NIH Public Access
Author Manuscript
J Stat Softw. Author manuscript; available in PMC 2011 June 9.

Published in final edited form as:
J Stat Softw. 2009 April 1; 30(3): 1–19.

SOCR Analyses: Implementation and Demonstration of a New Graphical Statistics Educational Toolkit

Annie Chu, **Jenny Cui**, and **Ivo D. Dinov**
The SOCR Resource, Department of Statistics, and Center for Computational Biology, 8125 Mathematical Science Bldg. University of California, Los Angeles, Los Angeles, CA 90095-1554, United States of America, Telephone: +1/310/825-8430, Fax: +1/310/206-5658, URL: http://www.SOCR.ucla.edu/

Ivo D. Dinov: dinov@stat.ucla.edu

Abstract

The web-based, Java-written **SOCR** (Statistical Online Computational Resource) tools have been utilized in many undergraduate and graduate level statistics courses for seven years now (Dinov 2006; Dinov *et al.* 2008b). It has been proven that these resources can successfully improve students' learning (Dinov *et al.* 2008b). Being first published online in 2005, **SOCR** Analyses is a somewhat new component and it concentrate on data modeling for both parametric and non-parametric data analyses with graphical model diagnostics. One of the main purposes of **SOCR** Analyses is to facilitate statistical learning for high school and undergraduate students. As we have already implemented **SOCR** Distributions and Experiments, **SOCR** Analyses and Charts fulfill the rest of a standard statistics curricula. Currently, there are four core components of **SOCR** Analyses. Linear models included in **SOCR** Analyses are simple linear regression, multiple linear regression, one-way and two-way ANOVA. Tests for sample comparisons include t-test in the parametric category. Some examples of **SOCR** Analyses' in the non-parametric category are Wilcoxon rank sum test, Kruskal-Wallis test, Friedman's test, Kolmogorov-Smirnoff test and Fligner-Killeen test. Hypothesis testing models include contingency table, Friedman's test and Fisher's exact test. The last component of Analyses is a utility for computing sample sizes for normal distribution. In this article, we present the design framework, computational implementation and the utilization of **SOCR** Analyses.

Keywords

web resources; Java applets; statistical analysis; education; cooperative learning; collaborative learning; distance education; telelearning; distributed learning environments; improving classroom teaching; interactive learning environments

1. Introduction

The Statistics Online Computational Resource http://www.SOCR.ucla.edu/, an NSF-funded project, provides integrated tools for probability and statistics education (Dinov 2006; Dinov *et al.* 2008a). This project is to provide a free, web-based and browser-independent suite of tools; to develop a well-designed, extensible and open-sources library; to introduce a graphical user interface (GUI) to statistical resources; and to present an integrated framework for course-material, simulation and computation. An experimental study conducted with UCLA undergraduate students, showed significantly higher performance of the **SOCR** treatment group versus the control group using traditional teaching methods. This study also shows that student performance in the treatment group was more homogeneous. The treatment group reported higher satisfaction and found the **SOCR** treatment courses

NIH-PA Author Manuscript
NIH-PA Author Manuscript
NIH-PA Author Manuscript

NIH-PA Author Manuscript

NIH-PA Author Manuscript

NIH-PA Author Manuscript

more interesting compared to corresponding control groups using traditional pedagogical methods (Dinov *et al.* 2008b).

In the earlier years of SOCR project, the following SOCR components were developed: Experiments, Distributions and Games (Dinov 2006; Dinov *et al.* 2008b). On the other hand, three other components, ***SOCR** Analyses*, *Charts* and *Modeler*, are relatively new. In this article, we will discuss **SOCR** Analyses, with brief demonstrations in graphical functions of Charts and Modeler. The *motivation* of **SOCR** Analyses development is to complete a standard statistics curriculum. As described above, we already have developed probability materials for such curriculum. In short, with **SOCR** Analyses, we now have tools for hypothesis testing of both parametric and non-parametric models, for data modeling (linear regression and ANOVA), and for computation of power and sample size. Additionally, **SOCR** Charts and Modeler, although not part of **SOCR** Analyses and thus not a main topic of this paper, provide visual demonstration of descriptive statistics. These tools all together have built a complete standard curriculum. The **SOCR** Analyses *innovation* extends of the core **SOCR** development principles: (1) it is web-based so it is available to everyone with no cost; (2) it has virtually all materials needed for an introductory statistics class to reduce instructors' preparation time; (3) it is graphics-based and therefore lowers the learning curve; (4) the source code is available for interested researchers to use and extend. One of the goals is to make available a collection of all the commonly used statistics analyses accessible in one package; all with easy-to-follow GUI. Currently, for statistics analysis tools, students and scholars have to mostly rely on tools that require either a license or a steep learning curve, even for users with some statistical background. There are also some web-based tools available that do not provide code base or examples, which limits the options for code modification and broad use for variety of projects. Many other web-based tools scattered in the cyber-space are applicable for merely a single analysis, not integrated as collections and demand more of the user's time and effort on tool searching and interoperability (Dinov 2008).

