DECISION SCIENCES INSTITUTE

The Usage of R in Finance and Banking

Ceyhun Ozgur
Valparaiso University
Email: ceyhun.ozgur@valpo.edu

Sanjeev Jha
Valparaiso University
Email: sanjeev.jha@valpo.edu

Elyse "Bennie" Myer-Tyson
Valparaiso University
Email: bennie.myertyson@valpo.edu

David Booth
Kent State University
Email: dvdbooth8@gmail.com

ABSTRACT

R has grown tremendously over the years in terms of number of users and capability with the development of hundreds of packages. In this research, we investigate the usage of R in Finance and Banking areas. We begin with a comparative analysis of R with other computing software like SAS and Python. Then we discuss the reasons for the growth of R's usage in financial sector. We end with a comparative evaluation of Python and R's strengths and weaknesses in a classroom.

Keywords: R, Finance and Banking, teaching, research, Python, SPSS, SAS

INTRODUCTION

R is software designed to run statistical analyses and output graphics by user-input code. It can run on virtually any operating system, and is open source. (Ooi, 2011, the R Foundation, 2017). This makes the software highly appealing, as it is able to keep up with the demands of a growing number of varied business structures. Standard software has been SAS and Python; however, a growing number of jobs are posted looking for experience using R in the data analytics field (Muenchen, 2017). This shifting interest paradigm has consequences not only for those who have graduated and are seeking employment, but also for those who are currently in statistics or analytics programs. The changing field preferences should be reflected in the way data science is taught so as to give students the best possible education and preparation for working in the field. One of the core benefits of teaching R to students is that it is a highly standardized programming language (Economist 662f, 2014). This means that students need not worry about highly variable language structures or deep understanding of the different languages with which one can code. Rather, they are free to focus on the analytics side of R and understand the applications of the program in a real-world setting.

LITERATURE REVIEW

R'S GROWTH AND COMPARISON TO OTHER SOFTWARE

R's growth in the data analytics field has come to the point where it is beginning to surpass the field standard software such as SAS, SPSS, Stata, and MatLab (Muenchen, 2017). This surge in the number of R users has led to an increase in job postings asking that applicants understand and can use R. The field paradigm changes are well marked by Muenchen (2017) who shows the overall trend in Figures 1, 2, and 3 below. These figures clearly show the spike in R's popularity, especially over the last 5 years.

Figure 1: Number of data science jobs posted on a job search website with over 250 job hits

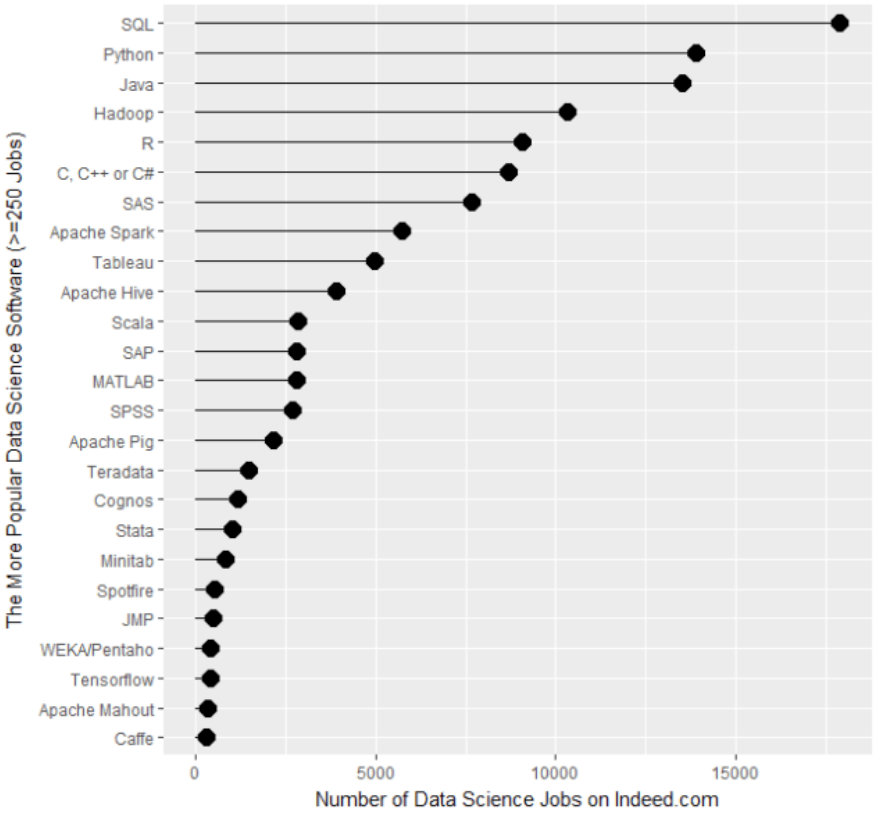


Figure 2: The job posting trends for SAS (orange) and R (blue)

