Project Number: 51-JP-0202

# STATISTICAL TEACHING AIDS

An Interactive Qualifying Project Report

submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

in partial fulfillment of the requirements for the

Degree of Bachelor of Science

by

[illegible]

**Brian P. Gottreu**

**Jeremy A. Slater**

Date: April 24, 2003

Approved:

[illegible]

**Professor Joseph D. Petruccelli, Major Advisor**

**1. statistics**
**2. teaching**
**3. Java**

Our project group designed, constructed and implemented two new web-based labs for introductory statistics courses. We incorporated the latest research on computer-based pedagogy into their design. The Java applets we created dynamically link graphs and data in an interactive, real-time environment. Student feedback obtained by using the labs in MA 2611 in A term, 2002, validated the efficacy of the labs and resulted in modifications and improvements.

# Contents

## List of Figures

# List of Tables

## 1 Introduction

The introductory statistics courses at WPI, MA2611 and MA2612, have a set of labs associated with them. Most of the labs involve the computer to analyze data or demonstrate statistical concepts. The original computer-based labs utilized SAS statistical software.

SAS is a large scale software system for statistical analysis. It has many features that allow it to perform almost any computation that one would be inclined to perform. It was not originally designed to be used in an educational environment, though it has successfully been used for such purposes. For statistics labs at WPI students use only a few portions of the larger SAS system, namely SAS/INSIGHT and SAS/EIS. This makes their interaction with SAS simpler, but students still find SAS difficult to learn and use. There are setup costs involved with using SAS, which are tolerable, but there are usually problems even after SAS has been correctly initialized. The chief among these problems are human computer interaction errors. These are to be expected with users new to a piece of software, but should be minimized, especially when the goal is not software competency, but learning other concepts. In this regard, a more specialized piece of software should allow for fewer errors during operation and easier acquisition of statistical concepts. Another problem with SAS is that it is only available on campus. Students would benefit if they could run the labs from off-campus locations.

The goal of this project was to create two new, educationally sound statistics labs that would eliminate the difficulties of SAS-based labs. To achieve this result we used advances in computational technology since the creation of the original SAS labs. "The web has the potential to transform traditional models of teaching."[1] A key force behind such potential is the use of Java. We used Java extensively in the creation of applets for the new labs. The applets allow for real-time interactive graphics and greater accessibility of the labs. Through this real-time interactivity, we created labs that allow for easier and more effective learning of the statistics concepts and removed a number of the difficulties with using SAS as lab software.

Java is a fairly new programming language produced by Sun Microsystems. It was designed to be platform-independent and to allow for simpler user interface implementation. To achieve this, Java programs are usually run on a virtual machine(VM). The advantage of running software on a VM is that once a piece of software is written, it can be run on any platform that supports a VM. Practically, this means that any software written in Java can run on most operating systems, including Windows, MacOS, and most varieties of Unix.

Earlier projects at WPI adapted existing SAS-based labs for the web using Java and other web-based solutions.[2][3][4] For our project, we designed and implemented new labs for the material in chapters 1 and 2 of *Applied Statistics for Engineers and Scientists.*[5] Lab 2.2 introduces students to data transformations and their effect on data distributions. Lab 2.3, based on the famous Deming funnel experiment,[6] explores process tampering and its effects.

[1]David G. Brown, ed., *Interactive Learning*, (2000), Anker Pub. Co., p. 56.

[2]Liesen, Eric and Whitfield, Paul, *Statistical Teaching Aids*, (2001)

[3]Kawato, Takeshi, *Statistical Teaching Aids*, (2003)

[4]Clein, Robert and Holmes, Samuel, *Online Statistics Labs*, (2003)

[5]J. D. Petruccelli, B. Nandram, and M. Chen, *Applied Statistics for Engineers and Scientists*, (1999), Prentice Hall.

[6]W. Edwards Deming, *Out of the Crisis*, (1986), Massachusetts Institute of Technology.

# 2 Literature Review

## 2.1 Previous Work

At the time that we began our project there were other project groups that either had done or were in the process of developing web-based statistics labs. These projects all entailed converting existing labs to a web environment or developing new labs for introductory statistics courses. Not all of these groups were developing Java applets, but for all of the projects the main goal was to design a web-based lab that can be accessed over a broad range of platforms. These projects were all of particular interest to us because we planned to design similar products and we essentially had the same goals. By evaluating these existing projects we were able to get an idea of the desired outcome and gain a better understanding of our task. We were also able to critique these designs to incorporate some of the innovative ideas while improving on their weaker features.

At the time of our implementation there were three available web-based labs at WPI[7], with others in development. These three web-based labs were all made by the same group and so their design characteristics are relatively similar. However, these labs were still helpful in providing us with ideas for our implementation. Following the link to any of these labs brings up an "Introduction" window which introduces the lab and supplies links to an applet, a glossary, printing and saving instructions, and a set of questions connected with the lab. The applet window includes the computerized part of the lab along with instructions for conducting the lab. Placing the lab

[7]located at http://www.math.wpi.edu/Course_Materials/SAS/lablets/statlab.html

instructions in the applet window so that they can be viewed as the lab is conducted represents a good pedagogical layout and is one key advantage of the web-based labs over the SAS labs. Overall these previous labs are well developed and proved useful to us as a guideline.

Another helpful source from WPI are web labs 4.1, 4.2 and 4.3[8]. These labs, which were still under development during most of our project, implement some of the features that we considered useful in professional statistical Java applets. Since the author of these labs provided extensive source code and customized packages, we were able to build upon material that already existed. This allowed for more advanced development of our own web-based labs.

Outside WPI there are a large number of applets pertaining to statistical analysis and education. Many Java applets are hosted on web sites as either labs coordinated with a class or as open-ended statistical tools available for free or for purchase. One site of particular interest is the Statlets site located at http://www.statlets.com. This web site specializes in Java-based statistics applets encompassing a broad range of functionality. This site includes many examples of functions that we intended to develop. Some key features include data entry fields, tabbed windows for switching between data entry and other views, and pop-up tip windows. While the applets on this web site are very well built, they are more complex than what we intended to build. These applets are good tools, but due to their lack of instruction are bad examples of an educational lab. This site is beneficial as a good example of interface design.

