Wiley StatsRef:
Statistics Reference Online

Statistical Software

By *Alfred G. Schissler*[1], *Hung Nguyen*[1], *Tin Nguyen*[1], *Juli Petereit*[1], *and Vincent Gardeux*[2]

Keywords: *statistical software, open source, Big Data, visualization, parallel computing, machine learning, Bayesian modeling*

Abstract: This article discusses selected statistical software, aiming to help readers find the right tool for their needs. We categorize software into three classes:Statistical Packages, Analysis Packages with Statistical Libraries, and General Programming Languages with Statistical Libraries. Statistical and analysis packages often provide interactive, easy-to-use workflows while general programming languages are built for speed and optimization. We emphasize each software's defining characteristics and discuss trends in popularity. The concluding sections muse on the future of statistical software including the impact of Big Data and the advantages of open-source languages.

This article discusses selected statistical software, aiming to help readers find the right tool for their needs (not provide an exhaustive list). Also, we acknowledge our experiences bias the discussion toward software employed in scholarly work. Throughout, we emphasize the software's capacity to analyze large, complex data sets (*see* **Big Data**). The concluding sections muse on the future of statistical software.

To aid in the discussion, we classify software into three groups: (i) Statistical Packages, (ii) Analysis Packages with Statistical Libraries, and (iii) General Programming Languages with Statistical Libraries. This structure allows a reader to narrow their search. Within our categories, we highlight commercial versus open-source software. Table 1 provides an overview of the selected software. We now proceed by visiting each of the three categories in turn.

1 Statistical Packages

The subsections below describe several popular and up-and-coming software packages designed for statistical analysis and data science. The typical workflow consists of either a point-and-click interface or a

[1] The University of Nevada, Reno, NV, USA
[2] Ecole Polytechnique Fédé rale de Lausanne, Lausanne, Switzerland

Update based on original article by Edward J. Wegman, Wiley StatsRef: Statistics Reference Online © 2014 John Wiley & Sons, Ltd.

DOI: 10.1002/9781118445112.stat00527.pub2

Table 1. Summary of selected statistical software.

Software	Open source	Classification	Style	Notes
R	Y	Statistical	Programming	Popular, active community
SAS	N	Statistical	Programming	Strong historical following
SPSS	N	Statistical	GUI: Menu/dialogs	Popular in scholarly work
Stata	N	Statistical	GUI: Menu/dialogs	Popular in scholarly work
Minitab	N	Statistical	GUI: Menu/dialogs	Suitable for teaching
Stan	Y	Statistical	Programming	Bayesian modeling
Tableau	N	Statistical	Programming	Data visualization
MATLAB	N	Analysis	Programming	Speedy, becoming less popular
Julia	Y	Analysis	Programming	Speedy, underdeveloped
Python	Y	General	Programming	Versatile, popular
Java	Y	General	Programming	Cross-platform/portable
C++	Y	General	Programming	Fast, low-level

scripting programming language. The users experience an interactive data analysis experience as a result of these workflows.

1.1 R

R[1] began at the University of Auckland, New Zealand, in the early 1990s. Ross Ihaka and Robert Gentleman needed a statistical environment to use in their teaching lab. At the time, their computer labs featured only Macintosh computers that lacked suitable software. Ihaka and Gentleman decided to implement a language based on an S-like syntax[2]. R's initial versions were provided to *Statlib* at Carnegie Mellon University and the user feedback indicated a positive reception.

R's success encouraged its release under the Open Source Initiative (*https://opensource.org/*). Developers released the first version in June 1995. A software system under the open-source paradigm benefits from having "many pairs of eyes" to develop the software. R developed a huge following and it soon became difficult for the developers to maintain. As a response, a 10-member Core group was formed in 1997. The Core team handles any changes to the R source code. The massive R community provides support via online mailing lists (*https://www.r-project.org/mail.html*) and statistical computing forums – such as Talk Stats (*http://www.talkstats.com/*), Cross Validated (*https://stats.stackexchange.com/*), and Stack Overflow (*https://stackoverflow.com/*). Often users receive responses within a matter of minutes.

