A Social Network Perspective on the Success of Open Source Software: the Case of R Packages

Gianluca Zanella, Charles Z. Liu, Kim-Kwang Raymond Choo
The University of Texas at San Antonio
gianluca.zanella@utsa.edu, charles.liu@utsa.edu, raymond.choo@utsa.edu

Abstract

In this paper, we seek to identify the factors that influence the impact of open source software (OSS) on users community through the analysis of the evolution of the OSS network. Based on longitudinal data collected from the comprehensive R archive network (CRAN), we empirically examine how the network of R packages evolves over time and exert its influence on the scientific community. We find that critical network features derived from CRAN, such as page-rank, closeness, and betweenness centralities, play a significant role in determining the impact of each package on the research and publication activities in the scientific community. Furthermore, the performance of R packages can be explained as a flow of information from the core to the periphery that exhibits strong spillover effects.

1. Introduction

"*If I have seen further than others, it is by standing upon the shoulders of giants*" (Isaac Newton). This is probably the best way to explain the crucial role of an open source software (OSS) network on providing support to the scientific community. OSS is a type of computer software released under a license that grants users the right to change, reuse, and distribute the software to anyone for any purpose [1]. OSS facilitates open collaboration that includes the contributions of thousands of talented volunteers (e.g. programmers and scientists) in making conceptual and practical impacts in their communities, and not surprisingly OSS has become more mainstream and commercially viable in recent times [2]. Some popular OSS, such as Linux, Python, and R, are developed, maintained, and reused both within and outside of academic institutions, through the contributions of individuals from academia, non-profit organizations, commercial organizations, and other professional entities. Many authors, whose names are often forgotten or unnoticed, spend hundreds of hours of their time to develop OSS that supports and empowers the scientific community. However, academic metrics do not include a systematic way to quantify the value of such effort, except for academic citations [3].

In the research community, very few researchers have proposed initiatives to quantify OSS contributions. The only exception is probably the open source project Depsy.org, developed by Impact Story [4]. Specifically, it tracks not only citations within academic literature, but also alternative metrics such as number of downloads, software reuse through reverse dependencies, and contributors to the OSS. Their dataset facilitates the creation of contributors and dependencies networks that, in turn, allows one to estimate or quantify the impact of the packages' network features on the performance, namely number of downloads and citations [5, 6]. Although datasets like this have assisted researchers to obtain some interesting results [5, 7], past research lacks the longitudinal perspective to have causal relationship between package attributes and performance (since such a causal relationship may take a long time to realize).

Despite the lack of approaches to credit scientists and programmers for their efforts, the OSS ecosystem has expands significantly, particularly in the last two decades [7]. The introduction of technological artifacts and software-based artifacts for knowledge sharing and creation has been crucial for the OSS ecosystem [8]. For example, the literature on free/libre open source software (FLOSS) emphasizes the role of knowledge exchange and collaboration in OSS development [9, 10]. Online OSS free repository facilitates collaboration and social interaction among developers that, in turn, improve the effectiveness of distributed teams [11]. Such repositories also record and keep track of critical usage information beyond software description, such as authorship, date of

URI: https://hdl.handle.net/10125/63797
978-0-9981331-3-3

publication, number of daily/monthly downloads, version, dependencies, reverse dependencies, and scientific publications. This, in turn, allows both the authors and users to see the source of contribution and the path of adoption. The approach used in past research focusing on the OSS collaboration networks is largely cross-sectional quantitative or qualitative evolutionary. Hence, one can observe that there is a lack of quantitative analysis on the evolution of OSS collaborative networks over a period of time.

Leveraging the longitudinal data collected through web scraping of the comprehensive R archive network (CRAN), our analysis contributes to theory and practice of OSS movements in three ways. First, we identify the factors that have the most significant contributions to the performance of R packages, prior to establishing causal relationship between such factors and the outcomes over an extensive period of time during which the network grows. Second, our longitudinal approach allows us to uncover how the network structure changes over time and examine if such dynamics can affect the package's performance. Finally, the longitudinal approach may reveal patterns and characteristics of the network and its components that are not identifiable through cross-sectional analysis. A better understanding of the network dynamics will contribute to the development of alternative metrics that reveal the under-recognized contribution of many scientists and programmers [3] and provide better incentives to facilitate the development efforts and consequently the growth of the network. This is the contribution we seek to make in this paper.