[8]These labs provided by Takeshi Kawato

## 2.2 Techniques

The key to good software is proper design. We implemented an interaction design methodology to construct our web-based labs. Interaction design differs from interface design by specifying more than where windows and icons are located. This design method describes how the user interacts with the software, how the user will use the mouse and keyboard to affect and control the software, and how the software will lead the user through its use. Interface design is stationary, but interaction design is temporal; it describes the user experience through time.

Incremental development is a widely-approved method of design implementation. However, when using this method it is tempting to also use incremental design, instead of the preferred method of interaction design. Incremental design has the user design the software in pieces which leads to problems when sections of code fail to interact correctly in the future. Interaction design, however, requires that the software be completely designed before development with user interaction in mind. This means that there will be fewer problems in the development stage.

Incremental development requires every piece of software to be designed beforehand. By designing this way, the interaction specification is followed during each step of design and development. This avoids the situation where a fundamental portion of code is created incorrectly and features are then added which are dependent on that code. When the original code is fixed to make it adhere to the interaction specification, newer features can be made inoperable. Adherence to the methodologies of interaction design ensures that the program code does not deviate from specification, and thus additional code never becomes invalidated.

A large problem in software development is having a specification that continues to change during implementation. Changes in the design that invalidate large portions of the code base should not occur. This only delays the project and dampens the spirits of the coders, thus making future development slower. For our project, development time was at a premium, so changing the design and throwing away work was not an option.

Generally, when developing a computer-based educational tool, there are many user psychological attributes to consider in designing for effectiveness. A well thought out and properly implemented cognitive design will provide the most beneficial experience for a user. Some considerations in design include the users' knowledge of computers and knowledge of the material being taught. An understanding of browser mentality[9] is necessary so that the developer can create a captivating experience, while still maintaining the educational goals. Aesthetics such as text characteristics, window sizing and screen clutter all need to be considered to create a successful learning environment.

In any multi-user educational tool certain assumptions need to be made about the students' knowledge of computer concepts and task concepts[10]. A lab should only include material covered up to the date of the lab. Ideally, the student will have already gone over the material in class, and will at the very least know the terminology. The lab design must be kept intuitive and provide adequate instructions on its use. The difference between the actions the user assumes are necessary to complete the project and the actual actions

[9]Instructional and Cognitive Impacts of Web-Based Education, p.49

[10]Instructional and Cognitive Impacts of Web-Based Education, p.43

available is referred to as the gulf of execution[11]. To minimize the size of this 'gulf' a student needs the tools necessary to complete the assignment, such as a tutorial or glossary. If any student begins the lab and is uncertain of a lab instruction, or how the instructions relate to the Java applet, then that student will be at risk of not learning the material. With a simple tutorial a lab can contain some advanced features without the risk of losing the students' interest.

One danger of a web-based educational tool lies in a user's browser mentality. By "browser mentality" we mean a web user's inherent desire to merely skim web pages rather than fully read them. The risk then, is that with this attitude a user may attempt to race through a lab without fully comprehending the steps involved. Browser mentality is one of the more serious cognitive considerations that must be taken into account during the design process. The software must require the user to progress at a pace at which the lesson will be useful. One solution is requiring interactivity between the Java applet and the lab instructions. For example; a certain section of the instructions can be designed to pause and wait until the user has successfully performed a task. Another solution is requiring the student to do some of the less redundant work by hand, such as typing in initial conditions or results. Similarly, the student could be required to perform an operation by hand that would then be repeated by the computer over many iterations. This way the student learns what is going on, but also has the power of the computer for efficiency.

Psychology and behavior also play a key role in the graphical layout of the applet. There are many factors that can aid or harm the students'

[11]Human Computer Interaction, p.106

productivity. While one main focus is on developing a web-based tool, the underlying lesson is taught with a lab description including step by step instructions. These instructions need to be located in an appropriate location on the screen, they need to be written coherently, and they should be displayed in a legible font. This last point is actually an issue with one of the previously completed labs due to a bad font used on the lab machines. Overall, the text should be relatively low in density, so that the statements are short and easily read. At most, text should be split into screen-sized chunks such that the user does not need to scroll within a window. This is implemented in our lab description through the use of short sections of text tied to each other with forward and backward buttons. Bold and logical headings are helpful and well written instructions are necessary.

## 2.3 Project Alternatives

The Java language is not necessarily the only or the best option for producing a web-based lab. Other popular web languages exist, such as CGI, Javascript and PHP. These languages differ not only in their implementation, but also in their execution. Both CGI and PHP run on a server while Java and Javascript run on the client computer. All three of these alternatives are far less capable than Java, but because they are simpler, they are easier to implement and they maintain a better level of cross platform compatibility. A drawback of the server-side languages PHP and CGI is that real time graphical manipulations are infeasible due to client-server latency. A client side language is superior with respect to response time, especially for the quick data manipulation that our statistical labs require.

## 2.4 Current Issues

The current statistics labs are implemented using SAS. SAS is a powerful general purpose statistics tool designed for professional data analysis, but for statistics labs, such a powerful and general tool is not needed. SAS has a large number of features, but for the labs only a small subset of the features is used. The large number of unused features still remain, taking up space, and only hinders the use of the other features. The more windows and menus that are available, the slower the progress is through a program. SAS is a perfect example of this with its five default windows plus more for each macro. Alan Cooper[12] states that one of the key rules of software design is to not make the user feel stupid. From personal experience, SAS can and will do this.

SAS does have some strengths however. Since SAS is a general tool, students are free to try new things, perhaps gaining new insights. SAS can be adapted to perform many operations on any user-defined data. When an educator creates a new lab, it is possible that only the lab instructions need to be created. The features of SAS may be sufficient to implement the lab; however, new macros may need to be created if the lab is to be accessible to the average student.

When the labs are moved to a web-based solution, some of the features associated with SAS are lost. The flexibility and power of SAS is no longer at the disposal of the students or educators, but since most of the power of SAS is never used in a lab setting, this is not a major loss. An educator who wishes to change a web-based lab or create a new one, is most likely unable to do it alone. Programmers will need to be brought in to create or alter

[12]The Inmates Are Running the Asylum, p. 25

the Java applet. The cost of creating web-based labs is higher than that of SAS-based labs.

However, the strengths of web-based labs significantly outweigh the weaknesses. Since it is known exactly what is needed for the lab as the software is coded, only the features necessary for the lab are included. This lowers cognitive friction and creates a better environment in which the students can learn, which is one of the main goals of this conversion.