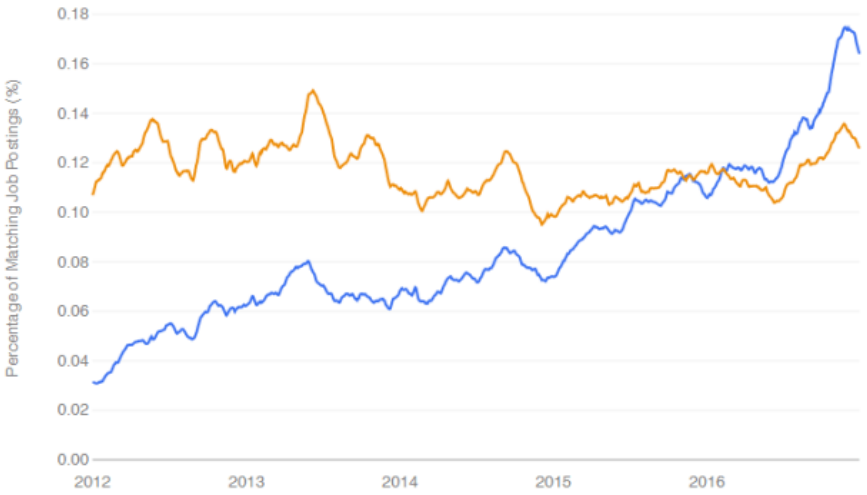
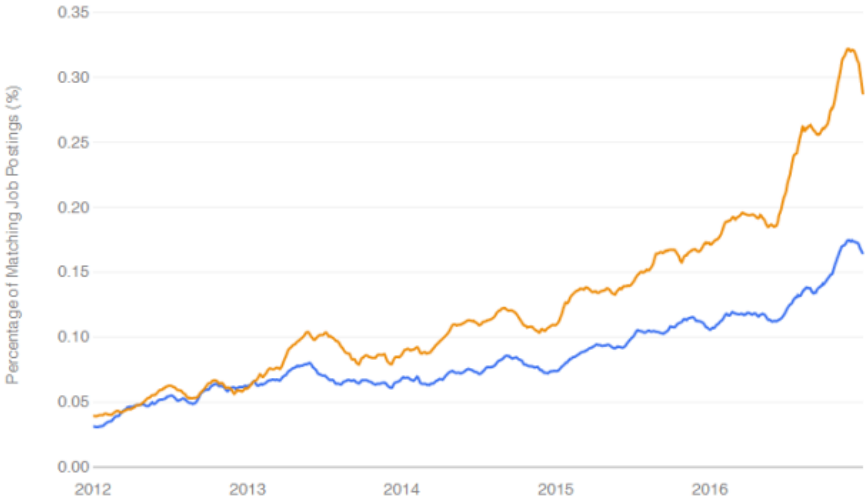


Figure 3: The job posting trends for Python (orange) and R (blue)



As is evidenced by the figures, R is beginning to overtake the big names in the field. While it is more popular than SAS, it has not yet reached the same level of triumph over Python. However, Muenchen (2017) notes that this could be due to the fact that Python jobs are much more diverse in nature than R jobs are, by virtue of the way the two different programs are set up. Python is just as code-heavy as R, but R is restricted in its application just to data and information science applications. Therefore, one must be careful when attempting to draw conclusions from the general trend and must look closer at the data science landscape specifically.

The popularity of R in the field is reflected by the amount of resources available with which to learn R. There are several books written devoted to the software overall and to general packages (Varma, 2016, Springer 2014) which can complement the various websites, blogs,

video tutorials, and conference papers available to new and veteran users (Rickert, 2014, Mejia, 2013). This plethora of available information also makes teaching the software easier, as students are able to access information about specific packages or the debugging process with a quick web search. The packages themselves are easy to use and install; they are available from R's CRAN depository (R Core Development Team, 2016) and need only be installed once (Arriata, 2016).

