Chapter 6

Future Improvements

Chapter 5

Design of the Implementation

5.1 Layout of the Program

The application is separated into several different screens, each describing or demonstrating convolution a different way. The different screens follow each other in order, and can be reached through "Previous" and "Next" buttons, or any particular screen can be accessed by going through the menubar.

-structure of the program, i.e. in screens that can be turned to, etc

5.2 The Screens

-description of the different screens that are in the program

Problems I had with writing "sliding functions" screen:

Many delays were introduced because of the lack of mathematical functions in cT. There is no way in cT to easily have 3 different graphs concurrently, so it is necessary to manually change from graph to graph. A lot of time had to be spent writing preliminary functions to handle and initialize the arrays and the graphs.

with all three platforms. Given enough work, because of its more low-level functions, a smoother interface could even be possible using Xtoolkit.

4.4 Program Maintenance

Maintainability of the program is very important in an educational environment, especially since versions will have to be continually updated, with the upgrades of the platforms, the changing needs of the professor, and the fact that there is a high turnaround of programmers. C programs are always quite difficult to maintain by other programmers, or even by the same programmer after a few months. Although the language is powerful, it does not encourage a style that is very easy to understand. cT will be easier to maintain, because there are already two courseware applications being used that are written in cT. Because of the existing applications, there are already people here proficient in cT, and they will not have to learn another new language to maintain the program. Because of there are no existing apps in Matlab, it also received a low score, which was higher than the score given to C++ because of the ease with which Matlab can be learned.

4.5 Choice of cT

Although Matlab is somewhat easier to program in than cT, the resulting user interface will be about the same, and some testing showed that it could even be smoother in cT. As shown by the results of the decision matrix, The existing support and easier maintenance for applications written in cT, including the mailing list and the courseware applications already written, outweighs any disadvantage of cT being harder to program.

good support for cT as well, in the form of the ctug mailing list. There were many minor difficulties that existed in cT that would not have happened with Matlab. Matlab was designed mainly for mathematical applications, so it has incorporated automatic array handling and many matrix and other high-level mathematical functions. The platform that is hardest to program in is C++. Both cT and Matlab (with its Graphical User Interface module) were designed for presentations, to allow the programmer to write a good interface with little difficulty. Xtoolkit is very powerful, but has fewer high-level primitives to help with interface programming.

4.2 Portability

Portability of the application is important to ensure that the application will not become obsolete when the hardware becomes obsolete, as had happened with the previous random variable application. (It would only run on VAXes, which are now being phased out of the computer system here.) Both cT and Matlab were given high scores in this aspect, with programs able to be interchanged among Macintoshes, 80x86-based PC's (running Windows), and workstations with Xwindows, with minimal changes in the program. I ranked Matlab slightly higher because of the fact that it is supported by a strong company (Mathworks) and I feel that it could be more supported in the future. C++ received a lower score on portability, because of the varying graphics systems that are required. As I was planning to write the application in Xtoolkit, it would only work on Xwindows systems, a third of the platforms as the other two.

4.3 User Interface

Because the application is intended as courseware, user interface issues are extremely important. Because Matlab and cT provide simple routines to work with graphics, they have a great advantage over Xtoolkit in this aspect, when compared with ease of programming. However, ignoring that factor, very similar interfaces are possible

Chapter 4

Choice of Platform

When designing my project, the choice of platform to write it on was difficult. The three different platforms I considered for the application were Matlab, cT, and C++ (with Xtoolkit). Different issues that had to be taken into consideration included the ease of use of the user interface and the difficulty to program in each. The following decision matrix displays the relative importance of each of the factors, and how well each platform was able to do each of them.

	Weight	cT	Matlab	C++
Programmability	1	5	9	3
Portability	2	9	10	3
User Interface	3	8	8	9
Maintainability	4	10	4	3
Totals:		85	67	48

4.1 Ease of Programming

Programmability was not given much priority because the only real question was whether the difficulty of the platform would cause the project to not be completed in a reasonable amount of time. Matlab was the easiest to use, with extremely clear manuals, and very good support. cT is next for ease of use, with a built-in editor. The editor enables the programmer to create a better interface with less work. There is

Convolution Equations As with Piscitello's application, there is a screen displaying the equations of convolution. Also, there is additional information available on convolution by clicking on the equations.

