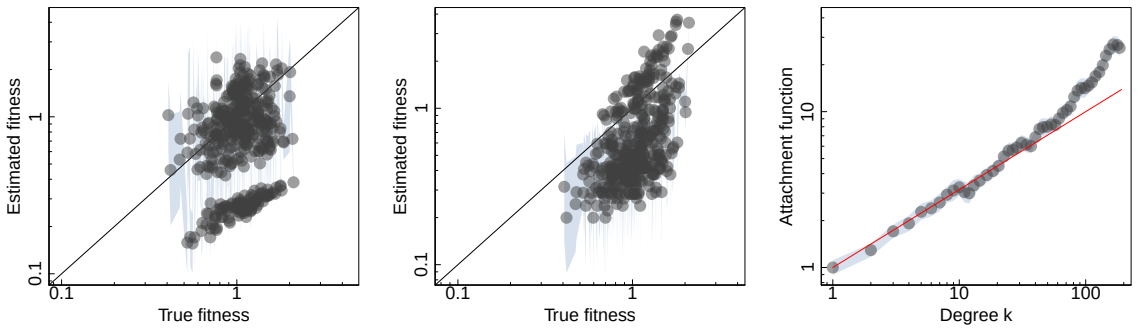
Since the mean of $\hat{\eta}_i$'s is normalized to one, the over-estimation of s leads to over-estimation of low-value fitnesses and under-estimation of high-value fitness, as can be seen in Figure 4c.

We show how joint estimation improves on estimating either node fitnesses in isolation (Figures 5a and 5b) or the PA function in isolation (Figure 5c). For estimating node fitnesses in isolation, two cases are shown: the result when we assume the BB model in which $A_k = k$ (Figure 5a) and the result when we assume the Caldarelli model in which $A_k = 1$ (Figure 5b). In either case, the estimated node fitnesses are visually worse than those of the joint estimation in Figure 4c. Similarly, estimating the PA function in isolation apparently led to overestimation of the PA function in the region of large k. To conclude, estimating either node fitnesses or the PA function in isolation would likely be worse than the joint estimation, if the underlying assumptions about the true node fitnesses and the true PA function are wrong.

6. Simulation Study

In this section, we present the results of a simulation that we conducted to assess the performance of `joint_estimation` function. We assume the functional form $A_k = k^\alpha$ for the attachment function. To cover the spectrum of PA and anti-PA phenomena, we choose four values for α: -0.5, 0, 0.5, and 1. We sample node fitnesses from a Gamma distribution with mean 1 and variance $1/s$. Three values for s are chosen: 5, 20, and 80. While small values of s lead to widely varied node fitnesses, large values of s leads to highly concentrated node fitnesses.

For each combination of α and s, we generated $M = 50$ networks, estimated A_k and s from each network using `joint_estimation`. We then fit the form $\hat{A}_k = k^\alpha$ to $\hat{A}_k$ in order to



(a) Estimated and true η_i (assuming $A_k = k$). $\hat{s} = 3.2$.

(b) Estimated and true η_i (assuming $A_k = 1$). $\hat{s} = 4$.

(c) Estimated PA function in isolation. $\hat{\alpha} = 0.60 \pm 0.05$.

Figure 5: Estimating either node fitnesses or the PA function in isolation. The red line in panel c is the true attachment function $A_k = k^{0.5}$.

estimate α. We then compared the means of the estimation results of α and s with the true values. Each simulated network has a total of 1000 nodes with the initial graph has 200 nodes and 50 new nodes are added at each time-step for a total of 10 time-steps. Each new node has 50 new edges.

The results are shown in Figure 6. The attachment exponent α was estimated reasonably well across all combinations of α and s. Except for the cases in which the attachment function grows fast ($\alpha = 1$) or the case in which node fitnesses have high variance ($s = 5$), the estimated values of s were also acceptable. When $s = 5$, it was slightly over-estimated, which perhaps is due to the high variance of the node fitnesses in this case. We also notice that s was slightly over-estimated when $\alpha = 1$, which may be caused by the fast growing rate of the PA function. One also notices that the confidence intervals for $\hat{s}$ are much larger than those for $\hat{\alpha}$. The above observations imply that it is much harder to estimate s than α.