In this paper, we use data collected from CRAN on R packages to generate 77 monthly snapshots in the time window between October 2012 and February 2019. The data for each package includes the number of monthly downloads, dependencies and reverse dependencies, the eventual scientific paper that builds on the package (if any), and the date of publication. This allows us to derive a graphical representation of the relationships among the various packages on monthly basis. Such a dynamic network construction provides a systematic way to identify the structural features of the network, which are then used as the predictors of each package's performance. Through our empirical analysis of this comprehensive panel dataset, we find that network measures, such as closeness and page rank, significantly influence the number of downloads. Moreover, we show that the number of downloads reflects the flow of information from the core to the periphery with a salient spillover effect. Finally, we demonstrate that the network of packages evolve over time with a consistent pattern, which applies to not only established entries but also new entries that are recently added to the network.

The rest of the paper is organized as follows. The next section provides background information on R packages, followed by a brief review of the literature. Then, we introduce our methodology and statistical approach. Finally, we present and discuss the implications of the results and conclude the paper, as well as discussing the implications for both research and practice.

2. Background

R is a free programming environment for statistical computing released in 2000 under the general public license GNU. It is available for various operating systems, and is highly extensible through user-submitted packages for specific functions or domains. This makes R one of the fastest growing data analysis software on the market. In particular, the multiplatform orientation and the ease of extending the functionalities through its lexical scoping rules have fostered the growth of an ecosystem, in terms of packages that interact with each other to provide hundreds of thousands of functionalities. In addition, the object-oriented nature of R language makes the reuse of functionalities included in other packages extremely easy. This generates a network of dependencies that offers a broad range of statistical techniques and graphs widely accepted in scientific publications, and high-quality documentation, such as LaTex-like output.

To manage the growing body of the releases of the new packages and the updates of the existing ones, in 2012 the CRAN was developed for users to submit their improvements to address reported bugs / vulnerabilities and for systematically storing the most recent releases of R code and documentation. Since then, the number of packages through CRAN has increased from 3,350 to 13,750 (as of February 2019). CRAN checks each submission to ensure compliance, verifies the consistency of the dependency network and the compatibility of packages with the R version, tracks the package's version, checks the code for malicious or antisocial activity, and then makes the compiled package available publicly. Such activities assure a set of high-quality standards is consistently applied to the large number of packages offered to the growing community of users across a wide range of domains. Although these packages contribute to scientific progress, there are no well-established measures that evaluate such contributions and their benefit. Hence the key objective of this research is to

develop metrics that give credit to the "unsung heroes" of scientific software for their contributions and explore the different metrics that can be used to predict the growth of the network.

3. Related Work

The creators of Depsy, a free website launched in 2015 that tracks the "value of software that powers (or empowers?) science", discuss the need to measure the contribution of software for academic purposes. In academia, publications are, probably, the most used metric to measure one's research achievements, although publications may not be representative of all contributions made by the researcher. For example, they do not cover the efforts devoted to developing a reusable software and its scientific benefits. Even when researchers are highly encouraged to explicitly cite the source of the software used in their research, merely doing so does not fully address the issue. For example, a software package may depend on multiple other packages published earlier. Hence, only citing the software used for the research does not give credit to the chain of dependencies on these earlier packages. For example, the partial least squares package, "plspm" [12], depends on the functionalities offered by five other packages, and in other cases the chain of dependencies can be longer. Therefore, it is not feasible to use citations as a measure of impact.

From an Altmetrics perspective, Zhao and Wei [6] propose three influence indicators to evaluate the impact of OSS, namely the number of downloads, the number of academic citations, and the network dependency factor. These three indicators reflect the three aspects of software reuse. First, software downloads reflect the usage, the visibility, and, to a certain extent, the reputation of the software. Second, the number of citations in scientific publications measures the usefulness and the direct impact of software on the research outcomes (although it is still not a widely-established practice to cite the software in scientific publications). Third, the network dependency factor reflects the chain of reuse of a software, thus measuring the indirect contributions to a research. From a network structure perspective, Korkmaz and Kelling [5] propose an approach that focuses on the relationship between centrality measures in coauthorship networks and scientific productivity [13]. They show that network measures, such as indegree, outdegree, closeness centrality, betweenness, eigencentrality, and clustering coefficient, are significantly associated with number of downloads and citations in both packages' dependency network and contributor social network. Conversely, they provide evidence that pagerank is not associated with the number of downloads in the dependency network. Although these studies provide interesting results, the cross-sectional nature embedded in these studies does not enable the inference of causality among variables. In addition, past research on co-citation networks was based on undirected networks [14], thus failing to recognize the asymmetric relationship between nodes.