DISCUSSION OF R

R in finance

R's flexibility as a software means that it is fully able to perform the necessary tasks for any branch of data analytics. One of the main subsets of analytics where R shines is the financial applications. There have been several programs written to facilitate the usage of R in such a setting; one of the most popular is jrvFinance (Trajanov, 2017). This package, alongside the standard tools of R and the bundle package Rmetrics (Trajanov, 2017), allows users to perform the necessary standard financial computations in the software. The largest advantage of this program is that it is set up to mimic the Excel format with which students and new users are likely very familiar. This decreases the time spent trying to decipher a readout or table produced from the R code and as well facilitates learning by analogy. As Varma (2016) notes in his overview of the jrvFinance package, there are several specific commands to ensure that the correct operations are being performed. In addition to this, the detailed breakdown of the options of the commands also can aid new and learning users in acquiring skillsets that will continue to help them develop within the data science field. JrvFinance specifically allows for the Internal Rate of Return (IRR) operations within R, which are arguably indispensable and should be taught to all incoming students.

These applications of R extend outside of the classroom and job settings. As Moore (2014) notes, one could use R to manage one's own finances. This handling of personal data on a personal level has twofold benefits. First and foremost, consumers are able to better understand the practices around data analytics (and may therefore be more willing to participate in data collection on the whole, which facilitates greater analytical power in future tests) because they can manipulate their own data and understand the security measures upon examining the files received from their banks. Secondly, this careful tracking of one's own finances could be considered good practice or even as a simple introductory exercise to learning R. This could reduce the user or student push back against learning a new method, as they can immediately see and apply the benefits to their own lives. In addition, there are several packages available to help students analyze financial data from large companies if they would rather work with truly anonymous data (Rickert, 2014).

TEACHING WITH R

As noted earlier, R is becoming more and more commonplace when looking at the data analytics landscape. This means that those wishing to go into a career in big data analytics should be looking to learn this software. However, there has been some pushback against it from the academic community.

As Mohamed (2017) notes, there are different optimal uses for different software packages. This means that users may feel they are wasting their time in learning new software if they can

manipulate the data in another. However, learning the new program can give one a competitive edge and make tasks which are possible in other software much less cumbersome when run in R.

The argument against learning R as a code is that it is typically difficult to handle, since it has its own syntax (Markham, 2015). However, this ignores the plethora of material available to help one learn the course, which has been previously addressed. The other main concern is that code-heavy software is not as user-friendly in the initial stages. While this may be true, R lends itself to a greater understanding of the data by those running the software. The plotting packages (especially ggplot2) can make the data readable to the layperson (Arratia, 2016, Markham, 2015, Meija, 2013, Nau, 2016). The main concern for those teaching others to use software should be whether or not the student understands the program's logic and method of operation. There is a multitude of ways to make data friendly to the average consumer later; the primary focus for education should be on mastery of software operation.

Returning to the financial application mentioned earlier, a simple overview program of teaching R for IRR could look as follows: the user would download FinCal or jrvFinance (or any similar package) and load its syntax. The next step would be to load the package in the current R session (as packages are not stored from session to session). In FinCal specifically, the IRR function is computed by providing cash flow (the first of which is the initial outlay). The syntax would be written as: *irr(cf)* where cf is the cash flow. To extend the example, the syntax *irr(cf=c(-10, 3, 4, 5, 6))* would compute an IRR of 0.2488813for the cash flow specified in the function. The FinCal package can be further extended into computations such as future or present value, however that is beyond the scope of this argument. The presented example is a fairly simple string of code to learn, and should not be considered difficult for experienced undergraduate students. Even those without coding experience could find this to be a simple exercise, as this is not drastically different from applying a function in hand-calculation of a value (Markham, 2015).

As students learn the basics of R and become accustomed to using the software, they can understand the bidirectional effect it has on the job and data analytics market currently (Rickert, 2014). The software grew in popularity in response to the increasing overall demands on big data analytics and the need for programs which could handle massive data files. As R grew in popularity by filling this need, it also changed how computational finance was done; the availability of a new and versatile tool is apt to reframe at least a portion of the discussions around calculation and application. This in turn affects the way people think about the standard equations and affects the solution to current, ongoing problems. All of these factors are highly relevant for those entering the field, and thus point to a necessity to teach this software.

While it may seem overwhelming at first to attempt to use R for financial applications, this is certainly not the case. Rickert (2014) notes that there were at least 70 separate finance packages available in 2014. This may seem daunting to instructors attempting to select the easiest packages to learn for first-time students, but there are several features and sources available to help one make the decision. The R and CRAN websites both showcase prominent packages (R Core Development Team, 2016, The R Foundation, 2017) and the packages themselves are often hosted on the authors' webpages which have further breakdowns and instructions (Rickert, 2014). There are also multiple blog or article comparisons of the package as well as video tutorials available to those who wish to examine packages in greater detail.