Defining/Adding independent variables There are two graphs shown displayed on this screen to represent the two PMFs.

- The user can find definitions of PMF and sample space by clicking on the words.
- *** new way of showing statistics ***

Convolution at Work This screen demonstrates the convolution of two functions, by showing the flipping and sliding of one function over the other.

- There is a slider that can be used to slide one PMF over the other to demonstrate the mechanics of convolution.
- The slider is snapped to the nearest whole number when the mouse button is released from sliding.

Convolution of predefined functions This screen consists of graphs of two functions, and a graph of their convolution. The functions can be set by calling the PMF edit screen.

- There is a button to switch from discrete to continuous variable convolution.
- If continuous variables are selected, a different screen is called to edit the functions.

the program before, is extremely important. Most importantly, the interface has to be very intuitive to use. Users have to be able to be able to figure out the application after looking at it for a short time only. Secondly, a lot of online help is also very useful, so that students can learn from the program more easily.

3.3 Usage Requirements

The design of RVadd is somewhat similar, being mainly based on the old application. It also consists of several different screens, formatted in basically the same way. Many of the features are taken from Piscitello's application, and there are a few added features to improve usability of the program.

There are several features that are relevant to the entire program.

- Links to the World Wide Web, so that the application can be accessed from elsewhere.
- Limited support for the convolution of continuous variables
- Problems designed specifically for students for use with this program
- A new screen is included to edit the PMFs. This screen can be called from any other screen, and includes options to set a PMF to arbitary values, or to predefined functions. PMF values can be set either by dragging sliders to the appropriate value, or by typing in values directly. The user is allowed to enter invalid values for the PMFs, and the program will then send an error message.
- Help screens on how to use each page, as was done previously
- Hypertext explanations and definitions for some terms
- Special fonts to distinguish hypertext from normal text
- "Next" and "Prev" buttons to navigate the screens
- A table of contents in the menu bar to jump to different screens