As proposed by Korkmaz, OSS development for scientific research is closely related to the social network of collaborative production [5]. Indeed, patterns of contribution and interaction among the contributors' network are crucial in explaining the success of FLOSS projects [15]. The topological properties of the OSS development community enable fast communication of information that optimizes resource allocation [16]. Perhaps, this highlights the crucial role of communication and information transfer in the development of FLOSS. Knowledge reuse, one of the mechanisms that enables information transfer, benefits the development of OSS in many ways, such as reduced projects' costs, shorter development time, and enhanced quality of the software produced [17]. Therefore, the inclusion of one or more OSS artifacts, such as R packages, in a project is a form of knowledge reuse. Given the nature of the interactions, the open source package network is directed and non-acyclic. It is directed because the dependency relationship is directional, reflecting the fact that package A requires package B. It is non-acyclic because it is not possible to return to the same node following a non-trivial path. In social networks, including the coauthors network, if author A is linked to B, B to C, and C to A, it is possible to follow a (non-trivial) path A -> B -> C -> A that returns to the starting point, which is the definition of cyclic network. In a dependency network, cyclic paths, such as the one shown above, are not possible due to the nature of the relationships. Since the direction of a link contains important information such as asymmetric influence or the direction of the information flow, a link between a pair of nodes may represent a fundamentally different dynamic when its direction is reversed. Therefore, disregarding the direction may fail to explain the dynamics and the function of the network.

We propose to approach the study of OSS networks from a one-to-many information dissemination perspective, which will contribute in two ways to the understanding of this topic. First, the broadcast of information to all recipients reflects the flow of information that exists between a package and its

dependents. In this sense, the creators of a package are like broadcasters of information that may benefit other users in the community [18]. Past research that adopted this perspective has focused on examining information dissemination in blogs [19] or microblogs (like Twitter) [20]. Second, the information flow perspective allows us to introduce the temporal dimension to our network analysis. For example, Yasseri and Sumi [21] estimate the geographical distribution of a network of editors through the study of differences in their temporal activity. For these reasons, our approach is consistent with the directed acyclic nature of OSS networks and adds the temporal perspective that, in our opinion, is crucial to understand the behavior of dynamic networks.

4. Methodology

4.1 Data

We collected data on all the R packages listed on CRAN (13,572 packages as of March 7th, 2019) and scraped the monthly downloads statistics using the R function *cran_stats* included in the package *dlstats* [https://cran.r-project.org/package=dlstats]. In total, the information collected spans over 76 time points (months), from November 2012 to February 2019, and includes several key characteristics for each package at each time point, such as the dependency and reverse-dependency list, monthly downloads, contributors' names, publication date, citations of the scientific papers that build on the package (if they exist), and tags (labels identifying additional characteristics of the OSS package). Table 1 presents the network statistics in two-year intervals throughout our sample period (except the last interval which covers only one year). One can observe that the nodes, edges, and the number of downloads increase steadily, along with the average Indegree measure and network diameter. At the same time, the number of packages with a zero indegree value also increases, and the list of top downloaded packages has shown a moderate turn-over rate, with several constant top performers constantly showing up on the list.

One can also observe from Table 1 that the top three downloaded packages are 'ggplot2', a popular graph package, 'plyr', a package that offers a set of function to manage datasets, and 'rcpp', a package to integrate c++ programs into R. The average indegree value changes over time, reflecting the fact that the complexity of the network is increasing. This is also confirmed by the increase in the network diameter, defined as the longest of the collection of shortest paths between each pair of nodes. The number of packages without indegree is almost stable at 75 percent of the whole population. These packages can be considered as the passive receivers of the flow of information in the network.

Table 1: Network statistics over time

Time Point	Nodes	Edges	Number of Downloads	Top Downloaded Packages	Avg Indegree	Packages w/out inDegree	Network Diameter
Nov 2012	3,438	3,846	529,359	plyr / colorspace / stringr	1.90	2,469	9
Feb 2014	4,644	5,881	3,085,126	digest / plyr / ggplot2	2.06	3,394	9
Feb 2016	7,482	12,627	15,485,019	rcpp / ggplot2 / digest	2.42	5,545	9
Feb 2018	11,785	27,709	33,665,863	rcpp / tibble / rlang	2.96	8,800	11
Feb 2019	13,752	35,315	72,492,261	rlang / rcpp / ggplot2	3.14	10,335	11

4.2 Dependency Network

We perform the analysis of the OOS network based on approaches used in the information broadcast literature. In social network analysis, information relationships reflect the type and amount of information exchanged between actors (or nodes) [18]. The pattern of such relationships reveals the probability for actors to be included into an exchange of information which, in turn, is instrumental in assessing the level of influence of each node in the communications at a local level and across the whole network. The directional patterns of the communication describe how information moves around and how much actors can facilitate or control the flow. A number of aspects of information can be studied using approaches in social network analysis, including information needs, information exposure, information flow, information control, and information opportunities [22]. As discussed in the introduction, the major drawback of the social network approach lies in its cyclic nature. In our case, a cyclic network characterization is not possible due to the nature of the relationships between packages. In

addition, the increasing popularity of the analysis of communication network has led to the emergence of new and more sophisticated ways to model network structures [23]. Among these models, the broadcast communication network is a good fit for our study. The broadcast communication network is a form of acyclic directed network in which the information flows from sources of information toward the community of users [24]. This approach is useful in identifying patterns of information flow from a sender to receivers and to identify influential actors and gatekeepers in the network [25].