One of the best features that can be implemented with Java, but not with SAS, is real-time interactive graphics. For example, in a SAS-based lab dealing with data summaries, the student is asked to change the data value in a spreadsheet. The student watches as a set of graphical and numerical summaries change in response. This lab was adapted to the web by a previous project team. The web-based lab does the SAS version one better in that the user can change any data point in a graph by clicking and dragging with the mouse while the summary statistics and graphs update in real-time. Java brings interactive graphics to another level, which we feel allows students to more quickly get a feel for the concepts involved.

Another benefit of using web-based labs is that students can access them from any computer having internet access. With the SAS labs, students must use the CCC machines to complete the lab.

For students, the switch to web-based statistics labs will make it easier to perform the lab and understand the statistics behind the labs. Educators will need to use more resources to create new labs. However, this cost will lessen over time. Eventually most of the labs will be web-based, and the need for more labs will not be great enough to warrant the creation of new labs, until a new technology comes along that has a greater potential for making computer-based education more effective.

# 3 Procedure

## 3.1 Project Methods

### 3.1.1 Planning

Early in the project, we worked with Professor Petruccelli to develop outlines of the two new labs. From these we created storyboards detailing the layout of the labs and the way students would interact with them. When we designed the layout and interface of the labs, we used the recently-learned principles from our background research. Professor Petruccelli reviewed the storyboards and suggested changes. After incorporating these changes, we moved on to the design and implementation phases of this project.

### 3.1.2 Development Cycle

Before discussing the specifics of our development effort, we will describe the development cycle we used. There were three main phases of the development cycle: Design, Implement, and Test.

***Design***

Since the labs had already been laid out and all the main features specified, we knew what features or components we needed to create and we knew what they should do. From these criteria, we designed the internals of each component in turn so that it would perform its function.

***Implement***

Using the design we created, we implemented the component using the language we had decided upon, Java. Sometimes Java did not allow things to be implemented easily, if at all. When this occurred (and it did so more frequently than we would have liked), we returned to the design phase to rework the design so that it could be implemented in Java. When the component was fully constructed, we placed it in the larger system and moved on to testing.

***Test***

To test the component we placed it in both the system and a special framework specifically created for testing the component. With the testing framework, we could easily see if the component met the specification. This was because the component in question was the only unknown. All the other variables, the other parts of the system, had been removed.

Even if the component worked flawlessly in this initial testing, it needed to be tested in the end system. Only in the actual system could all the subtleties of its interaction with the rest of the system be tested.

When a problem occurred, the design and/or implementation phase was revisited, depending on the origin of the fault. With many design problems, one could avoid fixing the faulty design and just add more code to work around the problem and hopefully make things function properly. We steered away from that technique and tried to do things the right way, which is to redesign the component and then implement the updated design.

### 3.1.3 Our Design

From the storyboards, we designed the overall framework of the labs and the most basic features. We started with Lab 2.2. Only after serious progress had been made did we start parallel development of Lab 2.3. The key advantage of this was that everything we learned from Lab 2.2 could be applied to Lab 2.3.

### 3.1.4 Our Implementation

Considerable portions of code from Lab 2.2 were reused for Lab 2.3, namely code for the histogram, box and whiskers plot, and slider bar. We reused code for the instructions at the bottom of the labs, which came from a previous project group. We also tried to reuse code from another project group that implemented the histogram and box and whiskers plot, but were unable to achieve proper functionality with it.

When the labs began to take shape, we started testing them and Professor Petruccelli reviewed them regularly. As the labs materialized, we encountered some issues that we had been unable to imagine when they had existed only on paper. The storyboards were updated as these problems arose.

Two of the initial requirements for the labs were that students should be able to print and save the various graphs and tables in the labs. With Java, printing and saving are both possible, but due to difficulties only saving was implemented in the final version of the labs. Our main difficulty was having the labs print correctly across all platform, browser, and printer combinations. While we were working on a solution to this, we learned that the newer versions of Internet Explorer had an incompatible interface for printing and would not work with our labs. Since printing would have worked for fewer users as time went on, Professor Petruccelli decided that printing was not worth the additional effort and removed that feature from the list of requirements. Students can still achieve the same end result by first saving the graph or table then printing it from a new browser window or any other image viewing program.

### 3.1.5 Our Testing

Throughout this process we continually tested each component and feature thoroughly. Since one of the main aims of this project was to create a cross- platform teaching tool, we tested the labs on several combinations of hardware, operating systems, and web browsers. Professor Petruccelli reviewed and tested our work regularly, twice a week if not more often. He provided much needed feedback, giving us suggestions for improvements and new features.

### 3.1.6 Alternatives

We could have done the programming of the labs in another language instead of Java. The main contenders were JavaScript, Perl, and PHP. Both PHP and Perl suffer from the same problems and have the same benefits. It would have been possible that they would simplify printing and saving from the labs, but interactive graphics would have been enormously difficult to implement, if not impossible. The problem of compatibility between machines would have been much less of an issue, however, since all the processing for the labs would have taken place on the server, not on the users' machines. The downside to this is that during a lab section when 20 students would try to run the labs, the load on the server could have caused a noticeable performance hit.

JavaScript would have allowed the interactive graphics, but printing and saving from the labs would still have been an issue, even more so than with

Java. JavaScript implementations are not standardized, so large portions of code might have required rewriting to support two browsers. From a compatibility and feasibility standpoint, JavaScript is like Java but worse.

## 3.2 Research Methods

In order to determine the effectiveness of our labs we gauged the reactions of student users. Besides testing the basic compatibility of our software, we also tested the ability of our labs to teach the lessons. We had the students in the A term introductory statistics course, MA 2611, test our labs during one of their weekly lab sections. The software performance was essentially bug and incident-free. A survey was administered after each lab, asking students various questions about the lab and their experience with it. The survey inquired about both pedagogical and technical issues. Our observations and student responses to a questionnaire suggested features that could be added or improved, and the labs were subsequently modified in response.

# 4 Lab Descriptions

## 4.1 Lab 2.2

Data having a unimodal symmetric distribution is easier to analyze than non-symmetric data. Because of this, data analysts often transform non-symmetric data to obtain a more symmetric distribution. Lab 2.2 introduces the students to this concept of transforming data with a unimodal distribution in order to make its distribution more nearly symmetric. The lab looks at the class of power transformations: transformations that raise data values to a power (or, for the power 0, take the logarithm), and explores their effect on data with non-symmetric unimodal distributions.