Since humble beginnings R has developed into a popular, complete, and flexible statistical computing environment that is appreciated by academia, industry, and government. R's main benefits include support on all major operating systems and comprehensive package archives. Further, R integrates well with document formats (such as LATE X(*https://www.latex-project.org/*), HTML, and Microsoft Word) through R Markdown (*https://rmarkdown.rstudio.com/*) and other file formats to enhance literate programming and reproducible data analysis.

R provides extensive statistical capacity. Nearly any method is available as a R package – the trick is locating the software. The *base* package and default included packages perform most standard analyses and computation. If the included packages are insufficient, one can use CRAN (the Comprehensive R Archive Network) that houses nearly 13 000 packages (visit *https://cran.r-project.org/* for more information). To help navigate CRAN, "CRAN Task Views" organizes packages into convenient topics (https://cran.r-project.org/web/views/). For bioinformatics, over 1500 packages reside on Bioconductor[3]. Developers also distribute their packages via git repositories, such as github (*https://github.com/*). For easy retrieval from github, the *devtools* package allows direct installation.

DOI: 10.1002/9781118445112.stat00527.pub2

Most agree that R has extensive capabilities, yet some criticize the software for its slowness and inability to handle Big Data efficiently. However, it has become common place to call C from R to achieve optimal speeds (e.g., via the *Rcpp* package). Moreover, recent advances have increased R's Big Data capabilities in other ways. For example, the *bigMemory* package relaxes memory constraints and the *parallel* package enables easy parallelization across cores. Also, R handles a common Big Data file format – the Hierarchical Data Format (HDF5)[4] using the *rhdf5* package. Notably, native graphical processing unit (GPU) integration is underdeveloped in R compared to other languages.

In summary, R is firmly entrenched as a premier statistical software package. Its open-source, community-based approach has taken the stats scene by storm. R's interactive programming style makes it an attractive and flexible analytic tool. Although, R does lack the speed/flexibility of other languages; yet, for a specialist in statistics, R provides a near-complete solution. We see the popularity of R continuing – however, Big Data's demands could force R programmers to adapt other tools in conjunction with R.

1.2 SAS®

SAS was born during the late 1960s, within the Department of Experimental Statistics at North Carolina State University. As the software developed, the SAS Institute was formed in 1976. Since its infancy, SAS has evolved into an integrated system for data analysis and exploration. The SAS system has been used in numerous business areas and academic institutions worldwide.

SAS provides packages to support various data analytic tasks. The SAS/STAT component contains capabilities one normally associates with data analysis. SAS/STAT supports **analysis of variance (ANOVA)**, **regression**, **categorical data analysis**, **multivariate analysis**, **survival analysis**, **psychometric** analysis, **cluster analysis**, and **nonparametric** analysis. The SAS/INSIGHT package implements visualization strategies. Visualizations can be linked across multiple windows to uncover trends, spot **outliers**, and readily discern subtle patterns. Finally, SAS provides the user with a matrix-programming language via the SAS/IML system. The matrix-based language allows custom statistical algorithm development.

Recently, SAS's popularity has diminished[5], yet it remains widely used. Open-source competitors threaten SAS's previous overall market dominance. Rather than complete removal, we see SAS becoming a niche product in the future. Now, however, SAS expertise remains desired in certain roles and industries.

1.3 SPSS®

Norman H. Nie, C. Hadlai (Tex) Hul, and Dale Brent developed SPSS in the late 1960s. The trio were Stanford University graduate students at the time. SPSS was founded in 1968 and it incorporated in 1975. SPSS became publicly traded in 1993. Now, IBM owns the rights to SPSS. Originally, developers designed SPSS for mainframe use. In 1984, SPSS introduced SPSS/PC+ for computers running MS-DOS, followed by a UNIX release in 1988 and a Macintosh version in 1990. SPSS features an intuitive point-and-click interface. This design empowers a broad user base to conduct standard analyses.

