INTERACTIVE CONTENT IN WEB PAGES TO TEACH STATISTICS ®

W. Douglas Stirling
Massey University
New Zealand

Computer-based teaching material must contain animation or interaction to offer substantial benefits over delivery on paper. Technology developed for use in web browsers — especially Java and JavaScript — makes it relatively easy to add interactive diagrams to web pages. The object-oriented nature of Java is well suited to developing a large collection of interactive diagrams (applets) for teaching statistical concepts. Many useful statistical objects, behaviours and displays are shared by the applets so their implementation is relatively easy in Java. By designing all applets together as a single collection of linked classes, a resource with hundreds of applets can be created in which interaction plays a major role in teaching all concepts. CAST is used to demonstrate that a complete introductory statistics course with over 330 interactive diagrams can be developed using this technology in a fairly short time.

IMPORTANCE OF DYNAMIC CONTENT

In recent years there has been an explosion in the number of lecturers putting course notes and other expository teaching material on the internet, but many have not stopped to consider what this delivery method offers over traditional printed teaching material. Transfer of static course notes to a web site offers few advantages to students. Nonlinear hyperlinks in web pages can replace traditional page references in textbooks but, unless navigation through a site is well designed, it is easier to get 'lost' when following hyperlinks than when turning pages in a traditional textbook. There are benefits to the lecturer since the material becomes easy to revise, but this does not translate into benefits for students who can become confused if their 'textbook' changes during the course!

These advantages must be weighed against the relative difficulties of reading and studying at a computer screen compared to paper — indeed, many students immediately print notes that are made available on web pages! Migration of teaching material to the internet should be contemplated only if features are added that could not be implemented in print.

Dynamic content must be added to get any real benefit from presenting expository material on a computer and this may be either in the form of video, as used extensively in ActiveStat (Velleman, 1998), other animations that play on the click of a button, or diagrams that require more extensive student interaction. All are useful for motivating students and explaining statistical concepts, but this paper concentrates on interactive content since student involvement in the learning process aids retention. Puranen (1998) further discusses the potential advantages and disadvantages of computer-based delivery of teaching material.

TECHNOLOGY TO SUPPORT DYNAMIC CONTENT

Expository teaching material should be delivered on a platform that supports text, static diagrams, plus video and/or interactive content. Hypercard (or other similar programs) would have been used a few years ago, but the technology that has developed for web browsers has largely superseded it. The major web browsers support text, graphics, audio, video (through plug-ins) and two programming languages, JavaScript and Java, which can be used to implement interactive content.

JavaScript is a scripting language that is relatively easy to use — programs are embedded as text in web pages — but it can only be used to implement simple interactions since it does not support any graphical output other than swapping images. In contrast, Java is a fully featured programming language that can be used to implement diagrams of arbitrary complexity, so most interactive diagrams for teaching statistical concepts are written in Java (West & Ogden, 1998). In web pages, the Java content appears in rectangular regions which usually behave like self-contained application programs (and are therefore called 'applets'). In the next section, efficient implementation of Java-based teaching material will be discussed. It should be noted here that the use of web browsers as a platform for displaying this type of material does not require that the material be delivered over the internet. The 'web pages' can equally well be accessed from a CD or local hard disk without any internet connection.

INTEGRATED DEVELOPMENT OF MULTIPLE APPLETS

When each interactive diagram is implemented with a self-contained program (an applet stored in a separate 'jar' file), development time usually limits the amount of interactive material that authors include in the resources that they develop. Many published browser-based 'textbooks' contain 30 or more Java applets, but this only scratches the surface of what is possible. Translated to a traditional textbook, this would correspond to perhaps one interactive diagram per ten pages.

There is no need for the applets in a teaching resource to be developed as separate programs. The object-oriented nature of the Java programming language allows all applets in a resource to be developed as a *single* 'program'. Many types of statistical object and behaviour are common to a wide range of useful applets and these can be implemented once in core classes that are shared by all the applets. Different applets are specified to the browser as different 'entry points' into the single code hierarchy that is developed. (Technically, there is a single sub-class of Java's standard *Applet* class in the class hierarchy for each applet that is displayed.) The resulting code hierarchy can be considered either as multiple applets that use shared code or as a single 'program' that displays in multiple rectangular regions of the pages (the applets).