In the same way, we can identify packages as actors in a communication network, and the reuse of a package is a relationship directed from the receiver toward the source of information. Under this setup, the dependency structure between packages available in the online repository CRAN defines the sender-receiver relationship in the network. The dependency is instrumental in measuring the flow of information within a network. An edge directed from package 'A' to package 'B' indicates that package 'A' reuses functionalities from package 'B'. From an information broadcast perspective, the direction of the link goes in the opposite direction of the flow of information. In other words, the link points towards the source of information. The network defined in this way is directed and acyclic, since it is impossible for a software project to be dependent upon itself. The OOS network is thus suitable to be analyzed as a communication broadcast network. From here thereafter, we will use the terms "information network", "broadcast communication network", and "communication network" interchangeably. Figure 1 shows an overview of the R packages network. Graph visualizations enable one to more easily understand the complexity and underlying structure of the graph. For example, Figure 1 depicts the evolution of the R packages network over three point of time (respectively 11/01/2012, 02/01/2016, and 02/01/2019). The size of each node reflects their inDegree centrality measure and the color reflect their respective cluster. These clusters can be explained by the functions and disciplines of the packages [14] and take the name of the most influential package.

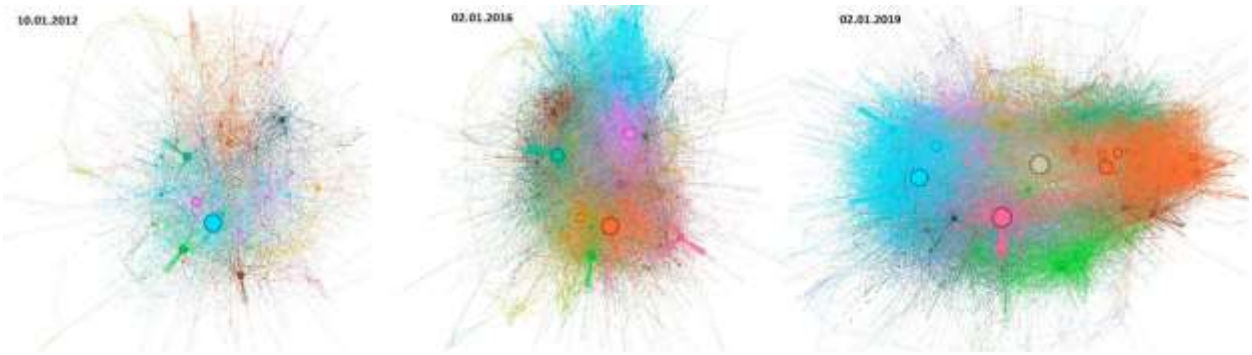

Figure 1. Evolution of the R packages network over time

4.3 Measures

The focus of this research is to examine how the performance of a package, as measured by the number of downloads, is affected by its network properties and measures of centrality such as indegree, outdegree and measures of dependency between nodes. As predictors of our model, we select centrality measures that are relevant for directed acyclic network. The selection is limited to the most commonly used measures of centrality in social network analysis, namely indegree, outdegree, betweenness, closeness centrality, and a variant of Eigenvector centrality, PageRank [26]. In our OOS network, the value of the *indegree* measure reflects how many times each package has been reused. Accordingly, the *outdegree* value shows how many packages have been reused in each package. From a flow of information perspective, the two most frequently used measures in the analysis of information transmission in social networks are the vertex *betweenness* and vertex *closeness* centrality [27]. These centrality measures are based on the assumption that when possible, information is transmitted along the shortest paths. While betweenness centrality measures the degree to which a node (vertex) may control the communication channel between any two vertices (the number of shortest paths that passes this node for a given pair of vertices), and closeness is just the inverse of the average shortest distance to other vertices. Intuitively, betweenness centrality represents the degree to which a node stands between each other. For example, a node with higher

betweenness centrality would have more control over the communication within the network, because more information will pass through that node. On the other hand, closeness centrality reflects the nominal definition of centrality. The more central a node is, the closer it is to all other nodes.