SPSS features a wide variety of analytic capabilities including one for regression, **classification trees**, table creation, **exact tests**, **categorical analysis**, **trend analysis**, **conjoint analysis**, missing value analysis, map-based analysis, and complex samples analysis. In addition, SPSS supports numerous stand-alone products including Amos™ (a **structural equation modeling** package), SPSS Text Analysis for Surveys™ (a survey analysis package utilizing natural language processing (NLP) methodology), SPSS Data Entry™ (a web-based data entry package; *see* **Web Based Data Management in Clinical Trials**), AnswerTree®

DOI: 10.1002/9781118445112.stat00527.pub2

(a market segment targeting package), SmartViewer® Web Server^{TM} (a report generation and dissemination package), SamplePower® (**sample size calculation** package), DecisionTime® and What if?^{TM} (a scenario analysis package for the nonspecialist), SmartViewer® for Windows (a graph/report sharing utility), SPSS WebApp Framework (web-based analytics package), and the Dimensions Development Library (a data capture library).

SPSS remains popular, especially in scholarly work[5]. For many researchers whom apply standard models, SPSS gets the job done. We see SPSS remaining a useful tool for practitioners across many fields.

1.4 Stata®

Stata is a commercial statistical software, developed by William Gould in 1985. StatCorp currently owns/develops Stata and markets the product as a "fast, accurate, and easy to use with both a point-and-click interface and a powerful, intuitive command syntax" (*https://www.stata.com/*). However, most Stata users maintain the point-and-click workflow. Stata strives to provide user confidence through regulatory certification.

Stata provides hundreds of tools across broad applications and methods. Even Bayesian modeling and **maximum likelihood estimation** are available. With its breadth, Stata targets all sectors – academia, industry, and government.

Overall, Stata impresses through active support and development while possessing some unique characteristics. Interestingly, in scholarly work the past decade, only SPSS, R, and SAS have overshadowed Stata[5]. Taken together, we anticipate Stata to remain popular. However, Stata's Big Data capabilities are limited and we have reservations whether industry will adopt Stata over competitors.

1.5 Minitab®

Barbara F. Ryan, Thomas A. Ryan, Jr., and Brian L. Joiner created Minitab in 1972 at the Pennsylvania State University to teach statistics. Now, Minitab Inc. owns the proprietary software. Academia and industry widely employ Minitab[5]. The intuitive point-and-click design and spreadsheet-like interface allow users to analyze data with little learning curve.

Minitab offers import tools and comprehensive set of statistical capabilities. Minitab's features include basic statistics, analysis of variance, fixed and mixed models, regression analyses, measurement systems analysis, and graphics including contour and rotating 3D plots. A full feature list resides at *http://www.minitab.com/en-us/products/minitab/features-list/*. For advanced users, a command-line editor exists. Within the editor, users may customize macros (functions).

Minitab serves its user base well and will continue to be viable in the future. For teaching academics, Minitab provides near immediate access to many statistical methods and graphics. For industry, Minitab offers tools to produce standardized analyses and reports with little training. However, Minitab's flexibility and Big Data capabilities are limited.

1.6 Stan

Stan[6] is a probabilistic programming language for specifying models, most often Bayesian. Stan samples posterior distributions using Hamiltonian Monte Carlo (HMC) – a variant of **Markov Chain Monte Carlo (MCMC)**. HMC boasts a more robust and efficient approach over Gibbs or **Metropolis-Hastings** sampling

DOI: 10.1002/9781118445112.stat00527.pub2

for complex models, while providing insightful diagnostics to assess convergence and mixing. This may explain why Stan is gaining popularity over other Bayesian samplers (such as BUGS[7] and JAGS[8]).

Stan provides a flexible and principled model specification framework. In addition to fully Bayesian inference, Stan computes log densities and Hessians, variational Bayes, expectation propagation, and approximate integration. Stan is available as a command line tool or R/Python interface (RStan and PyStan, respectively).

Stan has ability to become the *de facto* Bayesian modeling software. Designed by thought leader Andrew Gelman and a growing, enthusiastic community, Stan possesses much promise. The language architecture promotes cross-compatibility and extensibility and the general-purpose posterior sampler with innovative diagnostics appeals to novice and advanced modelers alike. Further, to our knowledge, Stan is the only general purpose Bayesian modeler that scales to thousands of parameters – a boon for Big Data analytics.