To summarize this article, we will present a brief introduction to the **SOCR** Analyses' design framework and implementation in Section 2 and provide documentation to readers who are interested in expending the **SOCR** open source for their own work. Section 3 furthermore presents utilization of **SOCR** as external libraries. Section 4 provides comparisons of **SOCR** versus several contemporary software packages. While Section 2 and Section 3 provide information written for programmers, end-users may briefly review or skip them and jump to section Section 4 or Section 5 directly. Section 5 provides a gallery of several examples and will serve as a tutorial for students and educators. In this article, we put a much heavier emphasis on the examples since the population of **SOCR** end-users is much larger than that of developers. Conclusion and discussion of future work follow in Section 6.

Another **SOCR** facility, a newly developed Interactive Hypergraph **SOCR** viewer provides a graphical overview of the entire **SOCR** infrastructure and relationship among all the computational tools, learning materials and instructional resources. The user may browse all **SOCR** materials in a hierarchical tree map, including the **SOCR** Analyses, the documentation of various analyses, activities and hands-on demonstrations. Starting from a macro viewpoint, the user can navigate the **SOCR** Resource hierarchy and access desired materials by double-clicking on a tree node. The Interactive Hypergraph **SOCR** viewer is available at: http://www.SOCR.ucla.edu/SOCR_HT_ResourceViewer.html.

NIH-PA Author Manuscript

NIH-PA Author Manuscript

NIH-PA Author Manuscript

2. Design framework and implementation

All **SOCR** tools, GUIs and libraries are developed and coded in Java. Java is a platform independent programming language; the advantage of using Java Applets is to enable the user to access these resources as computational libraries (JAR files) or applet GUIs from anywhere on the Web, *without* the need to install any software, register or login. Therefore, **SOCR** is browser, hardware and software independent. This is especially helpful for undergraduate and informal learners, since such audiences frequently come from various academic backgrounds and may not be comfortable, or allowed, to install any programs on their computers.

The **SOCR** software itself is quite easy to learn, whether one is sitting in a laboratory class guided by an instructor, or one is just self-learning by following our tutorial pages. This capability for quick exposure and testing significantly reduces the steepness of the learning curve and improves the learning experiences. There are no software prerequisites or authentications necessary for using **SOCR** Analyses, other than having access to a Java-enabled browser. This enables quick and reproducible teaching and learning (Dinov 2008). In addition, not only the software is available through the Internet, we provide a ZIP file on our site so that the user can retrieve the binary code that allows offline use of the Analyses tool on local computers.

We apply OOP (object-oriented programming) to the development of **SOCR** Analyses. The purpose of OOP is to achieve high cohesion and low coupling, for the ease of code development and maintenance. The **SOCR** Analyses component code follows the basic OOP principle of *model-view-controller* (MVC). Figure 1 depicts the design framework and relationships of MVC in terms of **SOCR** Analysis design. The *model* part does the statistical analyses computation; the *view* part is the applet GUI; and the *controller* part goes in between them, dispatching the request from the view to the model and receiving analysis results. An object instance is created to handling the data-carrying mechanism, by utilizing Java collections, HashMap and TreeMap, as its major aggregates. In the opposite direction, from view to model, an object of Java Collection is utilized (Figure 1). Validation is implemented for different data types and for different models. For example, “quantitative” data type has to be used in simple regression analysis, whereas “factor” type is appropriate for ANOVA regressors. The controller part fetches data from the applet and constructs an object to pass forward to the model part; it also receives an object that carries the analysis results, from the model and prints output in the applet results panel. As an alternative, a programmer can implement the application of the view part instead of using the applet by either getting the **SOCR** Analyses binary libraries or source code. See Section 3 for more description on the statistics libraries.

Additional information explaining the code, design, namespace and code examples may be obtained at the **SOCR** Wiki pages. In this paper, we present a few examples on how polymorphism is used in **SOCR** Analyses. For instance, an interface named *Analysis* is provided that declares all the general functions and member-variables, in terms of how a class should be called to perform a task. Child classes are implemented for simple linear regression, one-way ANOVA, survival analysis and Kruskal-Wallis test, etc. Currently, there are 21 child classes with their own implementation. Figure 2 depicts the inheritance relationships. Similarly, in order to carry around the analysis results, a parent class is created with general attributes, and sub-classed by those that concretely specify the result information of each analysis model. At the GUI part, classes of each analysis are extended from JApplet and a **SOCR** class Analysis, to present specific user interface at the front end. UML diagrams are created along with the development of the **SOCR** code base. Interested developers may find them available at: http://www.SOCR.ucla.edu/docs/SOCR_Core.jpg at

NIH-PA Author Manuscript

NIH-PA Author Manuscript

NIH-PA Author Manuscript

the core level and http://www.SOCR.ucla.edu/docs/SOCR_PackageLevel.jpg at the package level.

To introduce other statistical analysis tools, by extending **SOCR** Analyses, one needs to follow the following steps. We demonstrate the process of extending the **SOCR** Analyses using the recently added Fliger-Killeen homogeneity of variances test (Conover *et al.* 1981).