In this research, we use the normalized form of closeness and betweenness centrality that allows comparisons between nodes of graphs of different sizes. In addition to these four centrality measures, indegree, outdegree, closeness and betweenness, we also include a measure of network influence, the *pageRank* centrality measure [28]. PageRank measures a node's influence by taking into account how well connected a node is, and how many links their connections have, and so on, through the network. This measure fits our approach because high values indicate a strong influence over nodes that are more than a step away. In contrast to other measures of network influence, such as EigenCentrality, PageRank is designed specifically for directed networks. Therefore, it is able to uncover influential or important nodes in a directed graph whose reach extends beyond just their direct connections. In respect to an undirected network approach, our approach through an information flow perspective has some advantages. First, it fits very well the acyclic directed network of OSS packages. A flow of information assumes a sender (that creates the information) and a receiver (that uses the information). The inDegree and outDegree centrality scores of each package respectively measure the creation and the use of information. In contrast, in an undirected network, inDegree and outDegree will have the same value for each node. Second, from the information flow perspective closeness and betweenness centrality reflect the speed and frequency of exchange of information within a network. These are salient features of our longitudinal dataset, and first well fits our proposed panel data analysis (to be discussed later). Finally, using the amount of downloads as a proxy for package performance is consistent with our approach. The reuse of a package through inclusion in the dependency list reflects a transfer of knowledge between nodes in the network.

4.4 The Temporal Perspective

In a highly dynamic network such as the open source network for the R package, the temporal dimension contains rich information about the growth and evolution of the network. We find that the average number of dependencies per node increased by three times over the time frame of our study. Such a speed of evolution is usually not observed in a static network. In addition, the number of downloads increased 140 times in the same period, reflecting the increasing popularity of this statistics framework. Table 1 shows the change of the average indegree value over time, reflecting the fact that the complexity of the network is increasing. This is also confirmed by the increase in the network diameter, defined as the longest of the collection of shortest paths between each pair of nodes.

4.5. Analysis

We perform the longitudinal analysis of our sample through a panel data analysis. A key benefit of panel data is the ability to control for the effect of all stable covariates without explicitly including them in the model. We apply a longitudinal fixed-effect model that uses within-package variance to estimate the coefficients and then averages the estimates across the packages. The fixed-effect models are optimal for removing the pernicious effect of omitted variable bias when multiple panels (sections) of data are present and available. Moreover, the Hausman test [29] suggests some evidence against the random effects model and in favor of the fixed effects model. Due to the nature of our dependent variable (count data), we adopt a generalized linear model approach through the Poisson regression. Furthermore, to avoid the underestimation of the standard errors caused by overdispersion of the number of downloads, we adopt the quasi-likelihood estimation [30]. Instead of specifying a probability distribution for the data, only the relationship between the mean and the variance is specified by a function that includes a multiplicative factor (overdispersion or scale parameter) that is estimated directly from the data. Past research shows that the quasi-likelihood estimation for a Poisson distribution gives a better fit to the overall variance-mean relationships [31]. Given the dynamic nature of the OSS network, we cannot assume the invariance over time of the predictors' effects on performance. Therefore, to test for moderating effects we introduce interaction terms for each variable in the model [32]. We perform forward selection including the first-order interactions between predictors to identify only the significant variables.

We use normalized measures of indegree and outdegree centrality in order to allow for comparisons between nodes of graphs of different sizes. For the same reason, for each cross-section we normalize the number of downloads as the percentage of the number of downloads of the whole network. We would like to point out that the centrality measures are derived from

the network configuration at the first day of each month, while the number of downloads refers to the total number of daily downloads in that month. The purposely introduced time lag between the two measures provide additional support to the claim of causal relationship between predictors and dependent variable.

Another issue to address before running the analysis is the multicollinearity between many of the centrality measures included in this study. For example, the pageRank score depends upon the number of indegree links, therefore we can expect high variance inflation factor (VIF) values when both features are included in the model. Moreover, we expect the closeness to be correlated with indegree, because the higher the number of incoming links, the shorter the average path to each node of the network. Indeed, the Pearson correlations of pageRank with indegree and closeness are 0.76 and 0.79, respectively, thus implying that multicollinearity may be an issue. The correlation coefficient between closeness and indegree is 0.83. Following the best practices in literature, we set the VIF cutoff equal to 5. Table 2 reports the VIF scores for all the variables included in our model. The values are below the cutoff value.