1.7 Tableau®

Tableau stemmed from visualization research by Stanford University's computer science department in 1999. The Seattle-based company was founded in 2003. Tableau advertises itself as a data exploration and visualization tool, not a statistical software *per se*. Tableau targets the business intelligence market primarily. However, Tableau provides a free, less powerful version for instruction.

Tableau is versatile and user-friendly: providing MacOS and Windows versions while supporting web-based apps on iOS and Android. Tableau connects seamlessly to SQL databases, spreadsheets, cloud apps, and flat files. The software appeals to nontechnical "business" users via its intuitive user interface, but also allows "power users" to develop analytical solutions by connecting to an R server or installing TabPy to integrate Python scripts.

Tableau could corner the data visualization market with its easy-to-learn interface, yet intricate features. We contend that Big Data demands visualization as many traditional methods are not well suited for high-dimensional, observational data. Based on its unique characteristics, Tableau will appeal broadly and could even emerge as a useful tool to supplement an R or Python user's toolkit.

2 Analysis Packages with Statistical Libraries

In this section, we describe two analysis packages with statistical capabilities. A reader interested in a flexible mathematical tool not primarily focused on statistics may consider these options.

2.1 MATLAB

MATLAB begin as FORTRAN subroutines for solving linear (LINPACK) and **eigenvalue** (EISPACK) problems. Cleve Moler developed most of the subroutines in the 1970s for use in the classroom. MATLAB quickly gained popularity, primarily through word of mouth. Developers rewrote MATLAB in C during the 1980s, adding speed and functionality. The parent company of MATLAB, the Mathworks, Inc., was created in 1984.

MATLAB has a substantial user base in government, academia, and the private sector. The MATLAB base distribution allows reading/writing data in ASCII, binary, and MATLAB proprietary format. The data are presented to the user as an array, the MATLAB generic term for a matrix. The base distribution comes with a standard set of mathematical functions including trigonometric, inverse trigonometric, hyperbolic,

DOI: 10.1002/9781118445112.stat00527.pub2

inverse hyperbolic, exponential, and logarithmic. In addition, MATLAB provides the user with access to *cell arrays*, allowing heterogeneous data across the cells and creation analogous to a C/C++. MATLAB provides the user with numerical methods, including **optimization** and **quadrature** functions.

In summary, readers familiar with MATLAB may wish to explore the statistical capabilities, but we caution other users against its adoption. Our caution stems from MATLAB's diminishing popularity[5] – likely due to open-source competitors such as R, Python, and Julia.

2.2 Julia

Julia is a new language designed by Jeff Bezanson *et al*. and was released in 2012[9]. Julia's first stable version (1.0) was released in August 2018. The developers describe themselves as "greedy" – they want a software that does it all. Users no longer would create prototypes in scripting languages then port to C or Java for speed. Below we quote from Julia's public announcement (*https://julialang.org/blog/2012/02/why-we-created-julia*):

> We want a language that's open source, with a liberal license. We want the speed of C with the dynamism of Ruby. We want a language that is homoiconic, with true macros like Lisp, but with obvious, familiar mathematical notation like MATLAB. We want something as usable for general programming as Python, as easy for statistics as R, as natural for string processing as Perl, as powerful for **linear algebra** as MATLAB, as good at gluing programs together as the shell. Something that is dirt simple to learn, yet keeps the most serious hackers happy. We want it interactive and we want it compiled.

Despite the stated goals, we classify Julia as an analysis software at this early stage. Indeed, Julia's syntax exhibits elegance and friendliness to mathematics. The language natively implements an extensive mathematical library. Julia's core distribution includes multidimensional arrays, sparse vectors/matrices, linear algebra, **random number generation**, statistical computation, and **signal processing**.

Julia's design affords speeds comparable to C due to it being an interpreted, embeddable language with a just-in-time compiler. The software also implements concurrent threading, enabling parallel computing natively. Julia integrates nicely with other languages including calling C directly, Python via *PyCall*, and R via *RCall*.

Julia exhibits great promise but remains nascent. We are intrigued by a language that does it all and is easy to use. Yet, Julia's underdevelopment limits its statistical analysis capability. On the other hand, Julia is growing fast with active support and positive community outlook. Coupling Julia's advantages and MATLAB's diminishing appeal, we anticipate Julia to contribute in the area for years to come.