1. A new class in edu.stat.ucla.SOCR.analyses.model package needs to be added (this is the model package in Figure 1. It is named as FlignerKilleen and is a sub-class of Analysis class under the same package, just like any class in Figure 3, which resembles Figure 2. This class receives data from the controller (as in Figure 1) and does the appropriate statistical analysis. Then this class creates a Result object (as in Figure 2) to store the FlignerKilleen test results. The body of the FlignerKilleen class needs to be synchronized with the classes discussed below.
2. Another new class needs to be created in edu.stat.ucla.SOCR.analyses.result. It is named as FlignerKilleenResult and is a sub-class of AnalysisResult class (which is a concrete sub-class of Result class) under the same package, like any class in Figure 3, which resembles Figure 2. Each of the result attributes (e.g., test statistic, *p* value) have their own getter methods.
3. A new class in edu.stat.ucla.SOCR.analyses.gui needs to be added. For clarity, it is named as FlignerKilleen and is a sub-class of Analysis class (a sub-class of JApplet) under the same package, like any class in Figure 3, which resembles Figure 2. This class handles the work of receiving inputs from the user and presenting results back to the user.
4. The connections between the above three classes are implemented as follows: When a request arrives from the controller in Figure, 1 the AnalysisType class (under edu.stat.ucla.SOCR.analyses.model) needs to know what sub-class under the model package to delegate the necessary tasks to. Therefore, the AnalysisType class needs to be modified accordingly. Each class under the model package has validation code verifying its interface synchronization with its callers (programmed in the controller). In other words, if the caller looks for Two-Way ANOVA but Fligner-Killeen testing is being called by mistake, then an appropriate Exception will be thrown, to prevent mismatch between GUI request and model calculation.
5. Additionally, if the developer needs to add static examples, a sub-class under edu.stat.ucla.SOCR.analyses.example needs to be added, with appropriate numerical examples. Since analysis-specific examples are always displayed in the control-panel, the Analysis class under edu.stat.ucla.SOCR.analyses.gui needs to be modified to call for these concrete examples.

Note that, if only the code in the model package is provided, then the new analysis may be used as external computational library, but will not be accessible via the Analyses GUI. The components extended in the model package are independent of their counterparts utilized in the GUI package.

The **SOCR** Analyses tool relies on implemented code of other **SOCR** components (e.g., **SOCR** Distribution, **SOCR** Experiments, etc.) to handle probability and modeling calculations. Many of these previously built utility classes are reused in Analyses. For the graphical display of modeling diagnostic, we also employ **SOCR** Charts, a **SOCR** house-tailored component based on the **JFreeChart** project (Gilbert 2009).

One of the main purposes of **SOCR** is to improve the undergraduate statistics learning experiences. Thus, we develop our tools following the topics taught in typical undergraduate

NIH-PA Author Manuscript

NIH-PA Author Manuscript

NIH-PA Author Manuscript

statistics classes. The currently available tools include several categories: linear models, comparisons of samples for both parametric and non-parametric testing, categorical analysis, power analysis, survival analysis using the Kaplan-Meyer model, to name a few. The implementation of all 21 current **SOCR** Analyses is summarized in Table 1.

We provide the source code as open source http://SOCR.googlecode.com/ for the user who will use the **SOCR** Analyses functionalities as an application or computational libraries instead of interfacing these resources via a browser GUI. Code examples may be found on the **SOCR** Wiki Page http://wiki.stat.ucla.edu/socr/index.php/SOCR_Docs. API documentation is also provided there. New information is regularly uploaded online.

3. SOCR Analyses as external statistics libraries

In addition to the usage of the **SOCR** Analyses via the **SOCR**/HTML graphical user interface, Figure 4 (top), there are two explicit ways of using the **SOCR** Analyses tools as external libraries for statistical computing. The first one involves calling the **SOCR** JAR files from any language via the Java native interface (JNI) methods. The complete **SOCR** application programming interface (API) required for this is available online at http://SOCR.ucla.edu/docs/. The second approach is to use the **SOCR** Analyses command-line interface to invoke the **SOCR** statistical computing libraries.

Figure 4 (bottom) demonstrates the later approach. In this example, we are using the **LONI** Pipeline environment (Rex *et al.* 2003) to demonstrate the power of the **SOCR** command-line interface. The **LONI** Pipeline is a graphical workflow generation and execution environment that allows the integration of disparate and diverse computational resources via a platform agnostic, distributed and light-weight API interface. The left component of this pipeline workflow applies automated wavelet-based analysis of the 3D MRI volumes of 2 groups of 9 each. This is part of the automated wavelet analysis of image registration tool (Dinov and Sumners 2001;Dinov *et al.* 2002), which a common analysis method to quantify volumetric warping in terms of the displacement severity and alignment fidelity. The results at the end (bottom-left) represent text files with summary statistics for each subject. The right part of the pipeline contains 6 **SOCR** Analyses modules (nodes) each of which is executed via the Analyses command-line interface. The top 2 **SOCR** Analyses include parametric and non-parametric independent sample tests (comparing the wavelet analyses between the two populations). The bottom 4 modules demonstrate a Chi-Square goodness of fit test of the two group wavelet analyses, as well as pared sample tests (assuming the two populations include the same subjects scanned at two different time points). The mechanics of the Pipeline XML module description enables the specification of input (e.g., files, flags, controls, etc.) and output (e.g., files, strings, values, etc.) parameters for each tool. These parameters are the dots on top, bottom or to the side of each node, and allow the user to provide workflow controls. In the case of **SOCR** Analyses modules, these parameters provide the command-line specification of data-files and header-flags.