7. Analysis of a collaboration network between scientists

In this section, we demonstrate the complete workflow for the joint estimation of A_k and η_i on a collaboration network between scientists from the field of complex networks. In this network, nodes are scientists and an undirected edge exists between them if and only if they have jointly written a paper. The degree of a node represents the number of collaborators of a scientist, since multiple edges are not considered. The temporal network is stored in `coauthor.net`, and the names of the scientists are stored in `coauthor.author_id`. The network without timestamps was compiled by Mark Newman from the bibliographies of two review articles on complex networks (Newman 2006). Paul Sheridan, the second author of the present work, augmented the dataset with timestamps. More information on the dataset can be found in the package reference manual.

The first step in the analysis is to convert the edgelist matrix `coauthor.net` to a `PAFit_net` object, and get the summary statistics using the function `get_statistics`.

```
R> set.seed(1)
```

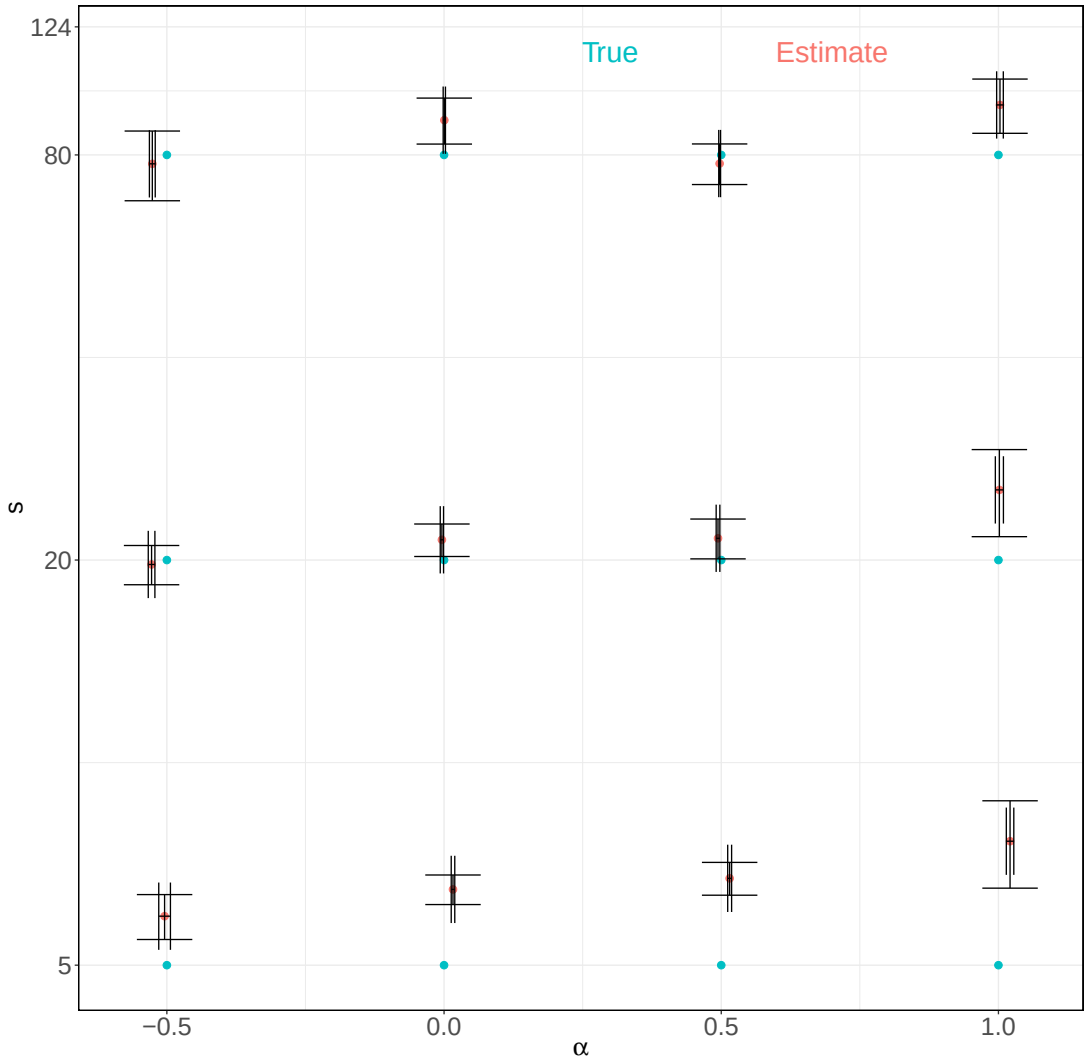


Figure 6: Simulation result. At each point, the horizon bar indicates the two-sigma interval for the estimated α, while the vertical bar indicates the two-sigma interval for the estimated s.

```
R> true_net  <- as.PAFit_net(coauthor.net, type = "undirected")
R> net_stats <- get_statistics(true_net)
R> summary(net_stats)

Contains summary statistics for the temporal network.
Type of network:  undirected
Number of nodes in the final network:  1498
Number of edges in the final network:  5698
Number of new nodes:  1358
Number of new edges:  1255
Number of time-steps: 145
Maximum degree:  37
Number of bins: 38
```

The temporal network grew in 145 time-steps from an initial network at September 2000, to a final state at September 2007. The resolution of those time-steps is monthly. The final network has 1498 scientists with 5698 collaborations among them.