3 General Languages with Statistical Libraries

Here we will highlight the statistics libraries available for three popular general-purpose programming languages. These languages are versatile and fast, but may lack statistical software specialization and ease of use.

3.1 Python

Created by Guido van Rossum and released in 1991, Python is a hugely popular programming language[5]. Python features readable code, an interactive workflow, and an object-oriented design. Python's

DOI: 10.1002/9781118445112.stat00527.pub2

architecture affords rapid application development from prototyping to production. Additionally, many tools integrate nicely with Python, facilitating complex workflows. Python also possesses speed, as most of its high-performance libraries are implemented in C/C++.

Python's core distribution lacks statistical features, prompting developers to create supplementary libraries. Below we detail four well-supported statistical and mathematical libraries: *Numpy*[10], *SciPy*[11], *Pandas*[12], and *Statsmodels*[13].

NumPy is a general and fundamental package for scientific computing[10]. NumPy provides functions for operations on large arrays and matrices, optimized for speed via a C implementation. The package features a dense, homogeneous array called *ndarray*. ndarray provides computational efficiency and flexibility. Developers consider *NumPy* a low-level tool as only foundational functions are available. To enhance capabilities, other statistic libraries and packages use *NumPy* to provide richer features.

One widely used higher-level package, *SciPy*, employs *NumPy* to enable engineering and **data science**[11]. *SciPy* contains modules addressing standard problems in scientific computing, such as mathematical integration, linear algebra, optimization, statistics, clustering, image and signal processing.

Another higher-level Python package built upon *NumPy*, *Pandas*, is designed particularly for data analysis, providing standard models and cohesive frameworks[12]. *Pandas* implements a data type named *DataFrame* – a concept similar to the *data.frame* object in R. DataFrame's structure features efficient methods for data sorting, splicing, merging, grouping, and indexing. *Pandas* implements robust input/output tools – supporting flat files, Excel files, databases, and HDF files. Additionally, *Pandas* provides visualization methods via *Matplotlib*[14].

Lastly, the package *Statsmodels* facilitates data exploration, estimation, and statistical testing[13]. Built at even a higher level than the other packages discussed, *Statsmodels* employs *NumPy*, *SciPy*, *Pandas*, and *Matplotlib*. Many statistical models exist, such as **linear regression**, **generalized linear models**, probability distributions, and **time series**. See *http://www.statsmodels.org/stable/index.html* for the full feature list.

Python's easy-to-learn syntax, speed, versatility all make it a favorite among programmers. Moreover, the packages listed above transform Python into a well-developed vehicle for data science. We see Python's popularity only increasing in the future. Some believe that Python will eventually eliminate the need for R. However, we feel that the future lies in a Python + R paradigm. Thus, R users may well consider exploring what Python offers as the languages have complementary features.

3.2 Java

Java is one of the most popular programming languages (according to the TIOBE index, *www.tiobe.com/tiobe-index/*), partially due to its extensive library ecosystem. Java's design seduces programmers – it's simple, object-oriented, and portable. Java applications run on any machine, from personal laptops to high-performance supercomputers, even game consoles and Internet of things (IoT) devices. Notably, Android (based on Java) development has driven recent Java innovations. Javas "write once, run anywhere" adage provides versatility, triggering interest even at the research level.

Developers may prefer Java for intensive calculations performing slowly within scripted languages (e.g., R). For speed-up purposes, Java's cross-platform design could be even be preferred to C/C++ in certain cases. Alternatively, Java code can be wrapped nicely in a R package for faster processing. For example, the *rJava* package allows one to call java code in a R script and also reversely (calling R functions in Java). On the other hand, Java can be used independently for statistical analysis, thanks to a nice set of statistical libraries.

Popular sources of natively written Java statistical and mathematical functionalities are JSC (Java Statistical Classes) and Apache Commons Math APIs (*http://commons.apache.org/proper/commons-math/*).

DOI: 10.1002/9781118445112.stat00527.pub2

JSC and Apache Commons Math libraries perform many methods including univariate statistics, parametric and nonparametric tests (t-test, chi-square test, Wilcoxon test), random number generation, random sampling/resampling, regression, correlation, linear or stochastic optimization, and clustering.

