On Categories of Interactive Computational Web Tools

Linda Becerra, Ongard Sirisaengtaksin, and Bill Waller
University of Houston-Downtown
Houston, Texas
ongards@dt.uh.edu

Abstract

Web-accessible mathematical activities can play a major role in providing a student learning support in the form of supplements, computational tools, or tutorials. These activities can be perfectly integrated into a Web course as well. Mathematical Web tools have several advantages over calculators and CAS software in that students can access them from virtually anywhere at no cost. Most of the mathematical Web tools contain an interface that alleviates any prior knowledge of syntax for a particular CAS software. Several authoring software applications are available to create mathematical Web tools. Each mathematical Web tool has its own structure, features, and means of accessibility and development. In this paper, we describe a classification of mathematical Web tools based on these properties. Comparisons between the categories of mathematical Web tools are also discussed, and general information is provided about what is involved in creating mathematical Web tools for each category.

Introduction

Accessibility is still a key issue when attempting to integrate technology into math courses. Even graphing calculators, not to mention CAS software licenses, can be a financial burden for students and universities. Today's busy students, moreover, complain about having little time to spend in campus laboratories. In addition, the push for distance education and online math courses has highlighted the need for making computational tools readily available via the Internet.

Recently, many authoring software, such as Mathwright [1], LiveMath [2], Wims [3], J/Link [4], and MathScript [5] have emerged that can be used to make computational and CAS tools accessible to students over the Web. With Mathwright and LiveMath, online "notebooks" can be created that provide students with calculator facilities and are opened via freely downloadable browser plug-ins that supply the computational engines. On the other hand, Wims and MathScript use programming or scripting languages that can be used to create HTML pages to provide remote access to computational kernels. Moreover, a web programming language such as Java provides a means in creating computational tools in the form of applets. Many websites contain notebooks, Web pages, or Java applets developed with one of these authoring tools or Java, covering both standard and more specialized undergraduate topics.

In this paper, we describe three different categories of interactive mathematical Web tools based on their structures and their functionalities: plug-in based, server based, and applet based.

- A plug-in based application requires a certain set of components at the local computer to work with the application. Plug-ins must be downloaded and installed locally to run a plug-in based application of that type, but this download only needs to be done the first time the

plug-in is utilized. The plug-in might consist of a class library, data link library or computational kernel.

- The server-based application uses a client-server model where the computational kernel resides on a server. This type of application consists of two parts: the client-side component and the server-side component. The client-side part is an interface responsible for gathering data from the user interface that is required by the server-side component and sending it to the server-side component. Then the server-side component processes the data it received and sends the results back to the client-side component for display. The server-side component may need a computational engine or kernel to process the data.
- An applet-based application is a Java applet that is in the form of byte code. The byte code will be executed by the Java virtual machine while the browser is loading the application. The components a Java applet needs are files from the Java class library that it uses when it was created. The browser will download the required classes it needs to run the applet.

We will illustrate examples of each category and cover some of the technical aspects in creating and publishing such tools via the Internet. A variety of examples on different categories are available at the Web site [11]. Advantages and disadvantages of these tools will also be presented.

Interactive Computational Web Tools

As the Internet or World Wide Web becomes more and more accessible, many educators are increasingly attempting to integrate technology into math courses. The technology integrations might be in the form of online applications such as Web courses, course supplements, computational Web tools, online tutorials or interactive online laboratories. A computational Web tool is a generic online software application that emulates and enhances many key functions of a graphing calculator, including graphing functions, tabulating functions, curve fitting, graphical animations, general calculations, and symbolic computations. They must be easy to use with no command syntax required. Recently, many authoring software, such as Mathwright, MathView, Wims, J/Link and MathScript have emerged to create online applications that are accessible to students over the Web. With Mathwright and LiveMath, online "notebooks" can be created to provide students with calculator facilities that are accessed via freely downloadable plug-ins that may supply computational engines. On the other hand, Wims, J/Link, and MathScript use programming or scripting languages to create HTML pages that provide remote access to computational kernels. Moreover, a web programming language such as Java provides a means for creating computational tools in the form of applets. Many websites contain notebooks, Web pages, or Java applets developed with one of these authoring tools or with Java, covering both standard and more specialized undergraduate topics.