The next step is to use `joint_estimate` for joint estimation:

```
R> full_result <- joint_estimate(true_net, net_stats)
R> summary(full_result)

Estimation results by the PAFit method.
Mode: Both the attachment function and node fitness were estimated.
Estimated r parameter:  10
Estimated s parameter: 45
Estimated attachment exponent:  0.9951764
Two-sigma confidence interval of the attachment exponent: ( 0.9715202 , 1.018833 )
-----------------------------------------
Additional information:
Number of bins:  38
Number of iterations:  607
Stopping condition: 0.00000001
```

We can visualize the estimated attachment function and the distribution of estimated node fitnesses by:

```
R> plot(full_result, net_stats, plot = "A")
R> plot(full_result, net_stats, plot = "f")
```

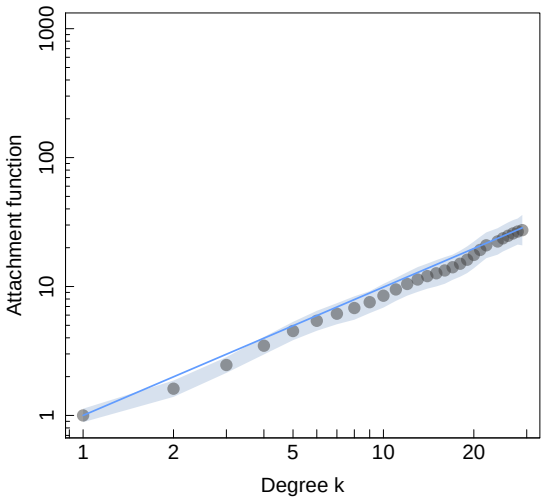
This snippet will sequentially generate Figures 7a and 7b.

The best fit model when we performed joint estimation is close to the BB model. In Figure 7a, the estimated A_k is an increasing function on average with $\hat{\alpha} = 1.00 \pm 0.05$. We can conclude that linear PA likely exists in this collaboration network. Let us take a concrete example: a network scientist with twenty collaborators has roughly twice the chance to get a new collaborator compared with someone who only has ten collaborators, assuming they have the same fitness. For comparison's sake, we also plot the estimation results of A_k in isolation using Jeong's method, Newman's method, and PAFit in Figures 7c, 7d, and 7e, respectively:

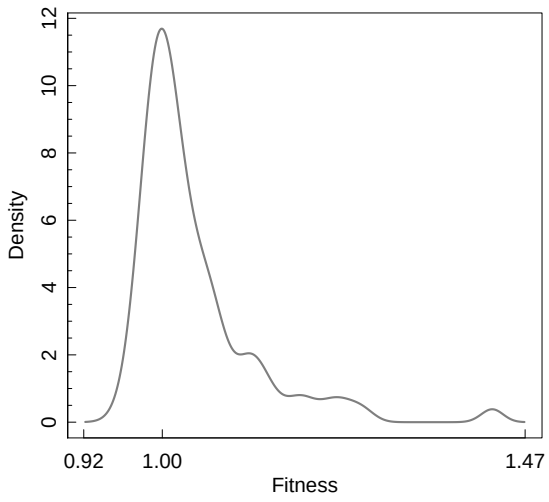
```
R> result_Jeong  <- Jeong(true_net, net_stats)
R> result_Newman <- Newman(true_net, net_stats)
R> result_onlyA  <- only_A_estimate(true_net, net_stats)
R> plot(result_Jeong, net_stats, plot = "A", min_A = 1, max_A = 1000)
R> plot(result_Newman, net_stats, plot = "A", min_A = 1, max_A = 1000)
R> plot(result_onlyA, net_stats, plot = "A", min_A = 1, max_A = 1000)
```

The options `min_A = 1` and `max_A = 1000` specify the range of the vertical axis and are needed for making the plots comparable.