Additionally, Java boasts an extensive number of **machine learning** packages and Big Data capabilities. For example, Java enables the WEKA[15] tool, the JSAT library[16], and the TensorFlow framework[17]. Moreover, Java provides one of the most desired and useful Big Data analysis tools - Apache Spark[18]. Spark provides Machine Learning support through modules in the Spark MLlib library[19].

As with other discussed software, Java APIs often require importing other packages/libraries. For example, developers commonly use external matrix-operation libraries, such as JAMA (Java Matrix Package, *https://math.nist.gov/javanumerics/jama/*) or EJML (Efficient Java Matrix Library, *http://ejml.org/wiki/*). Such packages allow for routine computation – for example, matrix decomposition and dense/sparse matrix calculation. JFreeCHart enables data visualization by generating **scatter plots**, **histograms**, barplots, etc. Recently, more popular web javascript libraries are slowly replacing them such as Plot.ly (*https://plot.ly/*), Bokeh (*bokeh.pydata.org*), D3[20], or Highcharts (*www.highcharts.com*).

As outlined above, Java could serve as a useful statistical software solution, especially for developers familiar with it or have interest in cross-platform development. We would then recommend its use for seasoned programmers looking to add some statistical punch to their desktop, web, and mobile apps. For the analysis of Big Data, Java offers some of the best machine learning tools available.

3.3 C++

C++ is a general purpose, high-performance programming language. Unlike other scripting languages for statistics such as R and Python, C++ is a compiled language – adding complexity (such as memory management) and strict syntax requirements. As such, C's design may complicate prototyping. Thus, data scientists typically turn to C++ to optimize/scale a developed algorithm at the production level.

C++'s standard libraries lack many mathematical and statistical operations. However, since C++ can be compiled cross-platform, developers often interface C++ functions in from different languages (e.g., R and Python). Thus, C++ can be used to develop libraries across languages, offering impressive computing performance.

To enable analysis, developers created mathematical and statistical libraries in C++. The packages often employ of BLAS (Basic Linear Algebra Subprograms) libraries, written in C/Fortran and offer numerous low-level, high-performance linear algebra operations on numbers, vectors, and matrices. Some popular BLAS-compatible libraries include Intel Math Kernel Library (Intel MKL)[21], Automatically Tuned Linear Algebra Software (ATLAS)[22], OpenBLAS[23], and Linear Algebra PACKage (LAPACK)[24].

Among the C++ libraries for mathematics and statistics built on top BLAS, we detail three popular, well-maintained libraries: Eigen[25], Armandillo[26], and Blaze[27] below:

Eigen is a high-level, header-only library developed by Guennebaud and Jacob[25]. *Eigen* provides classes dealing with vector types, arrays, and dense/sparse/large matrices. It also supports matrix decomposition and geometric features. *Eigen* uses Single Instruction Multiple Data vectorization to avoid dynamic memory allocation. *Eigen* also implements extra features to optimize the computing performance, including unrolling techniques and processor-cache utilization. *Eigen* itself does not take much advantage from parallel hardware, currently supporting parallel processing only for general matrix–matrix products. However, since *Eigen* uses BLAS-compatible libraries, users can utilize external BLAS libraries in conjunction with *Eigen* for parallel computing. Python and R users can call *Eigen* functions using *minieigen* and *RcppEigen* packages.

The National ICT Australia (NICTA) developed the open-source library *Armadillo* to facilitate science and engineering[26]. *Armadillo* provides a fast, easy-to-use matrix library with MATLAB-like syntax.

DOI: 10.1002/9781118445112.stat00527.pub2

Armadillo employs template meta-programming techniques to avoid unnecessary operations and increase library performance. Further, *Armadillo* supports 3D objects and provides numerous utilities for matrices manipulation and decomposition. *Armadillo* automatically utilizes Open Multi-Processing (OpenMP)[28] to increase speed. Developers designed *Armadillo* to provide a balance between speed and ease-of-use. *Armadillo* is widely used for many applications in machine learning, pattern recognition, signal processing, and bioinformatics. R users may call *Armadillo* functions through the *RcppArmadillo* package.