Interactive Web applications can be classified according to their structures and functionalities as follows:

1. Plug-in based

A plug-in based application can be executed over the Web with a required set of files on the client computer. These files may be data link library files and/or a library of classes used by that application. Normally, this set of files includes a computational engine to perform any type of computation needed by the application. If these files do not exist on the client computer, the

application usually will initiate a message prompt for the user to download that particular plug-in. Once the plug-in is downloaded and installed, the user can always run that type of application without having to download the plug-in again. Every time a plug-in based application is accessed, it will be stored on the client computer temporarily until the application is closed. But the plug-in files reside on the client computer permanently after they are installed.

Most of the plug-in applications are created using their own language or scripting language. A scripting language is similar to a high-level computer programming language such as FORTRAN, C++, or Java. But these scripting languages have been developed to implement their specific features.

To develop applications using a special type of software tool such as LiveMath or MathWright, one needs to know the language required by that software. For example LiveMath, previously called MathView, uses its own computational engine and has its own language. MathWright has a scripting language similar to Lisp.

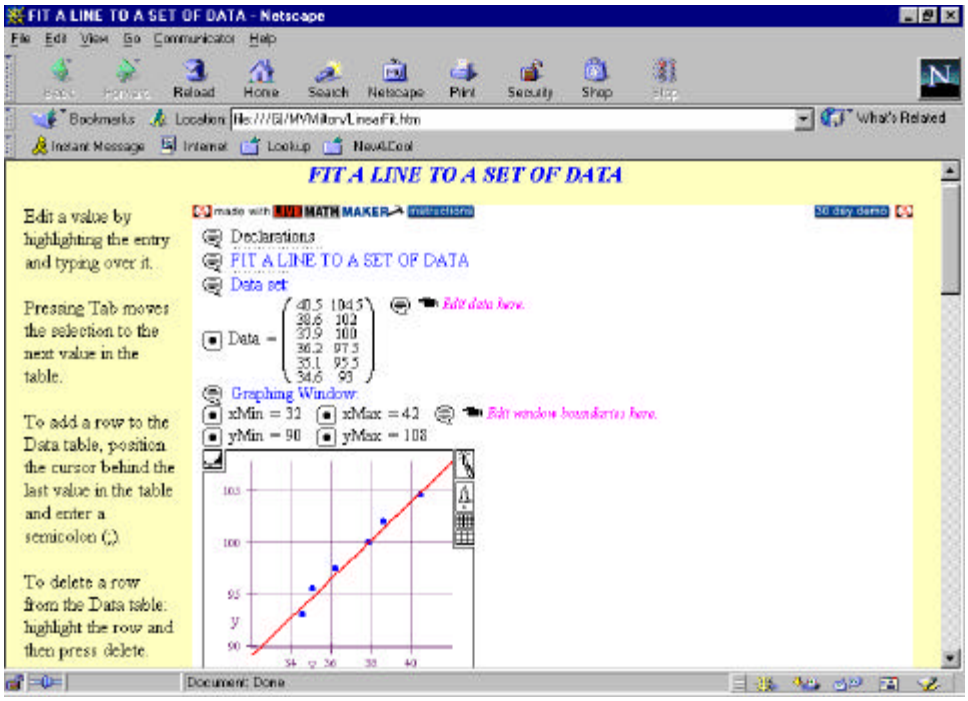

Figure 1: LiveMath Application - Fit a line to a set of data

1.1 LiveMath MathView, originally called Theorist and released in 1989, was purchased by Waterloo Maple in 1993. Then Waterloo Maple turned around and sold it to WebPrimitives in June 1999. Since then MathView has become LiveMath. LiveMath is a computer algebra system with a unique graphical interface. It has an intuitive graphical tool bar that users can use to easily set up computational applications with very little programming experience. LiveMath allows users to complete algebraic, numerical, and graphical computations with its dynamic recalculation feature. As an original input expression is changed in an interactive *notebook*, the computations are automatically recalculated. To share LiveMath applications via the Web, the authoring tool called LiveMath Maker must be used to create

files called notebooks, and then construct a Web page with the links to the notebooks. The LiveMath plug-in must be installed on a local computer in order to access any notebook. Figure 1 is an example of a LiveMath notebook, *Fit A Line To A Set Of Data* [6]. This application is designed to fit a line to a given data set. LiveMath notebooks are capable of including graphing functions, tabulating functions, curve fitting, graphical animations, general calculations, and moderate symbolic computations.