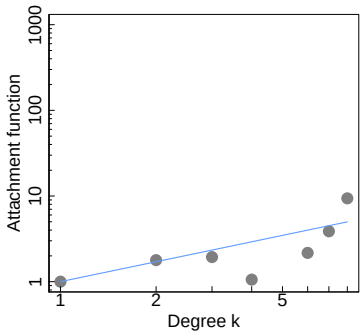
The high variance of $\hat{\alpha}$ from either Jeong's method or Newman's method would render qualitative assessments of the PA function inconclusive, if one relied only on those methods: one could not confidently ascertain which region the PA function belongs to: sub-linear, linear,



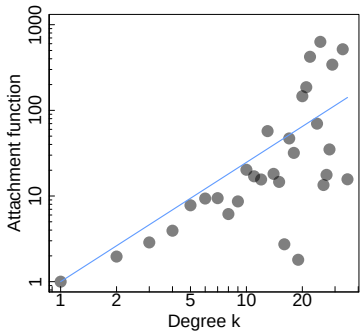
(a) Estimated A_k when joint estimation by PAFit ($\hat{\alpha} = 1.00 \pm 0.05$)



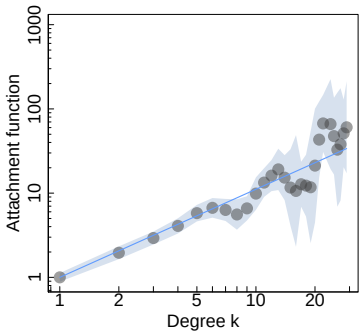
(b) Histogram of estimated node fitnesses



(c) Jeong's method ($\hat{\alpha} = 0.77 \pm 0.80$)



(d) Newman's method ($\hat{\alpha} = 1.39 \pm 0.56$)



(e) PAFit ($\hat{\alpha} = 1.05 \pm 0.07$)

Figure 7: Estimation of the attachment function and node fitnesses in the collaboration network of scientists in the complex network field. Panels a and b show the joint estimation result, while panels c, d and e show the results when we estimated the PA function in isolation.

or super-linear. We notice that the estimated A_k of the joint estimation resembles that of Figure 7e, when we estimate it in isolation. The reason is that estimated node fitnesses in Figure 7b are highly concentrated around the mean. Thus their distribution is not very far from the case when all the fitnesses are 1. Nevertheless, we observe that the estimated A_k from the joint estimation is reduced when compared with that of Figure 7e. This is expected since in the joint estimation, a portion of the connection probability in Eq. (1) is explained by node fitness.

Although the distribution in Figure 7b is concentrated around its mean, we notice that its right tail is rather long, which is a sign that this tail contains interesting information. We can extract the information from this region by finding the 'fittest' network scientists. This

can be done as follows:

```
R> sorted_fit    <- sort(full_result$estimate_result$f, decreasing = TRUE)
R> top_scientist <- coauthor.author_id[names(sorted_fit),]
R> print(cbind(sorted_fit[1:10],top_scientist[1:10,2]))
```

This snippet will produce the results show in Table 5. The table shows the top ten scientists that have the highest ability to attract new collaborators in the field of complex networks. If

Rank	Estimated fitness	Name
1	1.42	BARABASI, A
2	1.35	NEWMAN, M
3	1.26	JEONG, H
4	1.25	LATORA, V
5	1.24	ALON, U
6	1.23	OLTVAI, Z
7	1.23	YOUNG, M
8	1.22	WANG, B
9	1.21	SOLE, R
10	1.21	BOCCALETTI, S

Table 5: The top ten 'fittest' scientists in the field of complex networks.

one has some familiarity with the field, it is easy to recognize the names of many big-shots in the list. For example, at the top of the list is none other than Albert-László Barabási, who introduced the BA model. Number two and number three are Mark Newman and Hawoong Jeong, who respectively are the authors of the eponymously named Newman's method and Jeong's method.

8. Conclusion

We introduced the R package **PAFit**, which provides a comprehensive framework for the non-parametric estimation of PA and node fitness mechanisms in the growth of temporal complex networks. In summary, **PAFit** implements functions to simulate various temporal network models based on these two mechanisms, gathers summary statistics from real-world temporal network datasets, and estimates non-parametrically the attachment function and node fitnesses. We provided a number of simulated examples, as well as a complete analysis of a real-world collaboration network.

Acknowledgments

This work was supported in part by grants from the Japan Society for the Promotion of Science KAKENHI [JP16J03918 to T.P. and 16H01547 to H.S.].