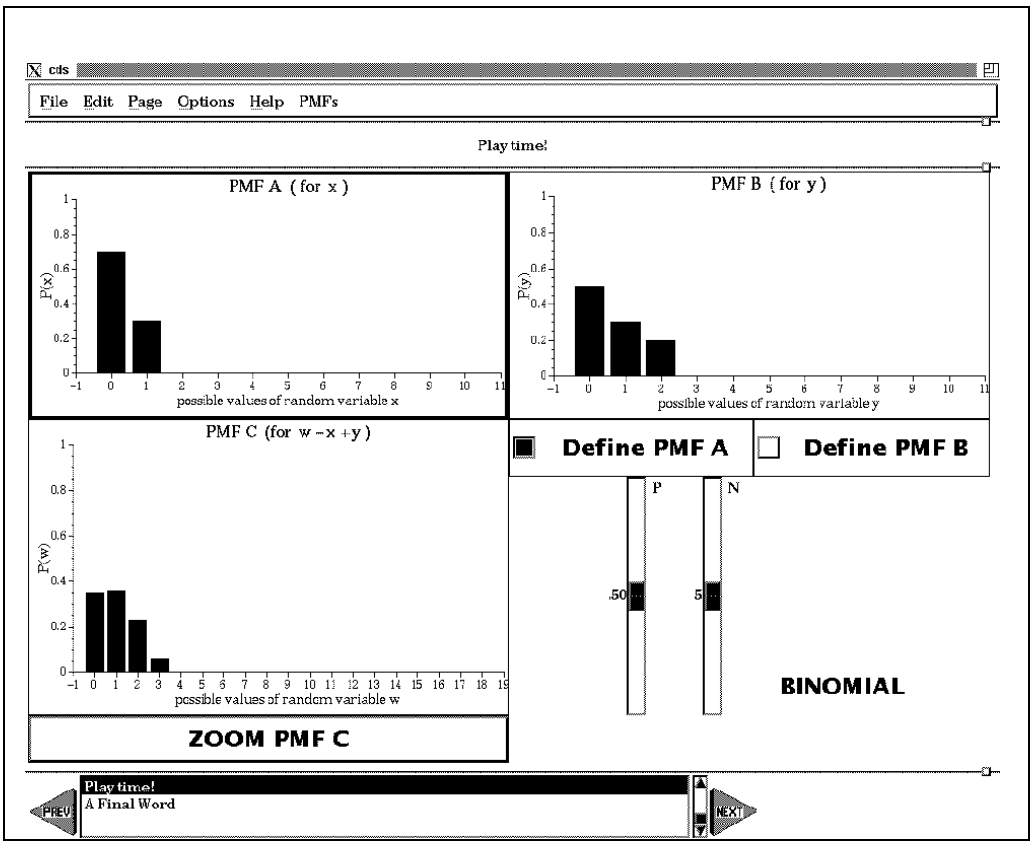

Figure 3-6: Convolving predefined functions

Convolution of predefined functions When this screen is called (Figure 6), there are graphs shown for two PMFs, and for the convolution of the two PMFs. Here, the user can choose many preset PMF functions using the "PMFs" menu option, and the application will graph the convolution of the user's choice. In addition, the button on the bottom left will zoom in on the convolved function, to show it more clearly. Use of this screen was very clear. The only minor problem was the fact that when another screen is called from this one, the menu option for "PMFs" is not erased, so it is confusing in the next screen when the menu option does not do anything. Also, the ability to choose these predefined PMFs in another screen could be useful.

3.2 Design issues

This application is designed as courseware, and hence, most people will only use it once. Thus, the first impression, and the ease of use for people who have never seen

have to switch to another screen, and then go to a screen from there, to edit the PMFs.

- There is no explanation on how to edit the PMFs at all. It is counterintuitive to have to go to another screen to edit, so it is hard to figure out.
- The screen is almost too cluttered to be useful. Cleaner formatting and better arrangements of the bargraphs and explanatory text could improve usability greatly.