1.2 MathWright MathWright is a mathematical and scientific software construction set. It is an authoring software to create or read documents called WorkBooks. These WorkBooks support interactive mathematical and scientific exploration. WorkBooks are created using Mathwright Author when it is set to *Author Mode*. Then WorkBooks may be read utilizing Mathwright Author when it is set to *Reader Mode*, or they may be read in the reader program that is called the Mathwright Library Player. The interactive WorkBooks that authors create with Mathwright Author will be immediately readable by the Mathwright Library Player. Since the Library Player is freely downloadable from the Library website, authors may easily arrange for their students to have access to any WorkBooks they create. Mathwright WorkBooks may place a variety of *Objects*

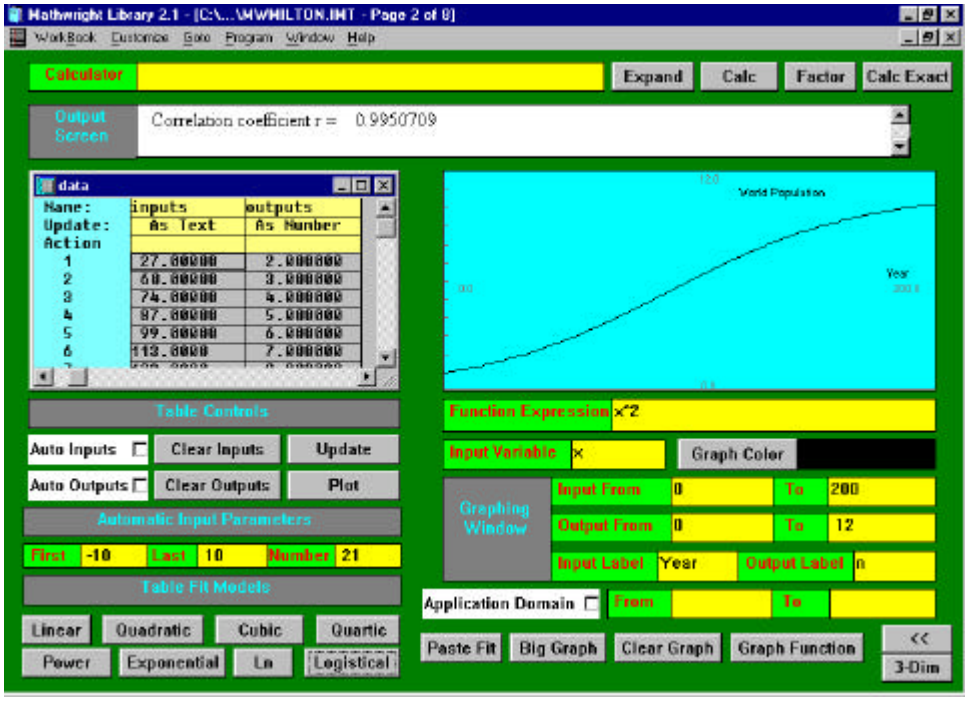

Figure 2: MathWright Workbook – MathWright Milton

on their pages. These objects are automatically connected to one another through a powerful underlying mathematics scripting language called **MathScript**. An author creates an object simply by drawing it, and later by tailoring it to her needs through its menu. Each object has its own menu of properties, and these may be modified at any time. Many of the objects may be "scripted" with their individual Scripts that are built upon both predefined and author-defined commands, programs, and mathematical objects. The objects that an author may create are: Graph2D Windows, Graph3D Windows, MathEdit Windows, Data Windows, Text Fields, Video Windows, Program Windows, Push Buttons, Scroll Bars, Check Boxes,

Wallpaper and Hotspots. (Hotspots behave just like pushbuttons, and they may be used to take the player to another page, to open another WorkBook, to pop-up some information or a message, or to draw a Workbook created with MathWright.) Because of these features, to develop MathWright Workbooks demands good programming skills. Authors may submit Workbooks to the Mathwright Library for publication in the Library. Once in the Library, they may be freely accessed and read by anyone. The Mathwright Library Player reads only "translated" Library books. In this way, authors may design freely distributable materials that support web-based courses. A screen snapshot of a MathWright WorkBook, *MathWright Milton* [7], is illustrated in Figure 2. The MathWright Milton is a computational tool that can perform numerical and graphical manipulations. In general, MathWright Workbooks are capable of including graphing functions, tabulating functions, curve fitting, general calculations, and simple symbolic computations.