The main website provides a basic introduction that informs the student of the lab objectives, and examples of real life applications. A frame on the left side of the lab window contains a menu with links to the lab introduction, a description of power transformations, the applet, a glossary of terms, instructions on saving and printing graphs, a set of questions to answer for the lab report, and instructions for installing a Netscape certificate needed for saving graphs when using the Netscape browser.

The menu is constantly visible so that students can access these resources conveniently while doing the lab. Thus, students can refresh their memories about what a power transformation is, or look up the definition of the median at exactly the moment they need to while doing the lab. The ability to access information as it is needed enhances the instructional power of the lab.

The website layout is designed so that each page only contains information vital to the immediate topic. This way, there are clearly-defined menu items on the left of the window and a clean content frame on the right. By clearly defining sections and keeping the content short and neat, we adhere to proper pedagogical and psychological norms. The student can navigate the web portion of the lab with greater ease and maintain interest in the material.

The applet window contains an information panel with an instruction panel below it (Figure 4.1). The instruction panel displays only a few lines at a time, but can be scrolled to display further text. The initial text in the instruction panel provides a more detailed introduction to the lab and applet and gives explanations of the lab controls. The instruction panel then provides step-by-step instructions for the student.

At the top of the applet window is a menu bar with two menus: File and Preferences. From the File menu, the student can select Save to save the graphs or select Exit to close the applet window. The Preferences menu contains three items: Customize Colors, Font Options, and Scaling Options. The first two are for aesthetic purposes only and allow the student to change the colors and font used in the graphs. There are three scaling options that can be selected: Auto Scaling, Static Scaling, and Zero Left. Auto Scaling sets the range of the histogram to be the current extremes of the transformed data set. This provides the most 'zoomed in' view of the histogram. The Static Scaling option sets the range of the histogram to be the maximum and minimum values that the data can obtain through the entire transformation range. The Zero Left option is similar to Static Scaling except that the histogram's minimum range value is set to 0 regardless of the value of the data set's minimum.

Performing this lab consists of navigating the instruction panel, sliding a power transformation bar and examining the information panel. This instruction panel provides the students with a concise introduction to the material, and short 'to the point' instructions. The slider bar shown in Figure 4.1 provides a quick method for selecting the power of the power transformation used to transform the data. As the data are transformed the applet updates the histogram, box-plot and a summary statistics table. The histogram window provides a range along the x-axis which is helpful in determining how the power transformation varies the data maximums and minimums. The box-plot assists in viewing the symmetry through the quartile ranges from the median. The summary statistics box contains the mean, standard deviation, first and third quartiles, median and the interquartile range (IQR): the range of the middle 50% of the data. These summary statistics complement the graphs nicely: as they make transformations, the students can see how the graphs and summary statistics vary together. So at the same time the lesson on transformations is learned, student understanding of graphical and numerical summaries is enhanced. Using the graphical information and summary statistics, the student can analyze the data to generate symmetric histograms.

The data are generated by clicking on the 'New Data' button. This process generates a set of pseudo-random data from a normal distribution. The normal distribution has a mean of 5 and a standard deviation of 1. In the unlikely case that a negative data value is generated, it is discarded and a new value is generated. Each data value is then raised to a randomly selected power, q. To select q, a number is first randomly selected from the range of values that the transformation slider can select, which is every one-tenth

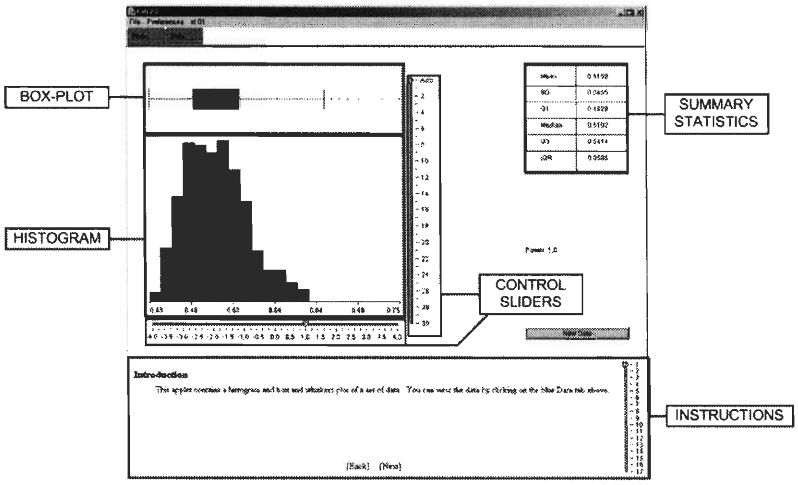


Figure 1: Lab 2.2 Applet

from -4 to 4. This number is then inverted to determine q. In the case where 0 is first selected, the data values are not raised to a power, but are exponentiated.The student sees the transformed data as a skewed unimodal histogram displayed in the applet window along with its accompanying box plot and summary statistics. The goal is for the student to transform the data using the transformation slider to restore the histogram to a unimodal symmetric appearance. Moving the slider is an easy way to quickly get close to the appropriate power. When the transformation is visually symmetric the student can use the provided summary statistics to evaluate the symmetry by comparing the quartile symmetry from the median. At this point the student saves the image and creates a new set of data that is skewed differently from the first set. The student then finds the best transformation for this second data set. The students are required to write a lab report detailing how the objectives were attained and how power transformations work. They may be asked by their instructor to answer the questions linked to the main lab menu.

Intuitive functionality and pedagogical soundness were important considerations in the design of our labs. The student should be able to determine the function of controls by merely looking at the layout, or through very simple instruction. The instructions in lab 2.2 are clearly listed at the bottom of the screen, with intuitive forward and backward buttons. The student knows that the next button will bring up sequentially-listed instructions pertaining to the current lab, and likewise, the back button will bring the user to the previous instructions. Not only is the simple navigation through the instructions essential, but also the format of the instructions themselves is integral to student learning. Each instruction is at most a few sentences and maintains a clear point. Brevity and clarity of instruction helps maintain student interest.