Figure 4 illustrates the construction and real-time execution of a brain image analysis work-flow, which includes 6 **SOCR** Analyses tools (right-most modules). Note that some of the **SOCR** analyses are running and some are complete, as indicated by the blue and green colors, respectively, of the circles surrounding each of the nodes. In Figure 4 (bottom), the workflow involved the wavelet analysis of human brain imaging data (left branch of the work-flow) and the subsequent model-fitting and statistical analyses of the results using the **SOCR** Analyses command-line interface (right branch). Notice the completion times on all modules (nodes) and the total execution time of 8’13. The dashed-green-circles surrounding each node indicate error-free execution of the entire pipeline workflow.

NIH-PA Author Manuscript

NIH-PA Author Manuscript

NIH-PA Author Manuscript

4. Comparisons with other tools

Different software tools target different audiences and provide appropriate balance of computational performance and efficiency in learning experience. Frequently, the cost of a computational tool and the easy of access may be major concerns, especially for educators and students. Here we compare **SOCR** to **Rweb** (Banfield 2004), **Rpad** (Short 2006) and Minitab (Minitab Inc. 2008). **Rweb**, **Rpad** are based on R (R Development Core Team 2008, http://www.R-project.org/), one tool package that provides a tremendous volume of statistical analyses, but also requires basic knowledge in R syntax. This might be intuitive for users with prior coding experience, but may be challenging for beginners and learners without coding experience. Aiming for this type of target audiences, **SOCR** requires only rudimentary skills of computer operations and web browsing, but no background in programming or code writing. Additionally, **SOCR** binaries reside on the **SOCR** server, however, these are executed on the user client machine. **SOCR** Analyses may be most useful to high school students and non-mathematics college majors. This contrasts with majority of R, and its extended tools like **Rweb** and **Rpad** users, who need considerable technical background. Generally, R users have much stronger computational background and may choose to run **Rweb** or **Rpad**, on the web or R program in a local shell, which requires prior software installation.

While **Rweb** is available on the web for use, it requires communication with a remote server and the computational time varies according to network speed and server load. **SOCR** Analysis was designed to run directly in a browser on the clientŠs machine.[3] For many users, having the computational work done locally enjoys the benefit of speed and independence from the network.

Many statistical packages require fees, licenses or registration. Minitab is a tool package that provides great graphical user interface, tailored for business usage, with an affordable price for students. **SOCR**'s free access benefits educators and students alike. More comparisons of **SOCR** against **Rweb**, **Rpad** and Minitab are listed in Table 2.

5. SOCR Analyses applet and demo of Charts and Modeler

In this section, we will demonstrate several examples from **SOCR** Analyses. Step-by-step tutorial on usage of **SOCR** Analyses is also given. Due to limitation of space, we only provide five examples here. Adequate operation details are provided for each example so that the readers may start with any example. Furthermore, since **SOCR** Analyses do not have the functionalities on displaying data graphically with bar charts, pie charts or histograms, we include brief demonstrations of such functionalities with two other **SOCR** components: Charts and Modeler.

5.1. Multiple linear regression example

A commonly taught topic in a standard statistics, Multiple Linear Regression, is used as the first example to demonstrate the operation of **SOCR** Analyses. The data used for the demonstration is from Jennrich (1995) (1995, Chapter 4, page 105). **SOCR** Analyses applet may be accessed online at http://SOCR.ucla.edu/htmls/SOCR_Analyses.html. The pull-down combo box, Figure 5, allows for selecting the desired analysis model. Note that there are buttons at the left panel to provide online help, file loading and screenshot facility. Options on precision of calculations are also included in this panel. See instructions below for details. In most statistical analysis protocols, it may be natural to first obtain data before final decision on the choice of models is made. However, most educational demonstrations of model utilization begin with selecting the type of analyses first, and then providing test

NIH-PA Author Manuscript

NIH-PA Author Manuscript

NIH-PA Author Manuscript

data that can be analyzed according to the model specifications. In **SOCR** Analyses, we use this latter type of operation flow, which common is most textbook settings

SOCR Analyses selection buttons include:

- About: This button directs to a web site to retrieve explanation of the specified analysis technique (e.g., http://mathworld.wolfram.com/).
- Help: This button retrieves general statistical information from http://wiki.stat.ucla.edu/socr/index.php/Help_pages_for_SOCR_Analyses.
- Snapshot: This button functions as an applet snapshot similar to “print screen”.
- Copy/Paste: This button assists with data copying/pasting. Note that the key sequence Apple-C and Apple-V (Apple/Macintosh) and Ctrl-C and Ctrl-V (Windows) may not work in the applet. Apple-Macintosh platforms may also allow text-selection followed by drag-and-drop to copy data or results in or out of **SOCR** Anslyses.
- File Open: For uploading data from a local computer.

Now we look at the right panel where we do most of the operations. Taking the Multiple Linear Regression as an example, we go through the operations step-by-step.