Table 2: Variance Inflation Factors (VIF)

Variable	VIF
closeness	4.18
betweenness	1.37
indegree	4.34
outdegree	1.02
pRank	3.25

5. Findings

Table 3 reports the results of the regression for the panel data analysis. All the centrality measures that we have included in our study have a significant effect on the dependent variable except for Page Rank (PR). This result echoes the findings reported in [5], and can be explained by looking at the definition of this centrality measure. Page Rank [33] is designed to reflect a global ranking of all web pages based solely on their location in the network. It performs very well on strongly connected and static networks, such as identifying influential websites on the Internet. However, it suffers from a number of limitations when analyzing dynamic and weakly connected topologies, such as identifying influential leaders in social networks [34]. In our case, the network or R packages is weakly connected and the topology changed rapidly since the beginning. This makes Pagerank not so useful for predicting influential nodes and, in turn, their performance.

Table 3: Results for Panel Data Fixed Error Poisson Regression with Robust Error Estimates

DV=Downloads	Coef.	Rob Std Err.	95% CI	
closeness	7.473*	3.04	1.517	13.43
betweenness	-11658***	2686	-16924	-6392
indegree	52.64***	13.34	26.48	78.80
outdegree	1019.9***	102.4	819.1	1220
prank	-33.97^{ns}	19.32	-71.85	3.907
time*inDegree	-.4009***	.1105	-.6175	-.1842
time*outDegree	8.855***	2.541	3.873	13.83

Note: '***' p < 0.001; '**' p < 0.01; '*' p < 0.05; ns = not significant

Within our approach of modeling the OOS network as a flow of information, closeness centrality plays a crucial role as it represents the speed of transmission. A node that is closer, on average, to all other nodes in the network, will have faster communication with nodes in the network. In other words, betweenness reflects frequency of arrival (or transit) of information, and closeness reflects time-until-arrival of the information flowing through the network. The betweenness score reflects how often the node plays a role in the communication between two randomly chosen nodes. Nodes with high betweenness score are more influential for the flow of information, because the removal of such nodes could seriously disrupt the communications [35]. In other words, packages that reuse more functionalities from other packages and that are reused by many packages become more influential in the network. This perspective explains the positive effect of closeness on the number of downloads. Surprisingly, betweenness centrality has a negative influence on the packages' performance (Figure 2). A plausible explanation is that, in a more complex network, it is easier to find an alternate route in respect to the path through the influential node.

The role of inDegree centrality shows a temporal pattern consistent with our approach. The number of incoming links reflects the number of packages that reuse the information included in each node. This reflects the level of influence of each node that, in turn, affects the number of downloads. In addition, the interaction term with time is significant and negative. This means that inDegree has more influence on the performance in the early stages of the network. In other words, high scores of inDegree centrality are more important in small networks than in bigger ones. There are two potential explanations. First, as shown

in Table 4, the proportion of nodes with incoming links slightly decreases over time, which, in average, negatively influences the effect of inDegree on the number of downloads over time. Second, over time the network become more complex. Table 1shows that the network diameter, a measure of complexity, increases from 9 in 2012 to 11 in 2019. In a more complex network, there is more competition among packages that, in turn, negatively affects the relationship between inDegree and performance.

Finally, outDegree has a strong positive influence on the number of downloads. This finding aligns with cross-sectional results in past studies [5]. From a communication perspective, in the OSS context, outDegree reflects the reuse of information provided by other packages. Therefore, a package with higher levels of outDegree centrality is more likely to contribute more to the local flow of information that, in turn, influences the visibility and performance of the node. Interestingly, the outDegree increases its influence on the performance when the network becomes more complex. This suggests that there are substantial network externalities in the OSS networks such that an increase in network size may exponentially leverage the impact of various network properties on outcome variables such as the number of downloads.

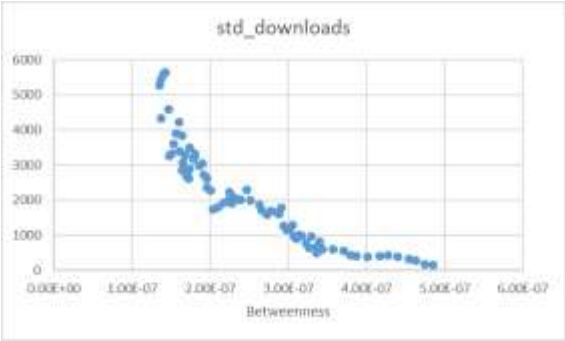

Figure 2. Betweenness vs Standardized Downloads

Table 4: InDegree and outDegree over time

Time Point	Graph Size	Number of Edges	Nodes with indegree>0	Nodes with outdegree>0
Nov 2012	3,438	6,525	969 (28%)	2,224 (65%)
Feb 2014	4,644	9,580	1,250 (27%)	3,093 (67%)
Feb 2016	7,482	18,117	1,936 (26%)	5,298 (71%)
Feb 2018	11,785	34,939	2,985 (25%)	8,901 (76%)
Feb 2019	13,752	43,166	3,417 (25%)	10,956 (77%)