Blaze is a high-performance math library for dense/sparse arithmetic developed by Klaus Iglberger *et al*.[27] *Blaze* extensively uses LAPACK functions for various computing tasks, such as matrix decomposition and inversion, providing high-performance computing. *Blaze* supports High-Performance ParalleX (HPX)[29] and OpenMP to enable parallel computing.

The difficulty to develop C++ programs limits its use as a primary statistical software package. Yet, C++ appeals when a fast, production-quality program if desired. Therefore, R and Python developers may find C++ knowledge beneficial to optimize their code prior to distribution. We see C/C++ as the standard for speed and, as such, an attractive tool for Big Data problems.

4 The Future of Statistical Computing

Big Data will shape the statistical-computing future. Big Data analytics will increasingly require optimized code, parallelization, and cloud/cluster computing support. Likely one tool will not meet all the demand and therefore cross-compatibility standards must be developed. All statistical software will likely require GPU integration to accelerate analyses. Algorithm developments, especially in high-dimensional statistics and Bayesian modeling, are also needed to handle the influx of data. Moreover, data visualization will become increasing important (including virtual reality) for large, complex data sets where conventional inferential tools are suspect or without use.

The advantages of open-source, community-based development have been emphasized throughout – especially in the scholarly arena and smaller businesses. The open-source paradigm enables rapid software development with limited resources. However, commercial software with dedicated support services will appeal to certain markets, including medium-to-large businesses.

5 Concluding Remarks

We have attempted to evaluate the current statistical software landscape. Admittedly, our treatment has been focused by our experience. We have, however, attempted to be fair in our appraisal.

We have also provided a limited prognostication with regard to the statistical-software future by identifying issues and applications likely to shape software development. We realize, of course, the future is usually full of surprises and only time will tell what actually occurs.

Acknowledgments

The work of the fourth author, Juli Petereit, was partially supported by the NIH grant (5U54GM104944) through the National Institute of General Medical Sciences (NIGMS) under the Institutional Development Award (IDeA) program. The article has been updated, authored originally by Edward J. Wegman and Jeffrey L. Solka. We thank the Wiley StatsRef staff and editor of this article, Walter W. Piegorsch, for their expertise and support.

DOI: 10.1002/9781118445112.stat00527.pub2

Related Articles

Statistics and Computing; **Parallel Computing: Statistical and Environmetric Uses**; **Software for Survival Analysis**; **Software for Statistical Analyses**; **Analysis of Complex Survey Data, Misuse of Standard Statistical Procedures**; **Software for Sample Survey Data**; **Software for Genetics/Genomics**; **Software for Genetic Epidemiology**; **Software for Behavioral Genetics**; **Software, Epidemiological**; **Software, Biostatistical**.

References

[1] R Core Team (2018) *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

[2] Becker, R.A., Chambers, J.M., and Wilks, A.R. (1988) *The New S Language: A Programming Environment for Data Analysis and Graphics*, vol. 1, Wadsworth & Brooks, Monterey, USA.

[3] Gentleman, R.C., Carey, V.J., Bates D.M., *et al*. (2004) Bioconductor: open software development for computational biology and bioinformatics. *Genome Biol.*, **5**, R80.

[4] Koziol, Q. and Robinson, D., HDF5. *https://www.osti.gov/doecode/biblio/9801*.

[5] Muenchen, R.A. (2017) The Popularity of Data Science Software. *https://r4stats.com/articles/popularity/* (accessed 3 December 2018).

[6] Carpenter, B., Gelman, A., Hoffman, M., *et al*. (2017) Stan: a probabilistic programming language. *J. Stat. Softw.*, **76**, 1–32.

[7] Thomas, A., Spiegelhalter, D.J., andGilks, W.R. (1992) BUGS: a program to perform Bayesian inference using Gibbs sampling. *Bayesian Stat.*, **4**, 837–842.

[8] Plummer, M. (2005) JAGS: just another Gibbs sampler, in: Proceedings of the 3rd International Workshop on Distributed Statistical Computing (DSC 2003),

[9] Bezanson, J., Karpinski, S., Shah, V.B., and Edelman, A. (2012) Julia: a fast dynamic language for technical computing, arXiv preprint arXiv:1209.5145.