1. Select the Multiple Linear Regression model from the left panel’s combo box. The user may also change the number of digits to output for the results (Figure 5).
2. Click of the Data tab to display the data spreadsheet (Figure 6). Data may be uploaded from a file, manually entered or randomly generated using the Example buttons.
3. Before the data are analyzed, the user must click on the Mapping tab. The mapping panel is for selection and assignment of variables (Figure 6).
4. Click the Calculate button to complete the calculations. The user can then go to the Result tab to inspect the results of the analysis (Figure 6).
5. The Graph tab displays plots tailored for this particular analysis. The multiple linear regression currently has the following charts available: scatter plots of the data, residual on fit, residual on covariate, and normal quantile-quantile plot (Figure 6).

5.2. Contingency table example

The operation of getting contingency table results is quite intuitive and flexible.

1. Select contingency table from the left panel’s combo box, like in Figure 5.
2. Click the Input tab for a panel, like in Figure 7.
3. According to the data, the user may select the number of rows and columns and change the text of row and column headings. The significance level may also be changed. Then the data needs to be entered.
4. Click the Calculate button to complete the analysis. The user may then click on the Result tab to inspect the results (Figure 7).

5.3. Two-way ANOVA example

The operation of two-way ANOVA example is very similar to multiple linear regression:

1. Select Two-Way ANOVA from the left panel’s combo box like in Figure 5.

NIH-PA Author Manuscript

NIH-PA Author Manuscript

NIH-PA Author Manuscript

2. Click on the Data tab to get the data spreadsheet (Figure 8).
3. Before the data are analyzed, the user must select and assign the variables in the Mapping tab (Figure 8).
4. Click the Calculate button to completes the analysis. The user may then click on the Result tab to inspect the results (Figure 8).
5. The Graph tab displays plots tailored for this particular analysis. Two-way ANOVA currently has the following charts available: scatter plots of the data, residual on fit, residual on covariate, and normal quantile-quantile plot (Figure 8).

5.4. Fligner-Kileen example

The operations of Fligner-Kileen Example are similar:

1. Select the Fligner-Kileen Analysis from the left panel's combo box like in Figure 5.
2. Click on the Data tab and select example data (Figure 9).
3. Click the Mapping tab to select two columns of data. The operation is very similar to the examples above on multiple linear regression and thus not shown here.
4. Click the Calculate button to complete the calculation. The user may then click on the Result tab to inspect the results (Figure 9).

5.5. Komogorov-Smirnoff example

Although the statistical model is completely different from multiple linear regression or two-way ANOVA, the operation is very similar:

1. Select the Komogorov-Smirnoff Analysis from the left panel's combo box like in Figure 5.
2. Click on the Data tab to select or upload data.
3. Click on the Mapping tab and select the variables (Figure 10).
4. Click on the Calculate button to complete the analysis. The user can then click on the Result tab to inspect the results (Figure 10).
5. The Graph tab displays plots tailored for this particular analysis. Komogorov-Smirnoff analysis currently has the following charts available: cumulative factor plot for before and after log is taken, for all selected variables (Figure 10).

5.6. Descriptive statistics with SOCR Charts and Modeler

Although **SOCR** Analyses does not include presentation of descriptive statistics methods, these utilities may be found in **SOCR** Charts (Figure 11), e.g., box plots, pie charts, time plots, etc. Histogram construction and distribution model-fitting are demonstrated in **SOCR** Modeler. The Modeler enables random sampling from a large number of distributions, plotting sample histograms, and various model fitting functions. These figures only briefly demonstrate the **SOCR** Charts and Modeler, and more information is available online.

6. Conclusions and future work

The **SOCR** resources have been successfully utilized in statistics education (Christou *et al.* 2007; Dinov *et al.* 2008b). The **SOCR** Analyses tool has been developed for a comprehensive integration of various popular analyses to serve the education and statistical computing community. As **SOCR** is a constantly evolving project, we will be implementing more data models in the future. However, **SOCR** Analyses can already offer much more

NIH-PA Author Manuscript

NIH-PA Author Manuscript

NIH-PA Author Manuscript

than the basics in a standard undergraduate course. To enhance **SOCR** Analyses, we will be adding additional built-in examples, new GUIs and regression diagnostic visualization components. Researchers and students around the world may use for research or educational needs the Analysis GUI or download and run offline these libraries. We have been continuously receiving aspiring comments, questions and suggestions from the wide community of over 230,000 **SOCR** resource users. Our development direction and purpose are dependent on input from researchers, educators and students.

In addition to **SOCR** Analyses, we are extending the project to provide a **SOCR**-R Analyses interface, which is a server-based tool that integrates the free statistical package R into **SOCR**. While **SOCR** Analyses have all the computations running at the client (browser) side, **SOCR**-R Analyses will employ the R software at the server side, allowing the code at the client side to handle only the GUI and delegating the statistical computation to the **SOCR**-R server. The purpose is to provide a graphics based tools for students without code writing experience, with more statistical analysis tools. We plan to build the **SOCR**-Java-R-Interface in-house using XML communication protocol. Although it might be a portability sacrifice, the benefit of dispatching all the statistical computation to R is to free our application from rewriting code of the same functionality, saving development time and improving the resource quality.

Acknowledgments

These new **SOCR** Analyses were designed and implemented with support from an NSF grants DUE 0716055 and 0442992, under the CCLI mechanism, and from NIH Roadmap for Medical Research, NCBC Grant U54 Grant U54 RR021813. We are also thankful to the numerous users, students, collaborators and instructors who have provided feedback, critiques and ideas about **SOCR** Analyses in the past several years.