2. Server based

A server-based application uses a client-server model where the computational kernel resides on the server. This type of application consists of two parts: a client-side component and a server-side component. The client-side part is an interface that is responsible for gathering data from the user interface that is required by the server-side component, and then sending it to the server-side component. Subsequently, the server-side component processes the data it has received and sends the results back to the client-side component for display. The server-side component may need a computational engine or kernel to process the data.

An interface can be written using HTML *form* components or any programming/scripting language that generates HTML form components. The programming/scripting languages such as Java, Java script, VB script or Perl script can be used to create such an interface for a server-based application. The corresponding server-side component must have been designed to receive the information from the client-side component and process it accordingly. An appropriate scripting language required by the kernel or computational engine must be used to create the server-side component.

2.1 Wims Wims stands for World Wide Web Interactive Mathematics Server [3]. Wims was developed using a systematic approach to provide internet-accessible mathematical computations. It is designed for server-side interactivity on Internet applications through the http protocol. Client-side interactivity via Java script or Java applets can be incorporated into Wims applications. The server-side interactivity is specifically designed to support mathematical computations. This allows Wims applications to use different CAS softwares such as MuPad, Gnuplot, Octave, Imps, COQ, etc., on the server. Wims has a capability for users to develop not only computational tools but also interactive exercises with random parameters and an automatic scoring system. It also offers step-by-step interactive exercises with sophisticated analysis of answers, virtual classes with progress controlled, and message boards. An example of a Wims computational tool is shown in Figure 4, Function calculator [9]. Function calculator allows users to differentiate and integrate functions symbolically and numerically. Wims applications have the capability of including graphing functions, tabulating functions, curve fitting, general calculations, and symbolic computations. Wims also provides other interactive activities such as interactive tutorials and online testing. The best feature of Wims is that it has its own server to host all

the applications at no cost. The home server is in France [3]. However, it also allows a mirror site to be constructed anywhere in the world.