If the core class structure is well designed, this approach considerably simplifies the writing of new interactive diagrams since code that is written for any applet can be easily reused for related applets. It is also efficient since the code shared by different applets is only stored once on permanent storage and once in memory, rather than separately for each applet; displaying a new applet may only require loading a small amount of additional code into memory. For example, *CAST* (Stirling, 2000) is an introductory statistics textbook that includes over 330 different interactive diagrams written in Java to help explain statistical concepts. Yet the whole package can be downloaded in a single 5MB zip file.

Before developing this type of resource, core classes should be designed to encapsulate much of the common behaviour of the different applets. In the context of a resource to teach introductory statistics, this basic class library should support data sets, variables, distributions, sampling and summary statistics, plus methods to link these together so that, for example, taking a random sample from a distribution automatically updates any summary statistics and displays of the sample data. In *CAST*, the core framework also includes classes that draw individual values, lists of values, axes, basic scatterplots, rotating 3-dimensional displays and many other data displays that form a basis for those used in the applets.

EXAMPLES IN CAST

Object-oriented design makes it easy to add a modified version of an existing applet with slightly different functionality or user-interface. Therefore a sequence of applets can be programmed relatively easily to develop a particular topic, each successive applet building on an existing applet with little extra programming.

In the simplest case, an existing applet's functionality can be modified by creating a sub-class of that applet's *Applet* class. The sub-class might, for example, change one distribution to another or add a second standard display object that shows the data in a different way. For example, the applet on the left of Figure 1 lets students adjust the width and positions of histogram classes for a small data set. It shows how a histogram's shape can be greatly affected by such choices and therefore that great care must be taken in interpretation of histogram shape when data sets are small. The applet on its right is similar but uses larger data sets that are randomly generated from a skew distribution. It is used to show that the major features of histograms from large data sets are fairly stable. It was implemented with a single sub-class of the previous diagram's *Applet* class that replaces the data set with a randomly generated one and adds a button to generate different random samples — a total of 35 lines of extra Java code.

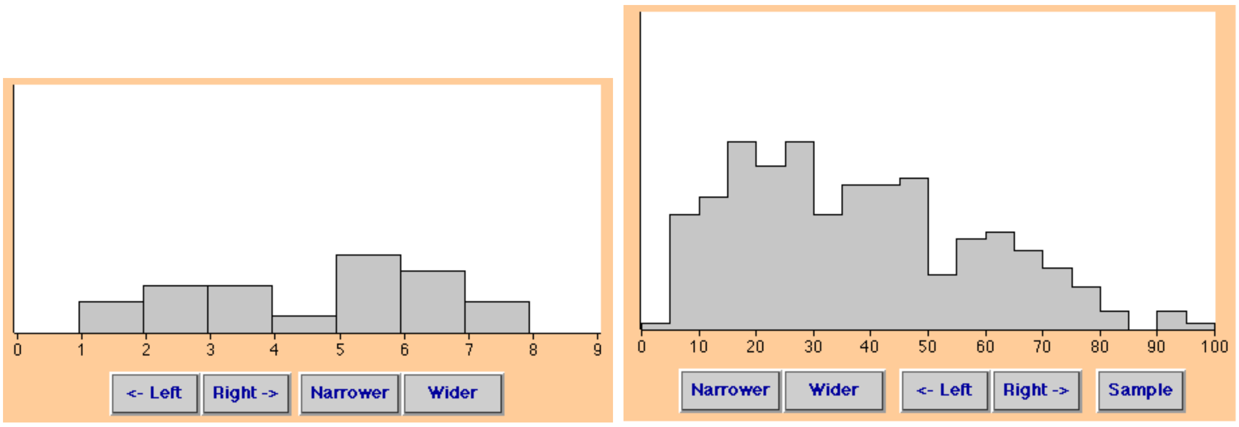


Figure 1. An applet to show the effect of changing histogram classes in small data sets (left) and a related applet showing how random sampling affects the shape of histograms of larger data sets (right).

More often, one or more further classes must be changed (sub-classed) to obtain the desired functionality in a new applet. The top applet in Figure 2 illustrates how running means smooth a time series. Related *CAST* applets show the effect of running medians, sequences of smoothers and exponential smoothing; each is implemented with minor modifications to the *Applet* class and a rewritten *FunctionVariable* class to perform the actual smoothing. The applet on the bottom of Figure 2 adds still further functionality with a different *FunctionVariable* sub-class that uses exponential smoothing to forecast one or more time periods into the future and a new *DataView* sub-class to display the error sum of squares. Students can adjust the slider to minimise the error sum of squares for the forecasts.