6. Discussion and Conclusions

This paper focused on the evolution of the OSS network of R packages over time and the effect of the network dynamics on each package's performance. By compiling a longitudinal dataset collected from the online repository CRAN, we were able to apply an information transmission approach for analyzing the network dynamics through a panel data analysis. We found that the betweenness, closeness, inDegree, and outDegree centrality measures influence the performance of each package, as measured by the monthly downloads. When starting an OSS project, the contributors should take into account the positioning of their software in a complex network such as CRAN. Their package should be strategically positioned in a way that can be accessed and reused by a meaningful number of relevant projects. Doing so will positively influence the centrality scores and, as a consequence, the visibility of the package. In addition, a package's betweenness centrality measure should be minimized by positioning the project close to the center of a specific area of the network. In other words, package developers should focus on features and functionalities that are related to the most popular packages currently available in the network. Over time, closeness and outDegree centrality measures are the best predictors of package performance. Within a network characterized with frequent communication and collaboration, and thus highlighting the fact that communications are most effective when conducted through shorter paths. On the other hand, the outDegree measure reflects the amount of information reused and subsequently propagated by the package to other downstream packages. Such a measure captures

the levels of connectivity and influence of each node within the communication network and, to some extent implies the ability of a network to inherit knowledge and pass it from generation to generation.

6.1 Implications for research

We offered a new approach to explore the factors that affect the impact of OSS packages on the users' community. From a methodological standpoint, due to the longitudinal perspective and the panel data analysis approach, it is not surprising that some of our results contradict past findings [5]. Past research focuses mainly on scientific literature contributions (i.e. citations), thus shifting the focus away to outcomes that are exogeneous to the network and failing to capture the important directional and noncyclical nature of the OSS networks. To address these limitations, our study seeks to capture the above network characteristics by modeling the creation and transmission of information through a directional network. With R packages and their contributors as nodes and information broadcast (package dependencies) as directional relationships, the resulting network and the relevant centrality measures allow us to assess the crucial role of generating scientific knowledge in term of influence and performance. Further conceptual work and literature review are required to fully validate our perspective. The longitudinal approach to the evolution of the OSS network from an altmetrics perspective should incorporate additional measures of performance (e.g. number of citations). In addition, it may be interesting to explore the potential interactions between the OSS artifacts network and the FLOSS developers' network. In short, do the developer team's social connections affect the positioning of the OSS artifact (e.g. dependency list)?

6.2 Implications for practice

The present research, even in its exploratory state, offers some suggestions for OSS artifact design. In the early stages of the artifact design, the developer team decides the functionalities that need to be created and what functionalities can be reused from other artifacts. These choices will affect the artifact initial positioning within the network and its future trajectory. Through our analysis, regular patterns of information flow reveal opportunities for the packages contributors in terms of exposure and performance. Moreover, the longitudinal perspective contributes to the discovery of the trajectory of each package's influence and performance over time, thus enabling the scientific community to recognize and evaluate the contributions of various network participants, and informing the contributors on the best routes for delivering scientific values. A detailed understanding of the factors that influence the artifact success would help FLOSS contributors in optimizing the artifact design and maximizing the impact on the community.

7. References

[1] Laurent, A.M.S., *Understanding open source and free software licensing: guide to navigating licensing issues in existing & new software*. 2004, Sebastopol, CA: O'Reilly Media, Inc.

[2] Fitzgerald, B., *The transformation of open source software.* MIS quarterly, 2006: p. 587-598.

[3] Singh Chawla, D., *The unsung heroes of scientific software.* Nature News, 2016. **529**(7584): p. 115.

[4] Piwowar, H., *Altmetrics: Value all research products.* Nature, 2013. **493**(7431): p. 159.

[5] Korkmaz, G., et al. *Modeling the Impact of R Packages Using Dependency and Contributor Networks*. in *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. 2018. IEEE.

[6] Zhao, R. and M. Wei, *Impact evaluation of open source software: An altmetrics perspective.* Scientometrics, 2017. **110**(2): p. 1017-1033.

[7] Kikas, R., et al. *Structure and evolution of package dependency networks*. in *Proceedings of the 14th International Conference on Mining Software Repositories*. 2017. IEEE press.

[8] Lanzara, G.F. and M. Morner. *The knowledge ecology of open-source software projects*. in *European Group of Organizational Studies (EGOS Colloquium), Copenhagen*. 2003.