References

Banfield, J. Rweb: Statistical Analysis on the Web. Montana State University; 2004. URL http://bayes.math.montana.edu/Rweb/

Christou, N.; Sanchez, J.; Dinov, ID. Design and Evaluation of SOCR Tools for Simulation in Undergraduate Probability and Statistics Courses. Proceedings of the 56th Session of the International Statistical Institute Meeting; August 21–29, 2007; International Statistical Institute; 2007. URL http://www.stat.auckland.ac.nz/~iase/publications/isi56/IPM40_Christou.pdf

Conover WJ, Johnson ME, Johnson MM. A Comparative Study of Tests for Homogeneity of Variances, with Applications to the Outer Continental Shelf Bidding Data. Technometrics. 1981; 23:351–361.

Dinov, ID. SOCR: Statistics Online Computational Resource; Journal of Statistical Software. 2006. p. 1-16.URL http://www.jstatsoft.org/v16/i11/

Dinov ID. Integrated Multidisciplinary and Technology-Enhanced Science Education: The Next Frontier. JOLT. 2008; 4(1):84–93. [PubMed: 21552453]

Dinov ID, Megan MS, Thompson PM, Woods RP, Sumners DWL, Sowell EL, Toga AW. Quantitative Comparison and Analysis of Image Registration Using Frequency-Adaptive Wavelet Shrinkage. IEEE Transactions Information Technology in Biomedicine. 2002; 6(1):73–85.

Dinov, ID.; Sanchez, J.; Christou, N. Central Limit Theorem: New SOCR Applet and Demonstration Activity. Journal of Statistics Education. 2008a. URL http://www.amstat.org/publications/jse/v16n2/dinov.html

Dinov ID, Sanchez J, Christou N. Pedagogical Utilization and Assessment of the Statistic Online Computational Resource in Introductory Probability and Statistics Courses. Journal of Computers & Education. 2008b; 50:284–300.

Dinov ID, Sumners DWL. Applications of Frequency Dependent Wavelet Shrinkage to Analyzing Quality of Image Registration. SIAM Journal on Applied Mathematics. 2001; 62(2):367–384.

NIH-PA Author Manuscript

NIH-PA Author Manuscript

NIH-PA Author Manuscript

Gilbert, D. The JFreeChart Class Library. Object Refinery Limited; 2009. URL http://www.jfree.org/jfreechart/

Jennrich, RI. An Introduction to Computational Statistics. Prentice-Hall; New Jersey: 1995.

Minitab Inc. Minitab 15: Statistical Software for Process Control, Improvement and Education. Minitab Inc; State College, PA: 2008. URL http://www.minitab.com/

R Development Core Team. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing; Vienna, Austria: 2008. URL http://www.R-project.org/

Rex DE, Ma JQ, Toga AW. The LONI Pipeline Processing Environment. NeuroImage. 2003; 19(3): 1033–1048. [PubMed: 12880830]

Short, T. Rpad: An Interactive, Web-Based Analysis Program. Electric Power Research Institute; 2006. URL http://www.rpad.org/Rpad/

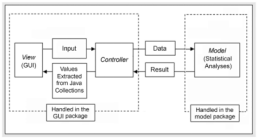

Figure 1.
SOCR Analyses MVP design.

NIH-PA Author Manuscript
NIH-PA Author Manuscript
NIH-PA Author Manuscript

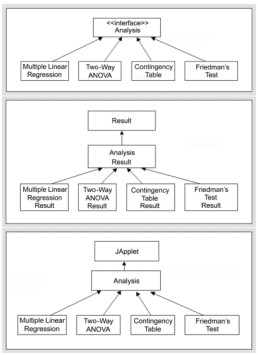

Figure 2.
SOCR Analyses usage of polymorphism. Top: inheritance of the model package. Middle: inheritance of the results. Bottom: inheritance of the GUI package.