References

Akaike H (1974). "A new look at the statistical model identification." *IEEE Transactions on Automatic Control*, **19**(6), 716–723. ISSN 0018-9286. doi:10.1109/TAC.1974.1100705.

Albert R, Barabási A (1999). "Emergence of Scaling in Random Networks." *Science*, **286**, 509–512.

Albert R, Jeong H, Barabasi AL (2000). "Error and Attack Tolerance of Complex Networks." *Nature*, **406**(6794), 378–382. URL http://dx.doi.org/10.1038/35019019.

Allison PD (1980). "Estimation and Testing for a Markov Model of Reinforcement." *Sociological Methods & Research*, **8**(4), 434–453. doi:10.1177/004912418000800405. https://doi.org/10.1177/004912418000800405, URL https://doi.org/10.1177/004912418000800405.

Barabási AL, Albert R, Jeong H (2000). "Scale-free characteristics of random networks: the topology of the world-wide web." *Physica A: Statistical Mechanics and its Applications*, **281**, 69 – 77. ISSN 0378-4371. doi:http://doi.org/10.1016/S0378-4371(00)00018-2. URL http://www.sciencedirect.com/science/article/pii/S0378437100000182.

Bender-deMoll S, Morris M (2016). ***tsna**: Tools for Temporal Social Network Analysis*. R package version 0.2.0, URL https://CRAN.R-project.org/package=tsna.

Bezáková I, Kalai A, Santhanam R (2006). "Graph Model Selection Using Maximum Likelihood." In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pp. 105–112. ACM, New York, NY, USA. ISBN 1-59593-383-2. doi:10.1145/1143844.1143858. URL http://doi.acm.org/10.1145/1143844.1143858.

Bianconni G, Barabási A (2001). "Competition and Multiscaling in Evolving Networks." *Europhysics Letters*, **54**, 436.

Borgs C, Chayes J, Daskalakis C, Roch S (2007). "First to Market is not Everything: an Analysis of Preferential Attachment with Fitness." In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*.

Box-Steffensmeier JM, De Boef S (2006). "Repeated events survival models: the conditional frailty model." *Statistics in Medicine*, **25**(20), 3518–3533. ISSN 1097-0258. doi:10.1002/sim.2434. URL http://dx.doi.org/10.1002/sim.2434.

Butts CT (2016). ***sna**: Tools for Social Network Analysis*. R package version 2.4, URL https://CRAN.R-project.org/package=sna.

Butts CT, Leslie-Cook A, Krivitsky PN, Bender-deMoll S (2016). ***networkDynamic**: Dynamic Extensions for Network Objects*. R package version 0.9.0, URL https://CRAN.R-project.org/package=networkDynamic.

Caldarelli G (2007). *Scale-Free Networks*. Oxford Universiy Press.

Caldarelli G, Capocci A, De Los Rios P, Muñoz MA (2002). "Scale-Free Networks from Varying Vertex Intrinsic Fitness." *Physical Review Letters*, **89**, 258702. doi:10.1103/PhysRevLett.89.258702. URL http://link.aps.org/doi/10.1103/PhysRevLett.89.258702.

Callaway DS, Hopcroft JE, Kleinberg JM, Newman MEJ, Strogatz SH (2001). "Are Randomly Grown Graphs Really Random?" *Physical Review E*, **64**, 041902. doi:10.1103/PhysRevE.64.041902. URL http://link.aps.org/doi/10.1103/PhysRevE.64.041902.

Clauset A, Shalizi CR, Newman MEJ (2009). "Power-Law Distributions in Empirical Data." *SIAM Review*, **51**(4), 661–703. doi:10.1137/070710111. http://dx.doi.org/10.1137/070710111, URL http://dx.doi.org/10.1137/070710111.

Coleman JS (1964). *Introduction to mathematical sociology*. Free Press of Glencoe London.

Csardi G, Nepusz T (2006). "The **igraph** Software Package for Complex Network Research." *InterJournal*, **Complex Systems**, 1695. URL http://igraph.org.

Dorogovtsev SN, Mendes JFF (2003). *Evolution of Networks: From Biological Nets to the Internet and WWW (Physics)*. Oxford University Press, Inc., New York, NY, USA. ISBN 0198515901.

Dunne JA, Williams RJ, Martinez ND (2002). "Food-web Structure and Network Theory: the Role of Connectance and Size." *Proceedings of the National Academy of Sciences*, **99**(20), 12917–12922. doi:10.1073/pnas.192407699. http://www.pnas.org/content/99/20/12917.full.pdf, URL http://www.pnas.org/content/99/20/12917.abstract.