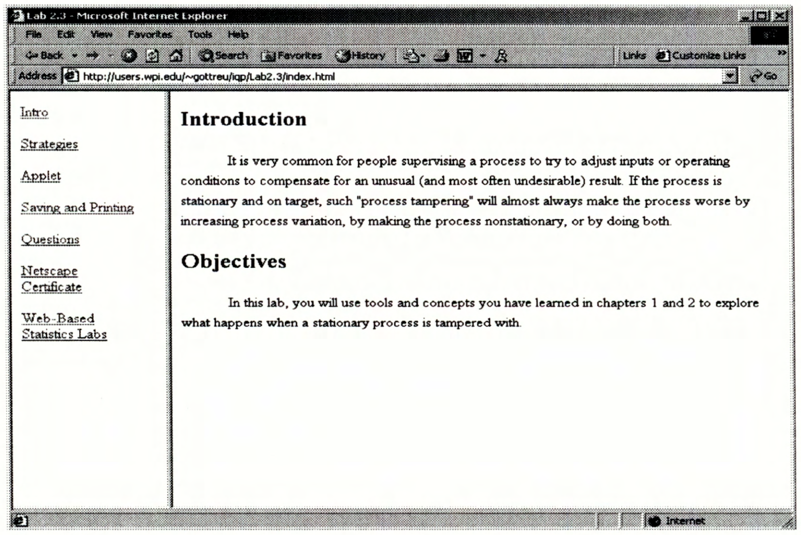
Another key design feature is the instantaneous feedback in the histogram and box plot when the power transformation slider is moved. The student not only immediately recognizes how the slider operates, but also quickly sees how transformations affect data distributions. The real-time display shows not only the histogram and box-plots, but also range information. If the original data set is between 0 and 1, then the range displayed under the histogram shrinks as the values are raised to powers greater than 1. When this type of data set is raised to powers between 1 and 0, then the range increases. When the original data set contains values greater than 1, then the range changes in an opposite manner. Powers greater than 1 increase the range, and powers between 0 and 1 decrease the range. With the small incremental levels of power, the data is only modified slightly for each tick of the slider, and this brings a very educational visual dimension to power transformations.

By default, the number of bars displayed in the histogram is automatically determined. This does not always make the histogram viewable through the entire range of transformations. To overcome this, the student can use the vertical slider bar to select the exact number of bars to display in the histogram. The fewer bars that are displayed, the wider and more visible they are.

The summary statistics are also updated in real time as the slider bar is modified. The summary statistics also help inform the student about the symmetry of the data distribution: the values of the quartiles and comparison of the mean and median are particularly helpful. Only the values essential to the histogram and box plot are displayed, and so the student is not overburdened with inconsequential information. Overall, lab 2.2 has clear instructions and a natural user interface.

## 4.2 Lab 2.3

The main ideas behind Lab 2.3 are process variation, process stationarity, and how they are affected by process tampering. The calibration of a cannon is used to show how adjusting a stationary process can cause it to become nonstationary or increase the variation.


Lab 2.3 - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Links Customize Links
Address
Go
Internet

http://users.wpi.edu/~gottreu/iqp/Lab2.3/index.html

Intro

Strategies

Applet

Saving and Printing

Questions

Netscape Certificate

Web-Based Statistics Labs

**Introduction**

It is very common for people supervising a process to try to adjust inputs or operating conditions to compensate for an unusual (and most often undesirable) result. If the process is stationary and on target, such "process tampering" will almost always make the process worse by increasing process variation, by making the process nonstationary, or by doing both.

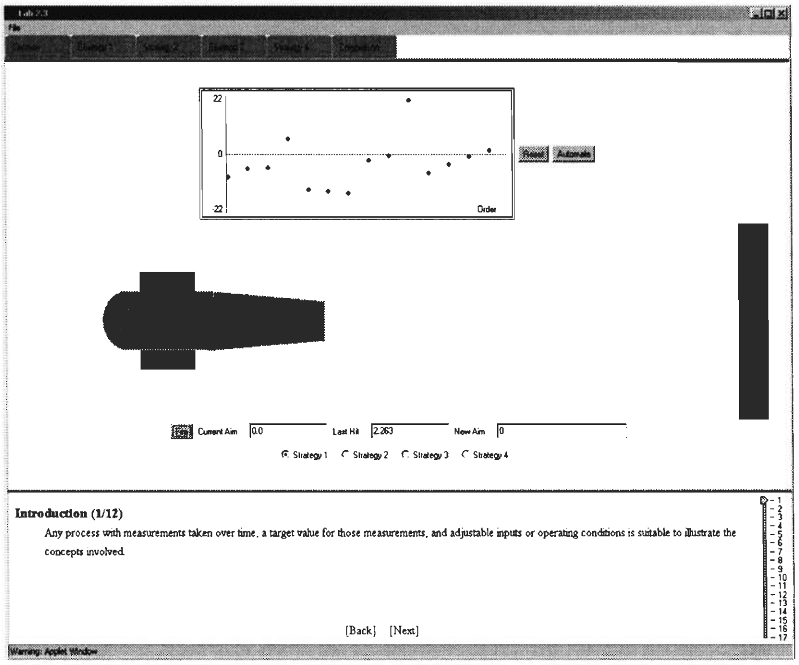
**Objectives**

In this lab, you will use tools and concepts you have learned in chapters 1 and 2 to explore what happens when a stationary process is tampered with.

Figure 2: Lab 2.3 Webpage

When a user selects a link to the Lab 2.3 web page, they are greeted with the page in 4.2. In the left frame are links to the various sections of the supporting web pages. In the right frame is the selected section. The student begins by reading the necessary background information contained in the Introduction and Objectives sections (selected by the "Intro" link). More detailed background information is contained in the Strategies section, which the student is directed to read during the course of the lab exercise. Once the lab has been completed, the student is to answer the questions contained in the Questions sections and include the answers in a lab report.

To start the applet, the student clicks the Applet link in the left frame. A new window containing only the applet will appear, leaving the supporting web pages unchanged. 4.2 shows what this window looks like. At the bottom of the applet window are the lab instructions. The instructions were placed in the applet window as opposed to the supporting web pages so that the user does not need to constantly switch between windows.



Current Aim 0.0 Last Hit 2.263 New Aim 0

[✓ Strategy 1] Strategy 2 Strategy 3 Strategy 4

**Introduction (1/12)**

Any process with measurements taken over time, a target value for those measurements, and adjustable inputs or operating conditions is suitable to illustrate the concepts involved.

[Back] [Next]

Figure 3: Lab 2.3 Applet

As can be seen in 4.2, six tabs run along the top of the applet. The first tab, labeled "Cannon", is where most of the interaction between the user and applet occurs. The graph at the top is a time series plot of the locations of all the previous shots for the selected strategy. Beneath the graph is an animated cannon that exists mainly for aesthetic reasons. Next are three fields describing the current aim of the cannon, the last location hit by the cannon, and the new location to aim the cannon. Finally, there are four radio buttons from which the user selects the strategy to use.