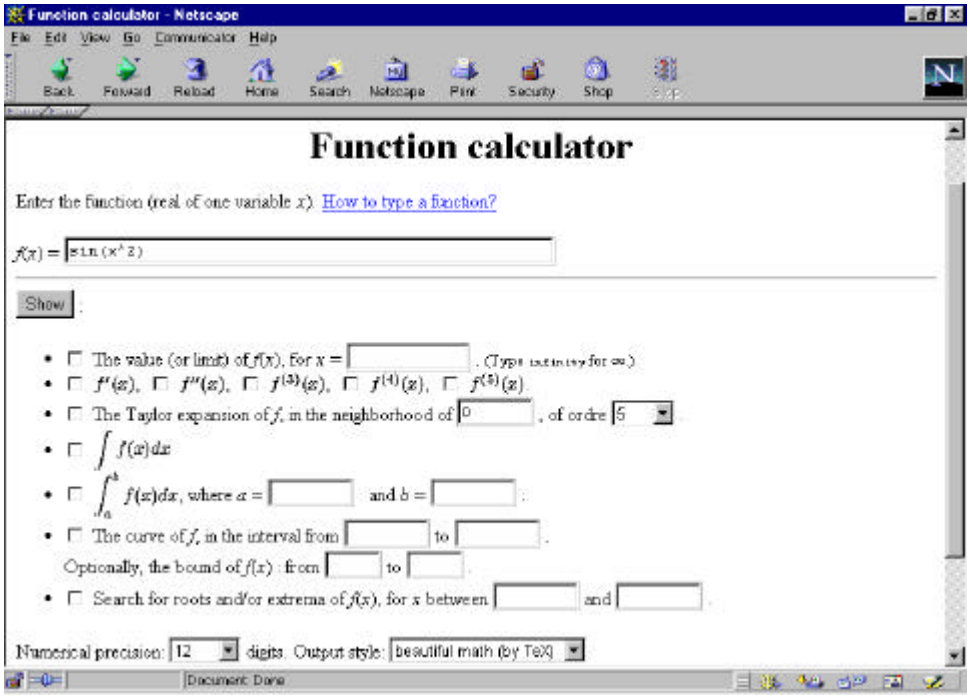

Figure 3: Wims - Function calculator

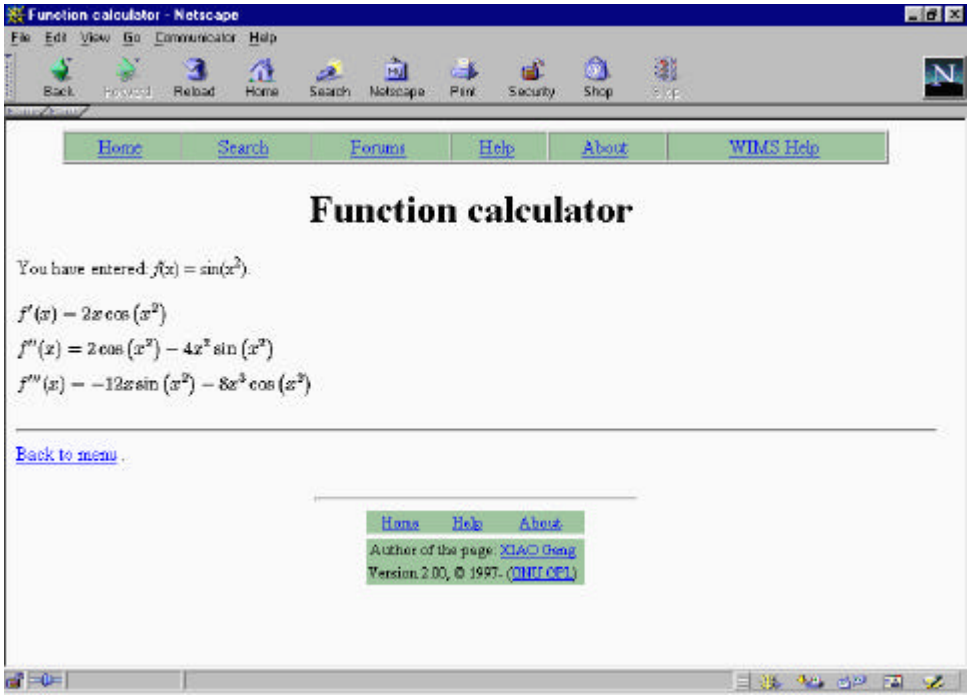

Figure 4: Wims - Result in Function calculator

2.2 MathScript MathScript is a product of Wolfram Research that enables Mathematica models to be run as scripts over the Internet. In addition to technical modeling, the Mathematica language can be used to model web pages and general applications logic.

MathScript Developer allows Mathematica users to interactively develop Web sites as Mathematica models. The MathScript Application Server provides a run-time Internet server