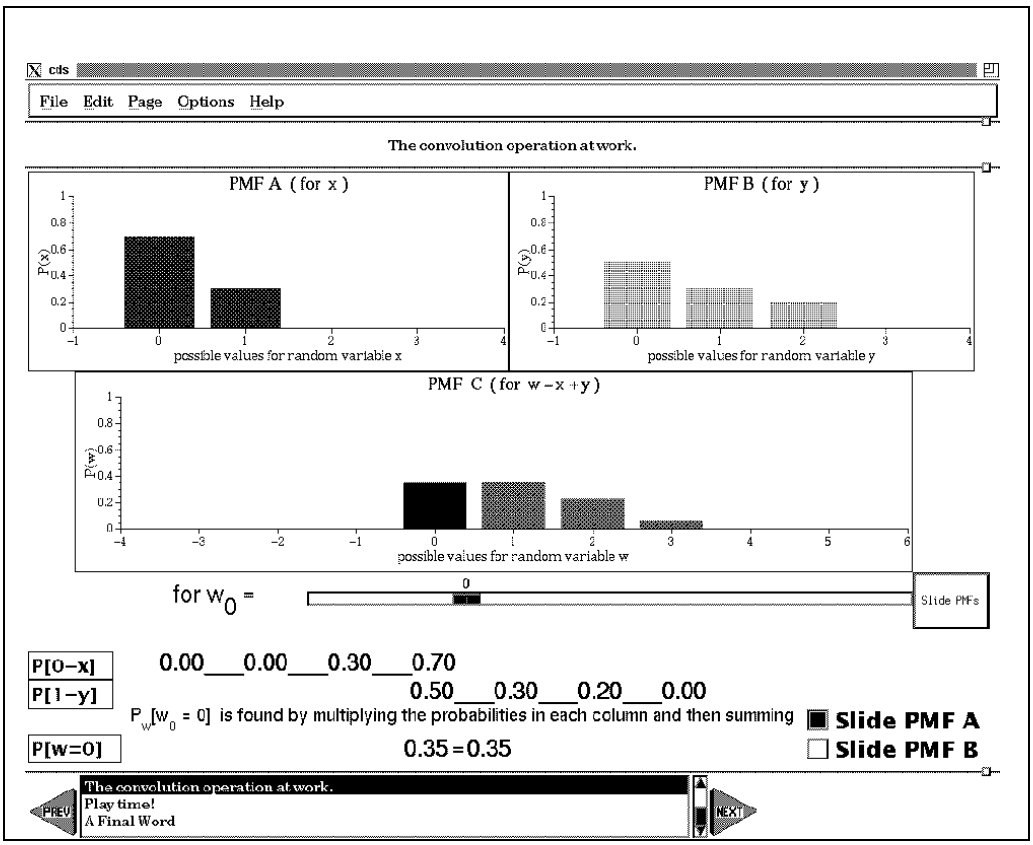

Figure 3-5: Convolution Demo

Figure 5 is the result of clicking on the button. As the user moves the slider, every time the slider reaches a whole number, the graphs will also move, corresponding to the amount they should be "slid". For more information on how this screen works, see ***johnnyp's thesis ***

- There are problems with the screen being slow, because there is a long delay as the computer has to calculate all the new graphs while the slider is being moved.

- The fact that the graphs are sliding over each other is not clear, because they do not move until the slider hits a whole number, and then jump to the next value. It would be extremely helpful if the graphs could smoothly and continuously move while the slider is moved.