NIH-PA Author Manuscript
NIH-PA Author Manuscript
NIH-PA Author Manuscript

NIH-PA Author Manuscript

NIH-PA Author Manuscript

NIH-PA Author Manuscript

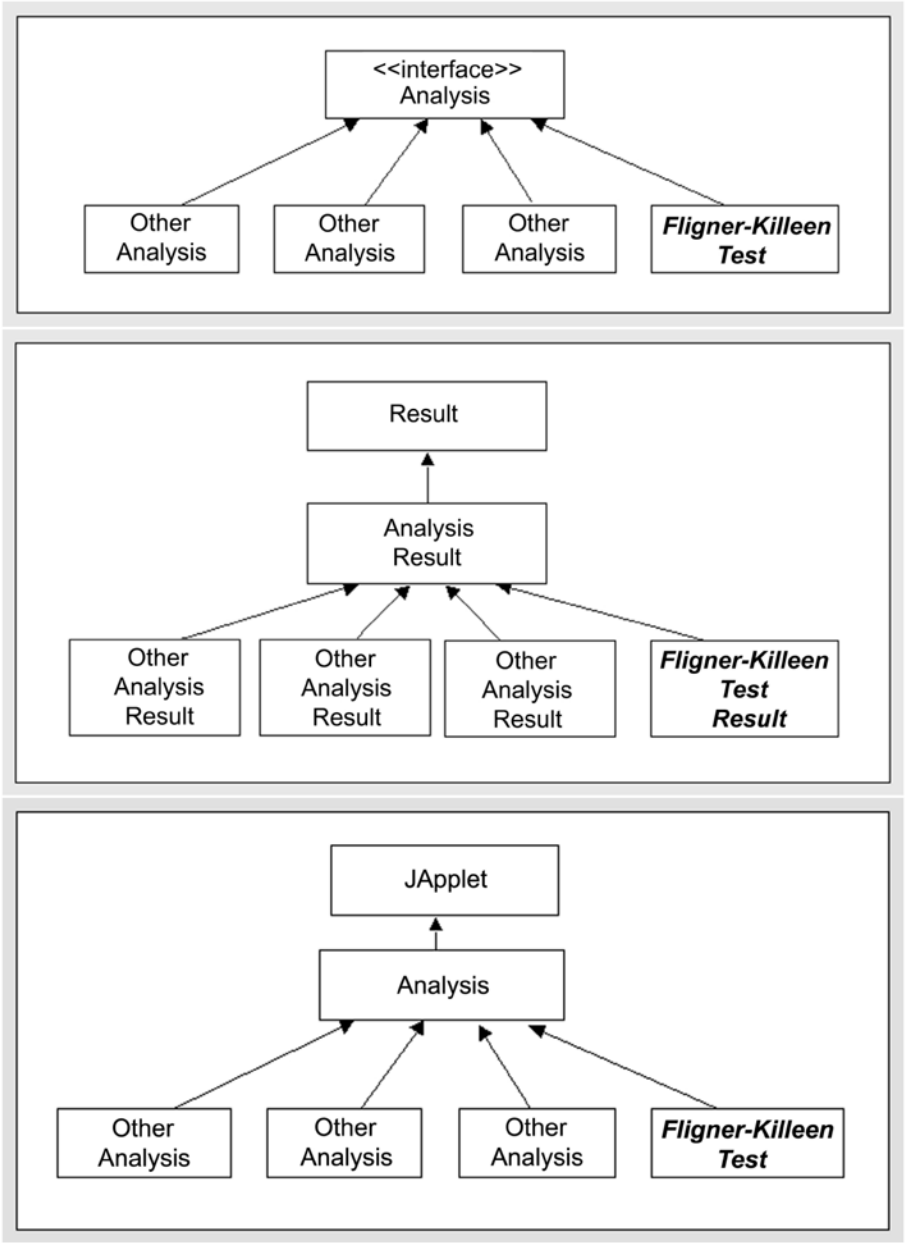

Figure 3.
Adding a new analysis. Top: adding a new class to the model package. Middle: adding a new class to the result package. Bottom: adding a new class to the GUI package.

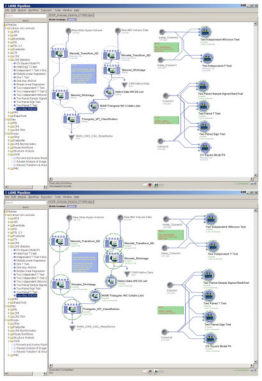

Figure 4.
SOCR Analyses as external libraries. Top: **SOCR** analyses as external libraries. Bottom: **SOCR** analyses as external libraries.

NIH-PA Author Manuscript
NIH-PA Author Manuscript
NIH-PA Author Manuscript

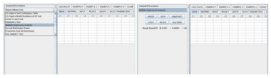

Figure 5.
Selecting an Analysis Model. Left: Pull-down combo box. Right: Options of digit display.

NIH-PA Author Manuscript
NIH-PA Author Manuscript
NIH-PA Author Manuscript

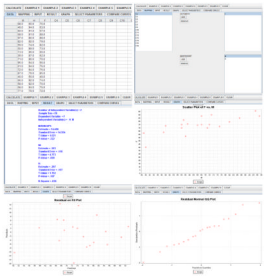

Figure 6.
Multiple linear regression example. Top left: data panel. Top right: mapping the variables. Middle left: result panel. middle right: scatter plot. Bottom left: residual on fit plot. Bottom right: normal QQ plot.

NIH-PA Author Manuscript
NIH-PA Author Manuscript
NIH-PA Author Manuscript

Figure 7.
Contingency table example. Left: entering data. Right: result panel.

NIH-PA Author Manuscript
NIH-PA Author Manuscript
NIH-PA Author Manuscript

NIH-PA Author Manuscript

NIH-PA Author Manuscript

NIH-PA Author Manuscript

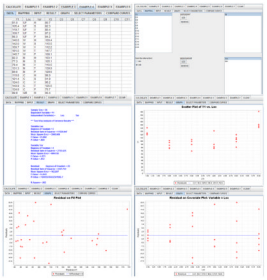

Figure 8.
Two-way ANOVA example. Top left: data panel. Top right: mapping the variables. Middle left: result panel. Middle right: scatter plot. Bottom left: residual on fit plot. Bottom right: residual on covariate plot.

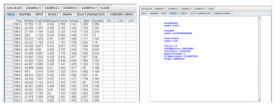

Figure 9.
Fligner-Killeen example. Left: data panel. Right: result panel.