The next four tabs contain graphs and summary statistics for their respective strategy. In each tab are a time series plot, a box and whiskers plot, and a histogram. Also provided are the mean, median, standard deviation, and interquartile range for the data generated for the given strategy.

The last tab, "Comparison", contains one time-series plot for each strategy. This should allow the student to easily see the differences between the various strategies.

After the applet window has opened, the student begins (hopefully) by reading the instructions and performing the operations indicated. For Lab 2.3, this involves calculating the new position at which to aim the cannon for a few data points for each strategy.

Strategy 1 is to leave the cannon completely alone. The aim is left at zero for all shots. This strategy is the best strategy to use for a stationary process, such as the one in this lab. The distribution of the shots of the cannon if it is not tampered with is a standard distribution with a mean of 0 and a standard deviation of 10. Strategy 2 is to change the aim of the cannon by -y units given that y is the location of the previous shot. This strategy keeps the process stationary, but it does increase the variation in the process. Strategy 3 is to aim the cannon at location -y given that y is that previous shot's location. This strategy greatly increases the variation of the process, while also making it nonstationary. Strategy 4 is to aim the cannon at location y given that y is the location of the previous shot. This strategy also causes the process to become nonstationary while also increasing the variation. This strategy produces a random walk.

For each strategy, the student is asked to calculate the first five shots manually. The student is then instructed to press the "Automate" button to fire the cannon 495 more times. This combination of manual calculation followed by computer automation allows the lab to use the power of the computer to demonstrate the consequences of each strategy, while both ensuring that the student understands what each strategy does, and keeping the student involved in the process.

Once data have been generated for all four strategies, the student can select the "Comparison" tab and easily compare the strategies visually. The consequences of the different strategies can be easily compared since each strategy tab contains various summary statistics along with the box-and-whiskers plot, histogram, and time series plot. The summary statistics included are the mean, the median, the standard deviation, and the interquartile range.

Most of the graphs in the applet may be saved to files. For the "Strategy" panels, the three graphs and the summary statistics will be saved to a GIF file. For the "Comparison" tab, the four time series plots will be saved to a GIF file. Nothing on the "Cannon" tab can be saved. This is because the graph for the selected strategy can be saved from its appropriate "Strategy" tab and because none of the other visible elements in the "Cannon" tab would be useful to save.

The graphs cannot be printed directly from the applet, but they can still easily be printed. To print them, one has only to first save the desired graphs then open the image in a new browser window. Once loaded, the image can be printed in the same manner as any other online content. The image could also be opened in any standard image viewing or editing software if the user wishes to edit the image or customize the printing more than is allowed by the browser's printing options. The image can also be imported into word processing software for direct inclusion in a lab report.

# 5 Programmers Reference

## 5.1 Lab 2.2

The main class for Lab 2.2 initializes the visual components and settings of the applet. The initialization routine begins by sizing the applet window dynamically to the user screen size. The applet window then takes up only a percentage of the screen. Standard menus are added to implement image saving and customization options. A tabbed pane created by Takeshi Kawato is inserted in the main window along with step-by-step instructions at the bottom of the applet. The instruction panel is mostly borrowed from previous labs with the exception of the slider bar, and is in the source provided by Takeshi Kawato.[13] The tabbed pane consists of two windows; one containing the plots and summary statistics and the other with the data set. The rest of Lab22.java implements object listeners which wait for mouse commands to change values of slider bars, instruction pages and the tabbed display. As the slider bars change the power transformation value, the data are forwarded to the histogram, box-plot, summary statistics and data table functions. Selecting different tabs brings to focus the two different tabbed window panes.

The summary statistics are immediately controlled by the top class Lab22. This is a simple six by two celled table containing the mean, standard deviation, first and third quartiles, the median and the interquartile range. These values are updated whenever the data is power-transformed and displayed in the table.

The data table located under the detailed data tab is also directly con-

[13]Kawato, Takeshi, *Statistical Teaching Aids*, (2003)

structed with the lab22 class. This table displays a title for the original data and for the transformed data. There are cells for 18 values under each category which display the whole data set through the use of a control slider bar.

The data are manipulated using the power transformation scrollbar. This scrollbar is created using the Slider class which generates a scrollbar in the allotted space. This scrollbar can be oriented vertically or horizontally using the setOrientation command. Automatic or manual tick marks and labels can be added. The maximum and minimum values are set using the setMinimum and setMaximum commands. Moving the slider results in a value being stored, which can be read with getValue. By polling these values, the power transformation value and number of bars value are changed.

The instruction panel is created with the DirectionPanel.java class which is in the package we were provided with. Setting up the instruction panel only requires adding new directions with the addDirection command. This command automatically increases the size of the instruction array, so that the scrollbar object knows the size of the instructions at all times. Directions are added with a title and a description. When the instruction panel is displayed, the first instruction appears, along with forward and back buttons in each frame and a scrollbar so the user can jump to any specific frame quickly.

The tabbed panels class allows many windows to be utilized in a relatively small location. Users can select from various tabs, to display the screens they want to display. Setting up the tabbed panels requires creating new panels with the addTabbedPanel function with a content panel and tab caption passed as parameters.

The plots window incorporates a histogram that shows the transformed data. The histogram is set to display the data by using the setData function. When this occurs, the histogram calculates the most extreme values that can occur when transforming the data. This is used when the Static Scaling option is selected. The number of bars in the histogram and their values are calculated when a new data set is provided, the data set is transformed, or the number of bars is manually selected. To improve visibility, the heights of the bars are automatically scaled to be large as possible.

The plots window also includes a box-plot directly above the histogram, displayed on the same scale. This plot is provided with the same data as the histogram through the setData function. The range is also set in the same way as the histogram with the setRange option, and supports zero-left, variable and fixed range options. The data processing is handled in the graphics update routine and consists of displaying the outliers, +A and -A, first quartile, third quartile and the median. Most of the processing occurs in resizing the data so that they are applicable to the screen size that the box-plot uses.

## 5.2 Lab 2.3

From a maintainability and reusability standpoint, Lab 2.3 would be a better choice than Lab 2.2 as a starting point for new web-based labs. Since substantial portions of Lab 2.2 had been completed when the implementation of Lab 2.3 began, we were able to use the lessons we had learned when coding Lab 2.3.