- There is no way to edit the PMFs from this screen. It is very inconvenient to

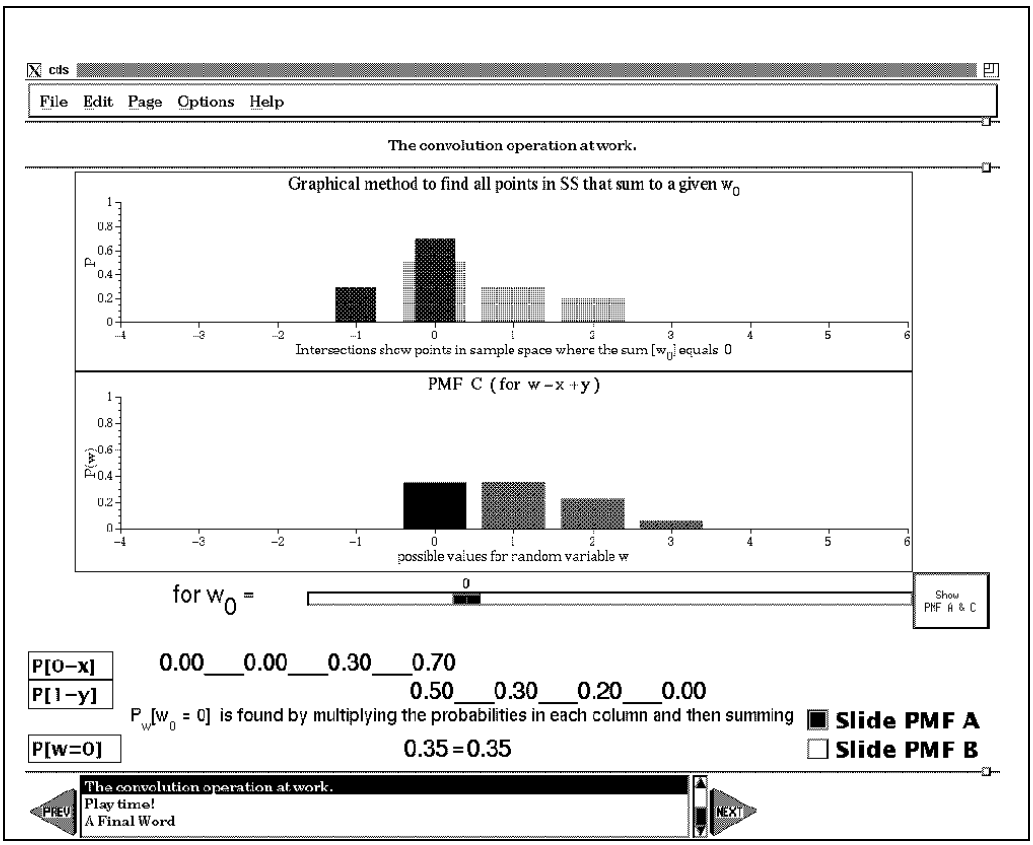

Figure 3-4: Convolution Demo

Convolution at work The next screen, shown as Figure 4, graphically demonstrates convolution, a method to add random independent variables. The top graph shows the first function, with the second function flipped and placed over the first, as the convolution function dictates. The next graph shows the values of the convolution of the two PMFs. The slider under the graphs determines the offset of the two functions in the graph on top, and the current value of PMFC that is being calculated. The button to the right of the slider switches the top graph between displaying the overlapping functions, and showing the two functions next to each other.

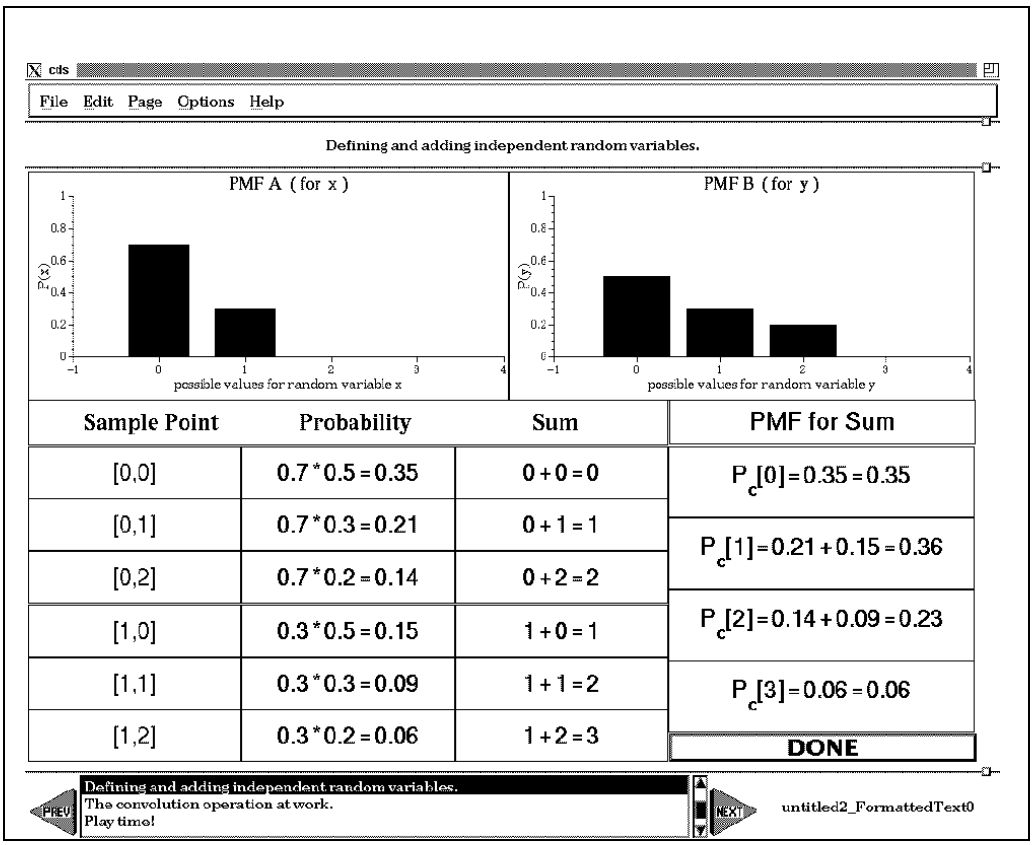

Figure 3-3: Sample Space Algorithm

Clicking on the second button will display a table that tries to show how to add the two pmfs using a sample space (Figure 3). For someone who is not already somewhat familiar with adding random independent variables, this screen is hard to understand.