NIH-PA Author Manuscript
NIH-PA Author Manuscript
NIH-PA Author Manuscript

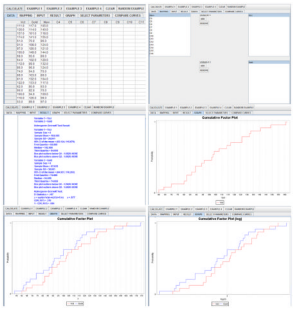

Figure 10.
Kolmogorov-Smirnoff example. Top left: data panel. Top right: mapping the variables. Middle left: result panel. Middle right: cumulative factor plot of one variable. Bottom left: cumulative factor plot of two variables. Bottom right: cumulative factor plot on log scale.

NIH-PA Author Manuscript
NIH-PA Author Manuscript
NIH-PA Author Manuscript

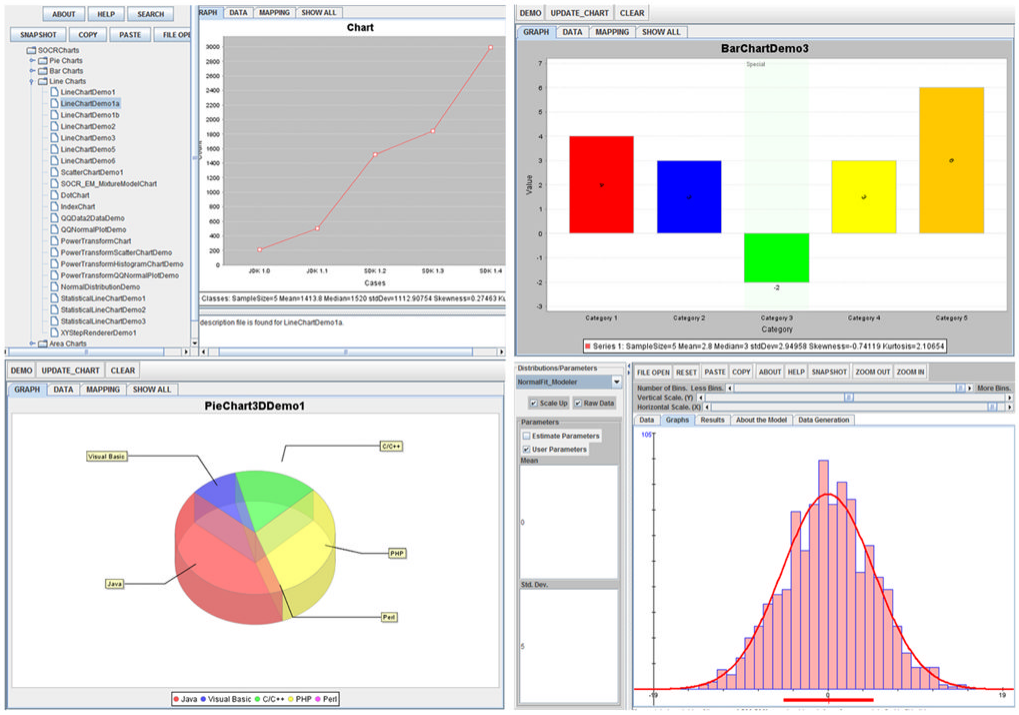

Figure 11.
Displaying data. Top left: selecting a chart to display. Top right: bar chart demonstration. Bottom left: pie chart demonstration. Bottom right: histogram and normal fit.

NIH-PA Author Manuscript
NIH-PA Author Manuscript
NIH-PA Author Manuscript

NIH-PA Author Manuscript

NIH-PA Author Manuscript

NIH-PA Author Manuscript

Table 1

Implementation of **SOCR** Analyses.

Models	**Sub-categories and implementation**		
Linear models	One-way ANOVA, Two-way ANOVA, Simple linear regression, multiple linear regression		
Sample comparison		Parametric	Nonparametric
	Independent samples	*t* test (pooled and unpooled)	Wilcoxon rank sum test, Kruskal-Wallis test
	Paired samples	*t* test	Friedman's test, Sign test, Signed rank test (Wilcoxon) Kolmogorov-Smirnoff test
	One sample	*t* test	
Categorical data analysis	Chi-square test for independence and homogeneity, Chi-square test for goodness of fit, Fisher's exact test, Proportion test		
Power analysis	Normal distribution		
Other analyses	Survival analysis, Fligner-Killeen homogeneity analysis		

NIH-PA Author Manuscript

NIH-PA Author Manuscript

NIH-PA Author Manuscript

Table 2

Comparison of **SOCR** Analyses with **Rweb**, **Rpad**, and Minitab.

Programs	SOCR	Rweb	Rpad	Minitab
Open source	Yes	Yes	Yes	No
Cost	Free	Free	Free	USD1195
GUI-based	Yes	No	Yes	Yes
Audience	K-college	College to Postgraduate	College to Postgraduate	Business and Industry
URL/file data	Yes	Yes	Yes	Yes
OS/platform	All	All	All	Windows only
Environment	Client side	Server	Local or server	Local
Network dependency	First time only	Each call	Each call	None
Coding exprience	No	Some	Some	No
Learning curve	Simple	Know R syntax	Know R syntax	Simple