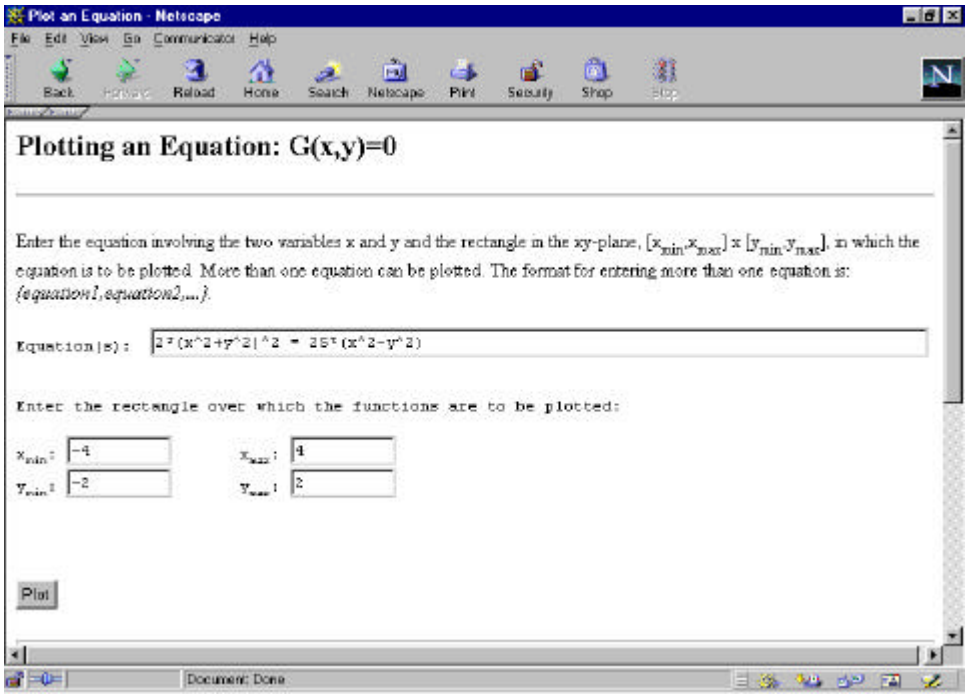

Figure 5: MathScript - Plotting an equation

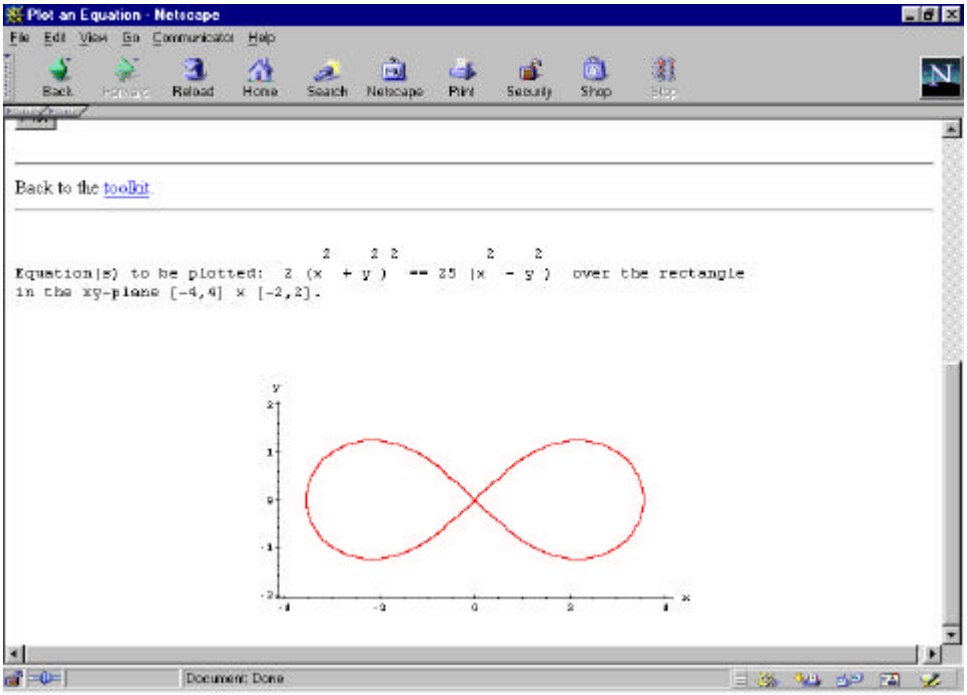

Figure 6: MathScript - Result of a graph from Plotting an equation

for web sites and other applications. The MathScript Application Server can have multiple sites and multiple accesses, but the MathScript Developer server has only a single site and single access. MathScript provides applications written in Mathematica code to interactively communicate with other application frameworks including the Web server CGI (Common Gateway Interface). As previously mentioned, Mathematica can be used to author Web

pages, which are interfaces, and scripts to perform any computations via the Mathematica kernel on the server. But the model is being commonly used to separate the interface or the Web page from the script. This makes it easy to maintain the site.