Eddelbuettel D (2013). *Seamless R and C++ Integration with **Rcpp***. Springer-Verlag, New York. ISBN 978-1-4614-6867-7.

Eddelbuettel D, François R (2011). "**Rcpp**: Seamless R and C++ Integration." *Journal of Statistical Software*, **40**(8), 1–18. URL http://www.jstatsoft.org/v40/i08/.

Eom YH, Jo HH (2014). "Generalized Friendship Paradox in Complex Networks: the Case of Scientific Collaboration." *Scientific Reports*, **4**. doi:10.1038/srep04603. URL http://dx.doi.org/10.1038/srep04603.

Erdös P, Rényi A (1959). "On Random Graphs." *Publicationes Mathematicae Debrecen*, **6**, 290–297.

Feld SL (1991). "Why Your Friends Have More Friends Than You Do." *American Journal of Sociology*, **96**(6), 1464–1477. doi:10.1086/229693. http://dx.doi.org/10.1086/229693, URL http://dx.doi.org/10.1086/229693.

Guetz AN, Holmes SP (2011). "Adaptive Importance Sampling for Network Growth Models." *Annals of Operations Research*, **189**(1), 187–203. ISSN 1572-9338. doi:10.1007/s10479-010-0685-2. URL http://dx.doi.org/10.1007/s10479-010-0685-2.

Handcock MS, Hunter DR, Butts CT, Goodreau SM, Krivitsky PN, Bender-deMoll S, Morris M (2016). ***statnet**: Software Tools for the Statistical Analysis of Network Data*. The Statnet Project (http://www.statnet.org). R package version 2016.9, URL CRAN.R-project.org/package=statnet.

Handcock MS, Hunter DR, Butts CT, Goodreau SM, Krivitsky PN, Morris M (2017). ***ergm**: Fit, Simulate and Diagnose Exponential-Family Models for Networks*. The Statnet Project (http://www.statnet.org). R package version 3.8.0, URL https://CRAN.R-project.org/package=ergm.

Handcock MS, Hunter DR, Butts CT, Goodreau SM, Morris M (2008). "**statnet**: Software Tools for the Representation, Visualization, Analysis and Simulation of Network Data." *Journal of Statistical Software*, **24**(1), 1–11. URL http://www.jstatsoft.org/v24/i01.

Handcock MS, Jones JH (2004). "Likelihood-based inference for stochastic models of sexual network formation." *Theoretical Population Biology*, **65**(4), 413 – 422. ISSN 0040-5809. doi:https://doi.org/10.1016/j.tpb.2003.09.006. Demography in the 21st Century, URL http://www.sciencedirect.com/science/article/pii/S0040580904000310.

Hunter D, Lange K (2000). "Quantile Regression via an MM Algorithm." *Journal of Computational and Graphical Statistics*, pp. 60–77.

Hunter D, Lange K (2004). "A Tutorial on MM Algorithms." *The American Statistician*, **58**, 30–37.

Hunter DR, Handcock MS, Butts CT, Goodreau SM, Morris M (2008). "**ergm**: A Package to Fit, Simulate and Diagnose Exponential-Family Models for Networks." *Journal of Statistical Software*, **24**(3), 1–29.

INRA, Leger JB (2015). ***blockmodels**: Latent and Stochastic Block Model Estimation by a 'V-EM' Algorithm*. R package version 1.1.1, URL https://CRAN.R-project.org/package=blockmodels.

Irwin JO (1963). "The Place of Mathematics in Medical and Biological Statistics." *Journal of the Royal Statistical Society. Series A (General)*, **126**(1), 1–45. ISSN 00359238. URL http://www.jstor.org/stable/2982445.

Jeong H, Néda Z, Barabási A (2003). "Measuring Preferential Attachment in Evolving Networks." *Europhysics Letters*, **61**(61), 567–572.

Kelly PJ, Lim LLY (2000). "Survival analysis for recurrent event data: an application to childhood infectious diseases." *Statistics in Medicine*, **19**(1), 13–33. ISSN 1097-0258. doi:10.1002/(SICI)1097-0258(20000115)19:1<13::AID-SIM279>3.0.CO;2-5. URL http://dx.doi.org/10.1002/(SICI)1097-0258(20000115)19:1<13::AID-SIM279>3.0.CO;2-5.

Kong J, Sarshar N, Roychowdhury V (2008). "Experience versus Talent Shapes the Structure of the Web." *Proceedings of the National Academy of Sciences of the USA*, **37**, 105.