[10] Oliphant, T.E. (2010) *Guide to NumPy*, vol. 1, Trelgol Publishing, USA.

[11] Oliphant, T.E. (2007) SciPy: Open source scientific tools for Python. *https://www.scipy.org/* (accessed 3 December 2018).

[12] McKinney, W. (2011) pandas: a foundational Python library for data analysis and statistics, in *Python for High Performance and Scientific Computing*, 1–9.

[13] Seabold, S. and Perktold, J. (2010) Econometric and Statistical Modeling with Python Skipper Seabold 1 1. 9th Python in Science Conference.

[14] Hunter, J.D. (2007) Matplotlib: a 2D graphics environment. *Comput. Sci. Eng.*, **9**, 99–104.

[15] Frank, E., Hall, M.A., and Witten, I.H. (2016) The WEKA workbench, *Data mining: Practical machine learning tools and techniques, 4*

[16] Raff, E. (2017) JSAT: java statistical analysis tool, a library for Machine Learning. *J. Mach. Learn. Res.*, **18**, 1–5.

[17] Abadi , M., *et al*. (2016) Tensorflow: a system for large-scale machine learning. In *OSDI*, vol. 16, pp. 265–283.

[18] Zaharia, M., Xin, R.S., and Wendell, P. (2016) Apache Spark: a unified engine for big data processing. *Commun. ACM*, **59**, 56–65.

[19] Meng, X., Bradley, J., and Yavuz, B. (2016) [seminal] MLlib: Machine Learning in Apache Spark. *J. Mach. Learn. Res.*, **17**, 1–7.

[20] Bostock, M., Ogievetsky, V., and Heer, J. (2011) D3 data-driven documents. *IEEE Trans. Vis. Comput. Graph.*, **17**, 2301–2309.

[21] Intel (2007) Intel® Math Kernel Library Reference Manual. *https://software.intel.com/en-us/mkl* (accessed 03 December 2018).

[22] Whaley, R.C. andPetitet, A. (2005) Minimizing development and maintenance costs in supporting persistently optimized BLAS. *Softw.Pract. Exper.*, **35**, 101–121.

[23] Xianyi, Z., Qian, W., and Chothia, Z. (2014) OpenBLAS. *http://xianyi.github.io/OpenBLAS* (accessed 03 December 2018).

[24] Anderson, E., Bischof, C., Demmel, J., *et al*. (1990) Prospectus for an Extension to {LAPACK}: {A} Portable Linear Algebra Library for High-Performance Computers. Proceedings of the 1990 ACM/IEEE conference on Supercomputing, 2–11, no. CS-90-118, IEEE Computer Society Press.

[25] Guennebaud, G. and Jacob, B. (2010) Eigen v3. *http://eigen.tuxfamily.org* (accessed 03 December 2018).

[26] Sanderson, C. and Curtin, R. (2016) Armadillo: a template-based C++ library for linear algebra. *J. Open Source Softw.*, **1**, 1–7.

DOI: 10.1002/9781118445112.stat00527.pub2

[27] Iglberger, K., Hager, G., Treibig, J., and Rüde, U. (2012) High performance smart expression template math libraries, in *Proceedings of the 2012 International Conference on High Performance Computing and Simulation, HPCS 2012*, IEEE, pp. 367–373.

[28] Dagum, L. andMenon, R. (1998) OpenMP: an industry standard API for shared-memory programming. *IEEE Comput. Sci. Eng.*, **5**, 46–55.

[29] Heller, T., Diehl, P., Montreal, P.M., *et al*. (2017) HPX An open source C++ Standard Library for Parallelism and Concurrency. 2017 Workshop on Open Source Supercomputing, p. 5.

Further Reading

Cti Statistics (2004). Alphabetical List of Reviews. *www.stats.gla.ac.uk/cti/activities/reviews/alphabet.html* (accessed 03 December 2018).

de Leeuw, J. (2004). *J. Stat. Softw. www.jstatsoft.org*.

Scientific Computing World (2004). Software Reviews from Scientific Computing World. *www.scientific-computing.com/reviews.html* (accessed 03 December 2018).

DOI: 10.1002/9781118445112.stat00527.pub2