[9] Crowston, K., et al., *Free/Libre open-source software development: What we know and what we do not know.* ACM Computing Surveys (CSUR), 2012. **44**(2): p. 2-37.

[10] Howison, J. and K. Crowston, *Collaboration Through Open Superposition: A Theory of the Open Source Way.* Mis Quarterly, 2014. **38**(1): p. 29-50.

[11] Crowston, K., et al. *Effective work practices for FLOSS development: A model and propositions*. in *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*. 2005. IEEE.

[12] Sanchez, G., *PLS Path Modeling with R*. 2013, Berkeley, CA: Trowchez Editions.

[13] Yan, E. and Y. Ding, *Applying centrality measures to impact analysis: A coauthorship network analysis.* Journal of the American Society for Information Science and Technology, 2009. **60**(10): p. 2107-2118.

[14] Li, K. and E. Yan, *Co-mention network of R packages: Scientific impact and clustering structure.* Journal of Informetrics, 2018. **12**(1): p. 87-100.
[15] Crowston, K., et al. *Core and periphery in free/libre and open source software team communications*. in *Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS'06)*. 2006. IEEE.
[16] Xu, J., et al. *A topological analysis of the open souce software development community*. in *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*. 2005. IEEE.
[17] von Krogh, G., S. Spaeth, and S. Haefliger. *Knowledge reuse in open source software: An exploratory study of 15 open source projects*. in *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*. 2005. IEEE.
[18] Haythornthwaite, C., *Social network analysis: An approach and technique for the study of information exchange.* Library & Information Science Research, 1996. **18**(4): p. 323-342.
[19] Adar, E. and L.A. Adamic, *Tracking Information Epidemics in Blogspace*, in *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence*. 2005, IEEE Computer Society. p. 207-214.
[20] Kwak, H., et al., *What is Twitter, a social network or a news media?*, in *Proceedings of the 19th international conference on World wide web*. 2010, ACM: Raleigh, North Carolina, USA. p. 591-600.
[21] Yasseri, T., R. Sumi, and J. Kertész, *Circadian Patterns of Wikipedia Editorial Activity: A Demographic Analysis.* PLOS ONE, 2012. **7**(1): p. e30091.
[22] Case, D.O. and L.M. Given, *Looking for information: A survey of research on information seeking, needs, and behavior*, ed. J.E. Mai. 2016, Bingley, UK: Emerald Group Publishing.
[23] Monge, P.R. and N.S. Contractor, *Emergence of communication networks*, in *The new handbook of organizational communication: Advances in theory, research, and methods*, F.M. Jablin and L.L. Putnam, Editors. 2001, Sage: Thousand Oaks, CA. p. 440-502.
[24] Zinoviev, D. and V. Duong. *A game theoretical approach to broadcast information diffusion in social networks*. in *Proceedings of the 44th Annual Simulation Symposium* 2011.
[25] Gursakal, N. and A. Bozkurt, *Identifying Gatekeepers in Online Learning Networks.* World Journal on Educational Technology, 2017. **9**(2): p. 75-88.
[26] Landherr, A., B. Friedl, and J. Heidemann, *A critical review of centrality measures in social networks.* Business & Information Systems Engineering, 2010. **2**(6): p. 371-385.
[27] Wasserman, S. and K. Faust, *Social network analysis: Methods and applications*. Vol. 8. 1994, Cambridge, MA: Cambridge university press.
[28] Brin, S. and L. Page, *The anatomy of a large-scale hypertextual web search engine.* Computer networks and ISDN systems, 1998. **30**(1-7): p. 107-117.
[29] Hausman, J.A., *Specification tests in econometrics.* Econometrica: Journal of the econometric society, 1978. **46**(6): p. 1251-1271.
[30] Wedderburn, R.W.M., *Quasi-likelihood functions, generalized linear models, and the Gauss—Newton method.* Biometrika, 1974. **61**(3): p. 439-447.
[31] Ver Hoef, J.M. and P.L. Boveng, *Quasi-Poisson vs. Negative Binomial Regression: How should we model overdispersed count data?* Ecology, 2007. **88**(11): p. 2766-2772.
[32] Allison, P.D., *Fixed effects regression models*. Vol. 160. 2009, Thousand Oaks, CA: SAGE publications.
[33] Page, L., et al. *The PageRank citation ranking: Bringing order to the web*. 1999.
[34] Chen, D.-B., et al., *Identifying influential nodes in large-scale directed networks: the role of clustering.* PloS one, 2013. **8**(10): p. e77455.
[35] Borgatti, S.P., *Centrality and network flow.* Social networks, 2005. **27**(1): p. 55-71.