Krapivsky P, Rodgers G, Redner S (2001). "Organization of Growing Networks." *Physical Review E*, p. 066123.

Krivitsky PN, Handcock MS (2008). "Fitting position latent cluster models for social networks with latentnet." *Journal of Statistical Software*, **24**(5).

Krivitsky PN, Handcock MS (2016). ***tergm**: Fit, Simulate and Diagnose Models for Network Evolution Based on Exponential-Family Random Graph Models*. The Statnet Project (http://www.statnet.org). R package version 3.4.0, URL http://CRAN.R-project.org/package=tergm.

Krivitsky PN, Handcock MS (2017). ***latentnet**: Latent Position and Cluster Models for Statistical Networks*. The Statnet Project (http://www.statnet.org). R package version 2.8.0, URL https://CRAN.R-project.org/package=latentnet.

Kunegis J (2013). "KONECT – The Koblenz Network Collection." konect.uni-koblenz.de. URL http://konect.uni-koblenz.de/.

Kunegis J, Blattner M, Moser C (2013). "Preferential Attachment in Online Networks: Measurement and Explanations." In *Proceedings of the 5th Annual ACM Web Science Conference*, WebSci '13, pp. 205–214. ACM, New York, NY, USA. ISBN 978-1-4503-1889-1. doi:10.1145/2464464.2464514. URL http://doi.acm.org/10.1145/2464464.2464514.

Leifeld P, Cranmer S, Desmarais B (2018). "Temporal Exponential Random Graph Models with **btergm**: Estimation and Bootstrap Confidence Intervals." *Journal of Statistical Software, Articles*, **83**(6), 1–36. ISSN 1548-7660. doi:10.18637/jss.v083.i06. URL https://www.jstatsoft.org/v083/i06.

Leskovec J, Krevl A (2014). "SNAP Datasets: Stanford Large Network Dataset Collection." http://snap.stanford.edu/data.

Liljeros F, Edling CR, Amaral LAN, Stanley HE, Aberg Y (2001). "The Web of Human Sexual Contacts." *Nature*, **411**(6840), 907–908. URL http://dx.doi.org/10.1038/35082140.

Matias C, Miele V (2016). "Statistical clustering of temporal networks through a dynamic stochastic block model." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **79**(4), 1119–1141. doi:10.1111/rssb.12200. https://rss.onlinelibrary.wiley.com/doi/pdf/10.1111/rssb.12200, URL https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/rssb.12200.

Matias C, Miele V (2018). ***dynsbm**: Dynamic Stochastic Block Models*. R package version 0.5, URL https://CRAN.R-project.org/package=dynsbm.

Momeni N, Rabbat MG (2015). *Measuring the Generalized Friendship Paradox in Networks with Quality-Dependent Connectivity*, pp. 45–55. Springer-Verlag International Publishing, Cham. ISBN 978-3-319-16112-9. doi:10.1007/978-3-319-16112-9_5. URL http://dx.doi.org/10.1007/978-3-319-16112-9_5.

Nekovee M, Moreno Y, Bianconi G, Marsili M (2007). "Theory of Rumour Spreading in Complex Social Networks." *Physica A: Statistical Mechanics and its Applications*, **374**(1), 457 – 470. ISSN 0378-4371. doi:http://doi.org/10.1016/j.physa.2006.07.017. URL http://www.sciencedirect.com/science/article/pii/S0378437106008090.

Newman M (2001). "Clustering and Preferential Attachment in Growing Networks." *Physical Review E*, **64**(2), 025102.

Newman M (2006). "Finding Community Structure in Networks Using the Eigenvectors of Matrices." *Physical Review E*, **74**, 036104. doi:10.1103/PhysRevE.74.036104. URL https://link.aps.org/doi/10.1103/PhysRevE.74.036104.

Newman M (2010). *Networks: an Introduction*. Oxford University Press, Inc., New York, NY, USA. ISBN 0199206651, 9780199206650.

Newman M, Forrest S, Balthrop J (2002). "Email Networks and the Spread of Computer Viruses." *Physical Review E*, **66**, 035101. doi:10.1103/PhysRevE.66.035101. URL https://link.aps.org/doi/10.1103/PhysRevE.66.035101.

Pastor-Satorras R, Vespignani A (2001). "Epidemic Spreading in Scale-Free Networks." *Physical Review Letters*, **86**, 3200–3203. doi:10.1103/PhysRevLett.86.3200. URL https://link.aps.org/doi/10.1103/PhysRevLett.86.3200.