The process of creating an application using MathScript is as follows. First, write scripts to perform any computations and receive any results back via the Mathematica kernel. Then author an interface using any HTML editor and form elements to collect and display data corresponding to the script that will process the data. The scripts can be created using either MathScript or Perl Script. This step requires that an author have good programming skills. The script files and the interface files can be housed on the Web server, which in turn can be housed on the same or different server as the Mathematica kernel. MathScript applications are capable of including graphing functions, tabulating functions, curve fitting, graphical animations, general calculations, and any symbolic computations. Unfortunately, Wolfram Research discontinued MathScript and replaced it with J/Link, which is described in the next section. Figures 5 and 6 show an example of a MathScript application, Plotting an equation [8]. Plotting an equation graphs any given equation with *x* and *y* variables in the *xy*-plane.

2.3 J/Link Very recently, Wolfram Research has released a new product called J/Link [4], which is basically a Java version of MathLink. MathLink is a protocol for sending data and commands back and forth from a Mathematica kernel to other programs. MathLink is intended for Visual Basic, C and C++ developers to write custom front ends or interfaces to Mathematica. J/Link integrates Java and the Mathematica kernel. It allows Java applets to use the Mathematica kernel on the server. The first step of creating an application using J/Link is to write scripts to perform any computations and receive any results back via the

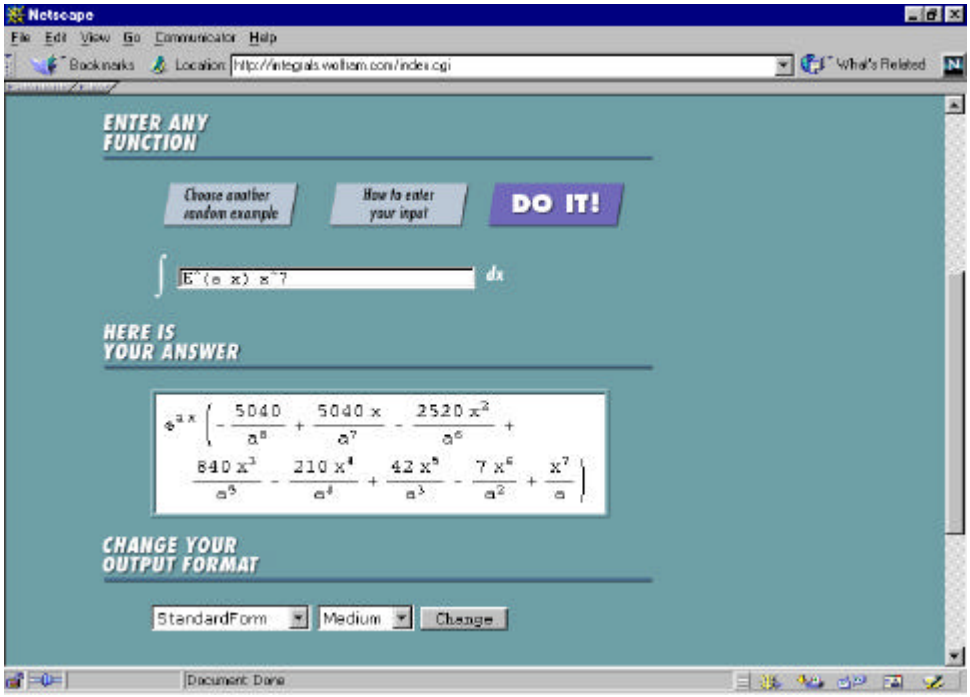

Figure 7: J/Link - The integrator

Mathematica kernel. Then the author must create an interface using Java applets and form elements to collect and display data corresponding to the scripts that will process the data.

This requires that an author have good programming skills. An example of a J/Link application, the Integrator [10] is illustrated in Figure 7. The integrator performs algebraic indefinite integrations. Similar to MathScript, J/Link applications make use of a Mathematica kernel, and are capable of including graphing functions, tabulating functions, curve fitting, graphical animations, general calculations, and any symbolic computations.

3. Applet based

An applet-based application is a Java applet that is in the form of byte code. When a client computer accesses the applet, the byte code of the applet will be executed by the Java virtual machine on the client computer. The components an applet needs are files from the Java class library that were included when the applet was created. The browser will download the applet along with the classes that are required when running the applet. Only the classes the applet needs to be executed will be downloaded and stored in the client computer. The Java applets do not have a computational kernel. The computations required by the applet are built into its byte code. This puts limitations on the ability of Java applets to perform any algebraic or symbolic computations.