CONCERNS OF TEACHING R

R is not without its drawbacks, however topical it may be (Ooi, 2011). There could be conflict between the main package and a supplemental one. This could especially be the case for older packages which may not necessarily be compatible with the latest version of R. While the development team does attempt to mitigate this problem by having strict submission guidelines (R Core Development Team, 2016), it is possible that a once-favored package could stop being updated and, in turn, break down other packages which were relying upon it. This could be compounded by the fact that open source software does not necessarily have a dedicated IT team (Ooi, 2011), which could lead to headaches for both instructors and students.

A further complicating factor of using R is that there could be issues ensuring version standardization across different machines (Ooi, 2011). This is an especially relevant concern if students are using their own personal computers or laptops to do homework. The versions available to them and written for their Operating Systems may have a different setup to what the classroom computers hold or what the instructor is familiar with. A majority of these problems, however, can be remedied by working through third-party software (Ooi, 2011). This software can help refocus R and the packages associated with it to fix buggy installations or known user errors. The major downside to using third-party software, however, comes in the form of cost (Ooi, 2011). Students may find themselves paying for additional software to fix one package that the instructor is familiar with when they could instead be using a separate package at no additional cost to them. This may mean ensuring that staff are all making the most optimal choices for their students both in terms of learning and cost, which can take up more instructor time (particularly when taking compatibility issues into account).

It is possible that the outside materials students could use to aid their understanding are outdated or perhaps incorrect for a particular application of the program (Economist 662f, 2014). This could lead to students having difficulty as they attempt to piece together separate methods of learning or perhaps incorrectly integrate classroom and outside information. R itself is also inconsistent once one gets into the user-created packages. The different authors all have a unique way of writing their code, and this lack of standardization could cause headaches for students and instructors. The easiest solution may be to spend time developing a course-standard language and package set, however this also adds to the time burden on the instructor.

The focus of R itself may not expand well outside of the classroom. R as a program is concerned with how interpretable a model is (Markham, 2015), which comes at the cost of forecasting integrity. While this is useful for a purely statistical course, this does not lend itself well to the finance world in particular. Forecasting is a large part of the field and cannot be ignored if one is to get a full understanding of how data interacts. This is especially true in analytics, where forecasts are often used to make business decisions and should therefore be as accurate as possible.

Data input into R can be difficult at first. This is because of the necessary steps to transform raw data into something meaningful (Markham, 2015). The more time a user has to spend inputting data, the more likely they are to get frustrated with the process. This is especially true when having to ensure that there are no format incompatibilities or inconsistencies with the software being used. R is also not the best tool on the market to explore data (another key component of real-world finance applications). There are packages out there but more elegant solutions can be found by abandoning the software altogether.

BENEFITS OF TEACHING R

While R, like any software package, does have its drawbacks, there are also massive benefits to using R. The first is the plethora of webpages with a comprehensive list of packages with brief description of each package on a variety of data analysis subjects. For example, Dirk Eddelbuettel (2017) maintains a webpage with a list of packages in the domain of Empirical Finance. The webpage is grouped by topics such as standard regression models, time series, finance, risk management, books using R for Financial analysis, data and date management, and a list of R packages. Similarly, others have maintained webpages on topics like time series, econometrics, multivariate analysis, and optimization. We truly believe this consolidated list of packages and methods would help instructors and researchers in keeping up with the latest development. This thorough documentation means that instructors who are concerned about getting the best value out of the packages used are able to examine each one in a good amount of detail. This can cut down on time commitments from the instructors, as they are able to research what has already been written rather than attempting to put together a course from scratch. Rickert (2014) also notes that there is a wide variety of resources out there to aid in the decision of which packages with which to begin.

The interdependent relationship mentioned earlier is also a benefit of teaching R. Software that has, as Rickert (2014) notes, changed the way a field looks at a problem is not likely to fall by the wayside. Additionally, keeping up with such innovative software gives students a competitive edge in the job market as well as a real-time understanding and application of the software. This allows for current-issue exploration and earlier research starting points, which can further bolster a student's career or opportunities.