When the applet begins initialization, it first sets up the window and frame in which it will sit. It then calls a number of initialization functions, each of which initializes a group of related components. There are functions to setup each tab, to create the menu at the top of the window, and to create the directions at the bottom of the applet.

When the cannon tab is selected, nothing occurs until one of the Reset, Fire, or Automate buttons is pressed. When the Reset button is pressed, the data set for the selected strategy is removed and aiming information is reset. When the Fire button is pressed, the current aim is first checked for correctness. If it is incorrect, a message is displayed to the student. If it is correct, the cannon is aimed at the location and a shot is 'fired'. This is done by taking the current aim of the cannon and adding a random value. This random value comes from a normal distribution with mean 0 and standard deviation of 10. After the shot has occurred, the last hit field is updated. When the Automate button is pressed, the same things occur as when the Fire button is pressed, except that correct data is generated even if the next aim is incorrect. The Automate button generates as many data values as are needed to have a total of 500 data points including the manually fired shots.

When a new strategy is selected from the strategy radio buttons, the aiming and last hit information are updated as well as the time series plot above the cannon.

To be more computationally efficient and to keep the applet from becoming sluggish, the graphs that are not in the Cannon tab are not updated until the tab in which they reside is selected. When the student switches to a strategy or comparison tab, a callback function is invoked. To register the callback, the tabs' constructors are passed an object, which contains a single method when the tabs are created. This object's method updates all the graphs contained in its associated tab.

# 6 Results

## 6.1 Student Results

A key step in developing software of any type is the testing phase, during which time a developer receives suggestions on improving different aspects of the software. With our labs complete except for finishing touches, we scheduled a test day for the web-based labs. During this day, the regular statistics students performed our lab instead of the standard SAS lab. Using a questionnaire, we were able to get opinions of the overall usefulness of our lab from nearly 100 students. We were also able to address the students' concerns on a more personal basis, because the 100 students were divided into four small groups split up over the day. Using this collected information we were able to determine the key areas of our lab in need of improvement.

In our survey (Appendix page 41) we decided to use a majority of short answer and numeric value questions. Due to our volume of surveys, 100 students with two surveys each, it would have been too difficult to interpret many short answer opinions. Our first six questions are all answered with a numeric value representing the level of various attributes of our lab. Questions 10 and 11 and the lab-specific question were simply answered with a choice. The numeric and multiple choice answers were then inserted into an Excel sheet (see attached disk) along with other information such as the student's section number, sex, major and year of graduation. These data were then analyzed for trends and patterns using SAS, so that we would know the general student opinion of our labs.

We had the students evaluate how successfully the lab achieved each of five goals, using five questions (one per goal). The responses were on a scale of 0 to 10 with 0 indicating complete lack of success and 10 complete success. These questions ask about the user interface, clarity of instructions, clarity of objectives, adherence to objectives and the general level of conceptual knowledge gained. The sixth question asks about the level of complexity of the lab, with a 0 being too easy and a 10 being too hard. The remaining three multiple choice questions ask about the student's previous statistical education, the usefulness of the labs' portability and either about the preferred scaling option in lab 2.2 or whether or not the strategy check boxes were confusing in lab 2.3.

Tables 1 and 2 show the basic summarized results of our survey. With a quick glance, we can determine that the student opinion of both labs was positive. The data from Lab 2.2 shows high responses for questions 1 through 4, and a somewhat lower response for question 5. The response for question 6 is slightly lower than its ideal value of 5. This response to question 5 suggests that the students did not feel the lab was as successful at conveying the concepts as it should have been. The response to question 6 along with comments provided by the students shows that the lab was a bit too easy. This could be improved with a more involved lab, but cannot be entirely avoided with introductory material. Lab 2.3 scored well on all the questions, dipping only slightly on question 5.

Table 1: Lab 2.2 Results Summary

| Question: | Q1 | Q2 | Q3 | Q4 | Q5 | Q6 | Q10 | Q11 | Scaling |
|---|---|---|---|---|---|---|---|---|---|
| Mean: | 8.8 | 8.7 | 7.7 | 8.1 | 6.5 | 4.7 | %37.5 | %92.9 | %82.1 |
| Std. Dev.: | 1.6 | 1.5 | 2.3 | 1.7 | 2.4 | 2.0 | % Yes | | % Auto |

Table 2: Lab 2.3 Results Summary

| Question: | Q1 | Q2 | Q3 | Q4 | Q5 | Q6 | Q10 | Q11 | Confusing |
|---|---|---|---|---|---|---|---|---|---|
| Mean: | 8.6 | 8.2 | 8.0 | 8.5 | 7.7 | 5.2 | %35.7 | %89.9 | %7.1 |
| Std. Dev.: | 1.7 | 2.1 | 2.3 | 1.7 | 2.2 | 1.7 | % Yes | | |

Our questionnaire also contained three questions where the students could elaborate on any suggestions they had. These questions involved the student's likes, dislikes and ideas for improvement. We summarized the student's answers in order to concentrate on the main categories of each suggestion. This way we were able to group many similar thoughts together and see where exactly our lab excelled and where we needed improvement. This summary can be seen in the appendix on page 43. Some trivial and possibly sarcastic comments have been omitted from this table. This should not affect our analysis in any way as only one or two students answered the questions this way.

The tables list the suggestions for labs 2.2 and 2.3 in order of most importance and so it is easy to see which issues should be addressed first. For example, the students think that the most important fix for lab 2.2 would be to improve the clarity of the instructions. The students feel that we need to resolve some ambiguity of the instructions and possibly the interface for lab 2.3. For both labs, sections explicitly stating the objectives were added. The background material was also expanded for both labs, especially the section explaining the various strategies for Lab 2.3.

Many students responded that the issues with printing should be resolved. They were resolved, but not in the manner the students had probably intended. Due to implementation issues, direct printing was dropped from the lab. Students can still print graphs, but only after first saving them to a graphics file.

# 7 Conclusion

Our implementation of a web-based lab proved to be very successful. Based on the observed lab sessions and survey results, it is clear that the students are very satisfied with labs 2.2 and 2.3. The labs provide clear objectives for the students and then clearly and concisely help the students through the labs. In an entry level educational environment, the web-based labs have a major advantage over SAS labs in respect to their quick learning curve and tailored interface. SAS is definitely the more powerful and versatile tool, but a custom web-based lab provides the students with a more accessible and targeted lesson.