There are three steps to create Java applets: first, design and write code for the applet; second, create a Java archive for the applet; and third, create a Web page that contains the link to the applet. Since Java is an object-oriented programming language, knowledge of coding in object-oriented programming is needed in order to create a Java applet. Any text editor can be used to write code for the applet, and then it can be converted from Java code into byte code through the Java Developer Kit. But it is far more convenient to develop applets using a Java compiler, such as JBuilder or VisualCafe. After an applet is created, the byte code will be produced and put into a Java archive together with the classes that the applet uses through the deployment process. Then a Web page will be built that contains the link to the applet. Java applets are capable of including graphing functions, tabulating functions, curve fitting, graphical animations, and general calculations. Figure 8 shows an applet, Function animations [12]. The Function animations applet is used to investigate the behavior of functions as the values of a coefficient are changed.

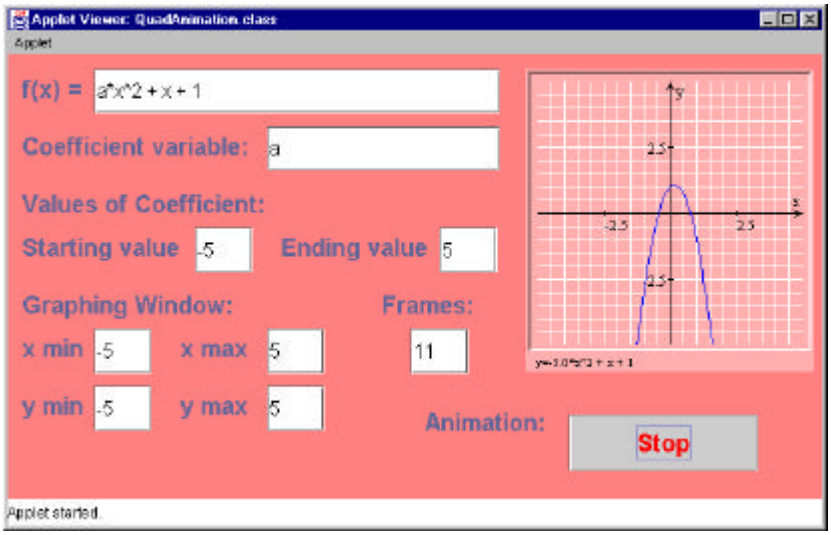

Figure 8: Java applet - Function animations

Summary

As previously mentioned, our objective is to provide readers with information on the categories of computational Web tools. We presented some software tools used to develop computational tools. Each of these software tools has advantages and disadvantages that depend upon the type and sophistication of the applications being developed. The following table summarizes the features of the software tools described in this paper.

Table 1: Summary Table

Feature	LiveMath	MathWright	Wims	MathScript	J/Link	Java Applet
Numerical	yes	yes	yes	yes	yes	yes
Symbolic	moderate	simple	moderate	extensive	extensive	no
Graphing	yes	yes	yes	yes	yes	yes
Animation	yes	no	yes	yes	yes	yes
Retail Cost	$99	$250	free	not available	$1495	free
Learning Curve	simple	extensive	extensive	extensive	extensive	extensive
Plug-in	yes	yes	no	no	no	no
Kernel/server	no	no	yes	yes	yes	no

References

[1] MathWright, http://www.mathwright.com/

[2] LiveMath, http://www.livemath.com/

[3] Wims, http://wims.unice.fr/~wims/

[4] J/Link, http://www.wolfram.com/solutions/mathlink/jlink/

[5] MathScript, http://www.mathscript.com/

[6] Fit a line to a set of data, http://cms.dt.uh.edu/Faculty/BecerraL/MVTitlePage.htm

[7] MathWright Milton, http://www.mathwright.com/

[8] Plotting an equation, www.math.vanderbilt.edu/%7Epscrooke/toolkit.shtml

[9] Function Calculator, http://wims.unice.fr/~wims/wims.cgi?session=8BD2E6F3.2&lang=en&module=tool%2Fanalysis%2Ffunction.en

[10] The Integrator, http://integrals.wolfram.com/

[11] Putting Computational Tools Online, http://cms.dt.uh.edu/faculty/becerral/links.htm

[12] Function animations, http://mathcsjava.emporia.edu/