The field of interest is highly relevant to the decision to teach or not teach R. While R may not be the best decision for finance based solely on the focus towards interpretability over forecasting, this is not to say it is not applicable (Markham, 2015). Certainly, there are those who would be more concerned with understanding models, especially in an introductory course to R. Better forecasts tend to come with higher understanding of a program or method and the underlying assumptions it contains. This understanding can only be garnered by first starting with highly interpretable models; from there, one could expand to forecasting-specific R packages (to shift focus) or to another language or software in order to double-check the computational output. While this may initially seem like more of a drawback than a benefit, the experience of double-checking models is very handy when considering outliers or unique market events (both of which are highly applicable in the real world).

If one starts in R, one is likely to have an easier time setting up the process. One need only read data into the frame, use a standard model in the simple formula language, and then review the output. This process can be much more challenging in Python or other languages (Markham, 2015), as there are a greater number of choices to be made regarding how one analyzes data or constructs the model to run the analyses. Therefore, the software with fewer initial choices, R, is better for those who are just beginning to work with statistical software. As one develops model complexity a shift to other languages or software may be more comfortable, but the basics should be presented in a bare-bones environment to allow for a simple breakdown and higher student understanding of the underlying processes. Even the installation of R is made simple; R Studio (the "de facto IDE" for R) and CRAN are both easily installed atop the basic package. Even the packages from the GitHub website is simple thanks to the devtools package that comes in CRAN (Markham, 2015). This means that students will spend less time on installation and more time working with the program directly and interpreting output, which facilitates faster learning. Additionally, the process of reusing installed packages (booting only the ones needed in a particular session) can save on CPU and RAM, which also speed the

learning process and decrease frustration by reducing or eliminating time spent fixing errors or waiting for lag to subside.

R is also superior in data visualization. The best data visualization is done through R's ggplot2 package (Arratia, 2016, Markham, 2015, Meija, 2013). Once the core principles (and the "grammar" of graphics) are understood by students, ggplot2 becomes a natural extension onto plot building and produces sophisticated and attractive plots. This means that students are able to work more with complex and realistic plots as they advance in the course and their understanding of the software overall (Meija, 2013). Ggplot2, when compared to other packages or software, tends to outperform. While there is some argument for working directly with the basic R capabilities (Nau, 2016), the overwhelming argument is that more complex plots are better created in ggplot2. This package has a higher caliber default setting than the barebones R installation and necessarily makes it easier to work with multiple levels of a variable in one plot rather than examining them side-by-side. This also means that users are able to begin presenting their findings in a method that is accessible to the layperson. This coding- and user-friendly graphic creation can therefore increase the time spent on interpretation and application rather than presentation aid creation. Multivariate plots are made more understandable by adding in a faceting command. This means that each value of a variable receives its own plot in its own color. This color scheme can carry over to continuous variables instead of just working well for categorical variables, and it can contribute to the multi-layered plots (Meija, 2013). This further increases the readability of graphs as the color distinctions make separate categories or treatments more apparent. Meija (2013) does note, however, that ggplot2 could be a poor choice if one is dealing with a large variety of fancy graphics, and to that end Nau (2016) agrees; the best solution for each course or student should be determined individually in order to ensure the highest quality outcome for effort input.

The package fidelity concerns have been greatly addressed by R's Core Development Team (2016). Their site submission guidelines for package acceptance are quite strict. Every submission must go through a form and must then be confirmed by the author of the package. Uploads must adhere to strict formatting guidelines and should be thoroughly debugged by the author before being submitted. There should not be any errors that arise when working with the package (or, if there are errors, they must be ignorable and clearly not interfering with the analyses). Furthermore, updates are expected to be the responsibility of the package owner. Therefore, continued use of a package is left up to the author, and special treatments of the package need to be noted upon submission (such as a lack of backward compatibility). R's Core Development Team (2016) expects that the authors are proactively involved in the maintenance and status of their submissions, and they also expect that patches should be few and far between. To that end, they require that packages submitted to them be tested in Windows (especially if the author is not working from a Windows machine), which could reduce compatibility headaches for instructors so long as one is able to secure a computer lab with working recent updates. They also require that any packages which may be affected by a software update are diligently reported to them and that those who rely on the packages are given ample time to make the necessary changes to their systems. This also means that the author is to be proactive in the application of their package and that they are expected to keep up with the community feedback around its usage. They need to stay informed with what's going on, and to that end, this prevents a large number of poor packages from being submitted.