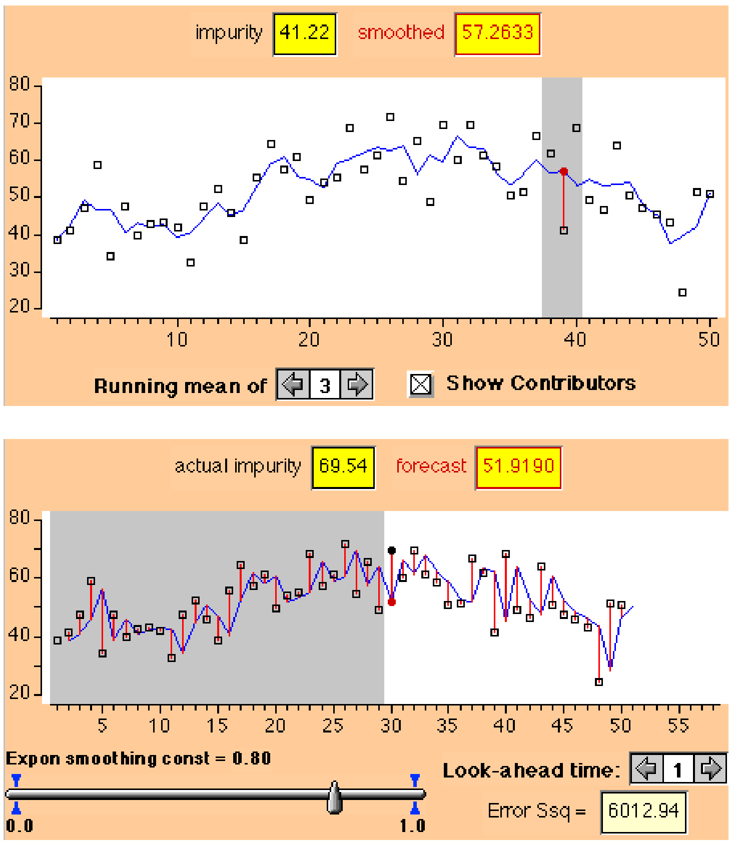
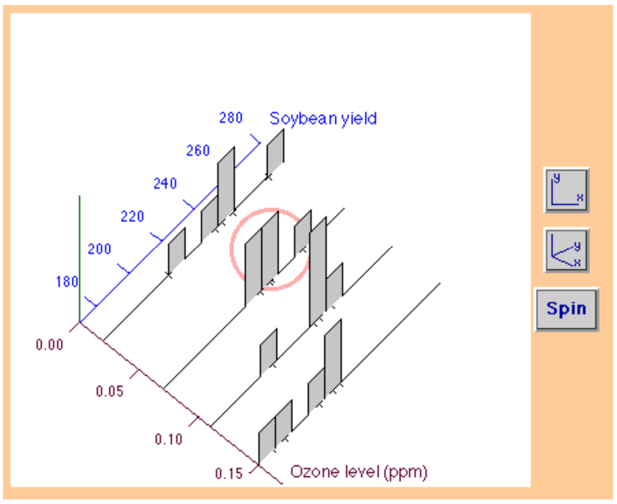


Figure 2. Applets showing a time series data set with running means (top) and with an exponentially smoothed series to predict into the future (bottom).

The Java class that displays data in the *CAST* applets is called a *DataView* — sub-classes draw the histograms and time series plots in Figures 1 and 2 and also the error sums of squares and highlighted values and forecasts in Figure 2. New applets often need to display data and models in new ways and may also involve novel interactions with the user (often by clicking or dragging in the display). New sub-classes of a *DavaView* are usually needed to implement these displays.

For example, a *Rotate3DView* is a standard *DataView* that draws a 3-dimensional display of data against three axes and implements rotation of the display either by dragging with the mouse or a continuous 360˚ spin. Sub-classes can be written to draw data or models in different ways against the three axes. The two diagrams in Figure 3 contain sub-classes that are used to introduce the idea of normal linear regression models. The applet on the left displays a data set in which repeated response measurements have been made at each of four values of an explanatory variable. The data are represented by histograms in this diagram and, using similar reasoning to that motivating the normal distribution as a model for univariate data, students are shown the applet on the right as a potential model.