Pham T, Sheridan P, Shimodaira H (2015). "PAFit: a Statistical Method for Measuring Preferential Attachment in Temporal Complex Networks." *PLOS ONE*, **10**(9), e0137796. doi:10.1371/journal.pone.0137796. URL http://dx.doi.org/10.1371/journal.pone.0137796.

Pham T, Sheridan P, Shimodaira H (2016). "Joint Estimation of Preferential Attachment and Node Fitness in Growing Complex Networks." *Scientific Reports*, **6**. doi:10.1038/srep32558. URL http://dx.doi.org/10.1038/srep32558.

Pham T, Sheridan P, Shimodaira H (2017). ***PAFit**: Generative Mechanism Estimation in Temporal Complex Networks*. R package version 0.9.9.0, URL cran.r-project.org/package=PAFit.

Price DdS (1976). "A General Theory of Bibliometric and other Cumulative Advantage Processes." *Journal of the American Society for Information Science*, **27**, 292–306.

Raftery AE (1995). "Bayesian Model Selection in Social Research." *Sociological Methodology*, **25**, 111–163. ISSN 00811750, 14679531. URL http://www.jstor.org/stable/271063.

Redner S (2005). "Citation Statistics from 110 Years of Physical Review." *Physics Today*, **58**(6), 49–54. doi:10.1063/1.1996475. physics/0506056.

Ripley R, Boitmanis K, Snijders TA (2013). ***RSiena**: Siena - Simulation Investigation for Empirical Network Analysis*. R package version 1.1-232, URL https://CRAN.R-project.org/package=RSiena.

Ripley RM, Snijders TA, Boda Z, VÃűrÃűs A, Preciado P (2018). *Manual for SIENA version 4.0*. R package version 1.1.1, URL http://www.stats.ox.ac.uk/~snijders/siena/.

Schweinberger M, Handcock M, Luna P (2018). ***hergm**: Hierarchical Exponential-Family Random Graph Models with Local Dependence*. R package version 3.1-1, URL https://CRAN.R-project.org/package=hergm.

Sheridan P, Kamimura T, Shimodaira H (2010). "A Scale-Free Structure Prior for Graphical Models with Applications in Functional Genomics." *PLoS ONE*, **5**(11), e13580. doi:10.1371/journal.pone.0013580. URL http://dx.doi.org/10.1371%2Fjournal.pone.0013580.

Sheridan P, Onodera T (2018). "A Preferential Attachment Paradox: How Preferential Attachment Combines with Growth to Produce Networks with Log-normal In-degree Distributions." *Scientific Reports*, **8**(1), 2811. ISSN 2045-2322. doi:10.1038/s41598-018-21133-2. URL https://doi.org/10.1038/s41598-018-21133-2.

Simon HA (1955). "On a Class of Skew Distribution Functions." *Biometrika*, **42**(3-4), 425–440. doi:10.1093/biomet/42.3-4.425. http://biomet.oxfordjournals.org/content/42/3-4/425.full.pdf+html, URL http://biomet.oxfordjournals.org/content/42/3-4/425.short.

Sinatra R, Wang D, Deville P, Song C, Barabási AL (2016). "Quantifying the Evolution of Individual Scientific Impact." *Science*, **354**(6312). ISSN 0036-8075. doi:10.1126/science.aaf5239. http://science.sciencemag.org/content/354/6312/aaf5239.full.pdf, URL http://science.sciencemag.org/content/354/6312/aaf5239.

Wang D, Song C, Barabási AL (2013). "Quantifying Long-Term Scientific Impact." *Science*, **342**(6154), 127–132. ISSN 0036-8075. doi:10.1126/science.1237825. http://science.sciencemag.org/content/342/6154/127.full.pdf, URL http://science.sciencemag.org/content/342/6154/127.

Yule GU (1925). "A Mathematical Theory of Evolution, Based on the Conclusions of Dr. J.C. Willis,F.R.S." *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, **213**(402-410), 21–87. ISSN 0264-3960. doi:10.1098/rstb.1925.0002.

Zhou H, Alexander D, Lange K (2011). "A quasi-Newton acceleration for high-dimensional optimization algorithms." *Statistics and Computing*, **21**, 261–273.

Affiliation:

Thong Pham
Mathematical Statistics Team
Generic Technology Group, Center for Advanced Intelligence Project, RIKEN
Nihonbashi 1-chome Mitsui Building, 15th floor, 1-4-1 Nihonbashi, Chuo-ku, Tokyo
E-mail: thong.pham@riken.jp