In order for our labs to be successful, we needed to provide the students with at least the same educational quality provided by the previous SAS labs. Clear objectives ensure that the students know exactly what material they should extract from our lab. The objectives are listed on the introductory web page for the lab, and are then described in depth in the lab instructions. From our survey results we can conclude that the students feel the lab objectives are well met. Assuming that our objectives are well defined and represent the same material covered by a SAS lab, it is safe to assume that our labs provide the same level of learning that a SAS lab provides. The custom web-based lab can cover the same lesson as a SAS-based lab, without the overhead of learning the software.

Defining clear objectives is not the only requirement for a good lab. The student needs to interact with the software, rather than just merely viewing a non-interactive lesson. By requiring that the student manually manipulate pertinent data we ensure that our objectives will be learned. In lab 2.2

the user must move a slider bar to transform the data while a histogram updates in real time. This presents the student with a clear image of how the transformation is altering the data. Lab 2.3 allows the student to perform five operations by hand which gives the student an interactive experience with process tampering. The automation that occurs afterward shows the student trends in the different tampering methods. This interactivity keeps the student's attention and requires the student to really learn the lesson.

One advantage of the web-based labs over the SAS-based ones is that a lesson is completely self-contained and customized for the material. The web-based approach contains the instructions and interactive lesson all neatly contained in one window. Unlike SAS, the web-based approach requires no setup time for initializing the software or configuring printers, during which the student can lose interest and become frustrated. In addition, the customizable nature of the web-based approach allows for the creation of a solid pedagogical layout maximizing the student's efficiency. While SAS is powerful and complicated, the web-based labs contain a straightforward lesson in a custom environment. Finally, since the labs are all hosted on the WPI math server, they are accessible to the entire campus at any time. Not only does this mean that the students can complete the assignment from anywhere, but specifically, the students do not need access to the SAS software.

SAS has two main advantages over the web-based solution. The first is its power as a highly developed statistical analysis program. An endless amount of data manipulation can be performed for many different combinations of data sets. The web-based solution is clearly not a replacement in any way for SAS, but is best used as a lab instruction tool. The second advantage that SAS has over web-based labs is its macro tools. Labs can be constructed relatively easily in SAS by writing instructions and generating a macro, as opposed to developing an entire application with the web-based approach. However, if a lab toolkit were to be developed, the web-based labs could be easily created with a specific layout standard.

Labs 2.2 and 2.3 proved successful because they effectively teach the student their respective lessons. These labs, along with the SAS-based labs are pedagogically sound and adhere to a basic standard layout which keeps the students familiar and comfortable. There are clearly-defined objectives provided for the students. The web-based approach is superior to a SAS lab because it communicates a custom lesson with less confusion and greater accessibility.

## References

[1] Abbey, Beverly, *Instructional and Cognitive Impacts of Web-Based Education,* (Hershey, PA: Idea Group Pub., 2000).

[2] Brown, David G., ed., *Interactive Learning,* (Bolton, MA: Anker Pub. Co., 2000).

[3] Cooper, Alan, *The Inmates Are Running the Asylum,* (Indianapolis, IN: Same, 1999).

[4] Clein, Robert and Holmes, Samuel, *Online Statistics Labs,* (2003).

[5] Deming, W. Edwards, *Out of the Crisis,* (Cambridge, MA: Massachusetts Institute of Technology, 1986).

[6] Dix, Alan J., *Human Computer Interaction,* (London: Prentice Hall Europe, 1998).

[7] Kawato, Takeshi, *Statistical Teaching Aids,* (2003).

[8] Liesen, Eric and Whitfield, Paul, *Statistical Teaching Aids,* (2001).

## A Appendix

### A.1 Survey

**Name:** ____________________

**Lab Number:** ☐ **2.2** ☐ **2.3**

**Sex:** ____ **Major:** ________ **Class year:** ________

***For both labs:***

1. How easy was the interface to use (on a scale of 0 to 10, 0 being Not at All, 10 being Very Much)? Describe any problems you had.
2. How easy were the instructions to follow (on a scale of 0 to 10)?
3. How clear were the objectives (on a scale of 0 to 10)?
4. How well were the objectives met (on a scale of 0 to 10)?
5. To what extent did the lab aid in your understanding of the concepts (on a scale of 0 to 10)?
6. Was this lab too easy, at right level, or too complex (on a scale of 0 to 10, 0 being too easy, 10 being too complex)? Why do you feel so?
7. What did you like about this lab?
8. What did you dislike about this lab?
9. How can this lab be improved?
10. Prior to this course, did you have any previous statistical education?
11. Is the option of being able to use these labs on any computer with Internet Explorer or Netscape beneficial? Why?

*For lab 2.2 only:*

Which scaling option did you find most useful: Auto scaling, Static scaling, or Zero left? Why?

*For lab 2.3 only:*

Was it confusing having both check boxes and tabs labeled 'Strategy 1' through 'Strategy 4'?

## A.2 Summarized Suggestions Table

Table 3: Lab 2.2 Suggestions

| Likes | Frequency | Dislikes | Frequency | Improvements | Frequency |
|---|---|---|---|---|---|
| Simplicity | 28 | Triviality | 7 | Clarity | 16 |
| Visualization | 16 | Boring | 7 | Saving | 11 |
| Length | 12 | Clarity | 6 | More Complex | 8 |
| Interface | 10 | Simplicity | 6 | None | 7 |
| Clarity | 8 | Printing Issues | 6 | Printing | 4 |
| Ease of Use | 5 | Saving Issues | 6 | Functionality | 3 |
| Organization | 4 | Too Short | 3 | More Examples | 2 |
| Lack of SAS | 4 | Too Long | 2 | Relative Data | 2 |

Table 4: Lab 2.3 Suggestions

| Likes | Frequency | Dislikes | Frequency | Improvements | Frequency |
|---|---|---|---|---|---|
| Visualization | 38 | Ambiguity | 14 | Ambiguity | 13 |
| Ease of Use | 9 | Clarity | 7 | None | 13 |
| Simplicity | 8 | Too Few Cannons | 4 | Clarity | 7 |
| Concepts | 8 | Bugs | 4 | Automation | 5 |
| Interface | 5 | Repetition | 3 | Animation | 4 |
| Lack of SAS | 4 | Boring | 3 | More Cannons | 2 |
| Interesting | 4 | Interface | 2 | Visualization | 1 |
| Organization | 3 | ⋮ | | Sound | 1 |
| Length | 3 | | | | |