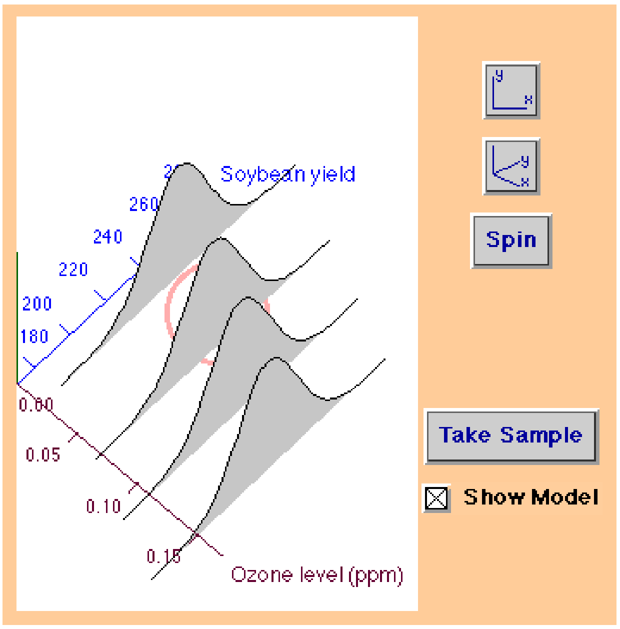


Figure 3. Applets showing a regression dataset with five response observations at each value of an explanatory variable (left) and a possible regression model for these data (right). Both diagrams represent 3-dimensional objects that can be rotated with the mouse.

IMPLICATIONS FOR TEACHING

The ease with which new applets can be written has had a major impact on how topics are introduced in *CAST*. Concepts can be approached with a sequence of simple interactive diagrams, each targeting one aspect of the topic, rather than a single all-encompassing complex applet. Since it is possible to include as many interactive applets as the number of static diagrams in conventional textbooks, there is real scope for improving on paper with this type of computer-based resource. For example, the concept of least squares is introduced in *CAST* with a series of interactive diagrams. All are based on an applet that draws a line against two axes.

- To revise the concepts of slope and intercept, a diagram allows students to adjust the two parameters (with sliders) and observe the effect on a line.
- A diagram then shows how a line can be used to predict *y* from *x*. Students can drag an x-value and observe how the prediction is determined both graphically and using the line's equation.
- A data set containing temperatures and latitudes of US cities is used to introduce the concepts of fitted values and residuals. Clicking on crosses on the scatterplot displays the city name, actual temperature, temperature predicted by the line and the residual.

- The next diagram shows a data set in which all residuals shown as vertical red lines. Students can drag the line to observe the effect on the residuals.
- The last diagram is a scatterplot that represents the squared residuals as squares and displays the sum of squared residuals. Students are asked to drag the line to minimise the total area and a button is available that minimises this total.

This approach means that each applet is only aimed at a single concept so its user-interface can be kept simple. By using a series of interactive diagrams to introduce a concept, they become a much more integral part of the text.

CONCLUSION

Web browsers are an excellent platform for presenting teaching material because Java applets can be easily added to provide interactive content and without this interactive content, videos or animations, computer-based learning material has few advantages over paper-based material. An integrated approach to programming the applets within a single class hierarchy greatly simplifies the programming task since new applets can often reuse substantial amounts of code that was written for earlier applets. However this requires a substantial initial effort to design core classes that are flexible enough to form a basis for the applets. This incremental approach allows many more applets to be written than most current developers have been able to develop. The ability to use a series of simple applets to introduce and explain topics allows interaction to become a much larger part of the learning experience of the students.

REFERENCES

Puranen, J. (1998). WWW and teaching statistics – a teacher's point of view. *Proceedings of the 5th International Conference on Teaching of Statistics* (pp.993-998). Singapore: International Association for Statistical Education.

Stirling, W.D. (2000). *CAST.* Accessible from http://cast.massey.ac.nz

Velleman, P. (1998). *DataDesk.* Addison Wesley Interactive.

West, R.W., & Ogden, R.T. (1998). Interactive demonstrations for statistics education on the World Wide Web. *Journal of Statistics Education 6*(3).