PERSONAL EXPERIENCE

One of the authors has personal experience with learning separate statistical analysis software. It should be noted that this experience was in the context of statistical applications rather than strictly business or finance, but it is nonetheless a student's perspective on the possible outline of a course in order to ensure students are receiving the maximum amount of exposure to different software. This author believes that Excel, as common software, works as a jumping-off point for a program. To that end, SPSS should be the next logical step. The setup is virtually identical except that SPSS contains built in statistical functions which are not available in the standard installation of Excel. This primes new users to understand the basics of statistics by computer computation in a format highly analogous to one they have seen before. As familiarity with SPSS increases, users are free to work outside of the spreadsheet window in order to directly code the statistical commands. The commands for SPSS are highly forgiving and very straightforward, meaning that little to no prior programming experience is not a hindrance. Moving forward into higher-level software, this author presents the possible path of SPSS to Stata to SAS to R. Stata has more stringent coding than SPSS but is still fairly lenient compared to languages set up to mimic computer coding. This makes Stata the perfect software to transition from a menu-based approach to a code-based one. Once a user has achieved Stata mastery, moving on to SAS is simple. While the formatting of SAS code is much stricter than Stata, the logic is fairly consistent. Plus, upon mastering SAS coding, one can easily shift to coding in R as the programs are quite similar. This experience could be gathered over three or four undergraduate courses, which are certainly manageable for students and instructors.

CONCLUSION

While R may be comparatively new in the data analytics field, this does not mean it should be discounted for undergraduate business students. In spite of its newness, there are several preset packages which solve a large portion of the current finance and data analytics problems. These packages will continue to remain relevant, especially as user-generated data continues to grow. Thus, giving students an understanding of the software in their undergraduate programs primes them both for more advanced courses (either in undergraduate or graduate programs) as well as for the real-world applications and problems that come with using software. There are drawbacks to R, just like there are to any specific program, but these do not outweigh the benefits. R is clearly making a mark on the analytics field, with finance in particular, and should therefore be considered when updating collegiate programs to reflect the modern concerns and approaches to problem solving.

REFERENCES

Arratia, A. (2016, October 7). Getting started with R. Retrieved from: http://computationalfinance.lsi.upc.edu/?page_id=87

Economist 662f. (2014). R v. Stata v. others. [Web log comment]. Retrieved from: https://www.econjobrumors.com/topic/r-v-stata-v-others

Eddelbuettel, D. (2017, April 25). CRAN Task View: Empirical Science. Retrieved from: https://cran.r-project.org/web/views/Finance.html

Markham, K. (2015, February 2). Should you teach Python or R for data science? Retrieved from: http://www.dataschool.io/python-or-r-for-data-science/

Mohamed, O. (2017 February 27). R software for statistical computing: useful in finance? [Web log comment]. Retrieved from: https://www.researchgate.net/post/R_software_for_statistical_computing_useful_in_finance#view=58bd86c196b7e4950c77810e

Moore, B. (2014, January 4). Analyse Your Bank Statements Using R. Retrieved from: https://www.r-bloggers.com/analyse-your-bank-statements-using-r/

Muenchen, B. (2017, March 1). *Data Science Job Report 2017: R Passes SAS, But Python Leaves them Both Behind*. Retrieved from: https://www.r-bloggers.com/data-science-job-report-2017-r-passes-sas-but-python-leaves-them-both-behind/?utm_source=feedburner&utm_medium=email&utm_campaign=Feed%3A+RBloggers+%28R+bloggers%29

Ooi, H. (2011, December 15). R and SAS in Banking. Retrieved from: http://files.meetup.com/1685538/R%20and%20SAS%20in%20Banking.pdf

Rickert, J. (2014, May 8). R and Finance. Retrieved from: http://blog.revolutionanalytics.com/2014/05/r-and-finance.html

R Core Development Team (2016, November 18). *Submission*. CRAN Repository Policy. Retrieved from: https://cran.r-project.org/web/packages/policies.html#Submission

The R Foundation. (2017). The R Project for Statistical Computing. Retrieved from: https://www.r-project.org/

The R User Conference. (2013 July 12). Book of Contributed Abstracts. Retrieved from: https://www.r-project.org/conferences/useR-2013/docs/useR2013_abstract_booklet.pdf)

Trajanov, D. (2017, February 27). R software for statistical computing: useful in finance? [Web log comment]. Retrieved from: https://www.researchgate.net/post/R_software_for_statistical_computing_useful_in_finance#view=58bd86c196b7e4950c77810e

Varma, J. (2016 August 29). Package 'jrvFinance'. Retrieved from: https://mran.microsoft.com/web/packages/jrvFinance/jrvFinance.pdf