- There is only one sentence available to briefly describe the screen.
- More help, either as a separate screen, or hypertext, to explain the theory would make this screen more comprehensible.
- A less cluttered display might improve readability. For example, more sub-screens for the table would make it easier to understand.

The third screen accessible from here displays a graph of the result of adding the two functions together.

- Being able to see the exact values of the PMFs on the bargraphs would be useful.
- Being able to type in values for the PMFs instead of having to drag with a slider would be useful.

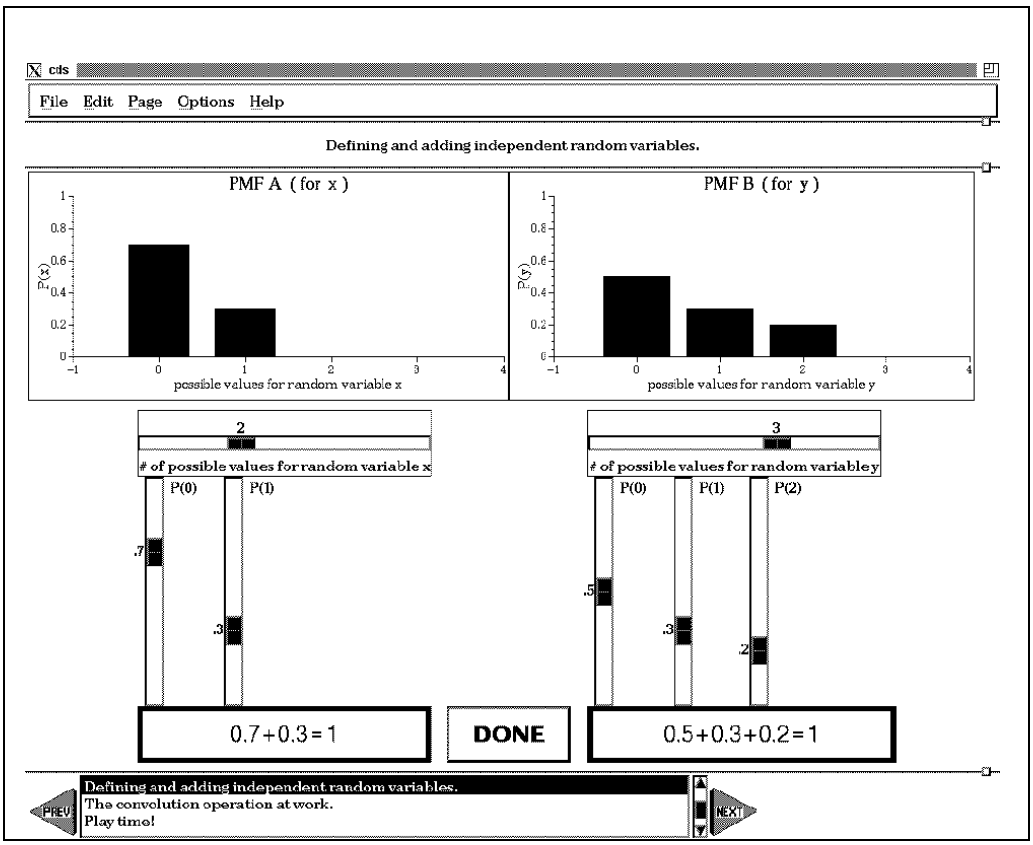

Figure 3-2: Editting PMFs

The first function, to define PMFs, will bring up the screen shown in Figure 2. To change the values of the PMFs, each of the sliders in the figure correspond to one possible value for the variable. When the user is done setting the variable, the program checks whether the sum of the probabilities of each of the values will add up to 1, and sends an error message if not. There are several good ideas on this screen, and also several things that could be improved:

- Allowing the user to edit PMFs to unacceptable values and then printing a warning has good pedagogical value.
- There is immediate feedback on the bargraphs as the PMFs are being editted.
- PMFs should be edittable without having to call another screen.
- Being able to edit PMFs by dragging the bars directly on the bargraph would be useful.

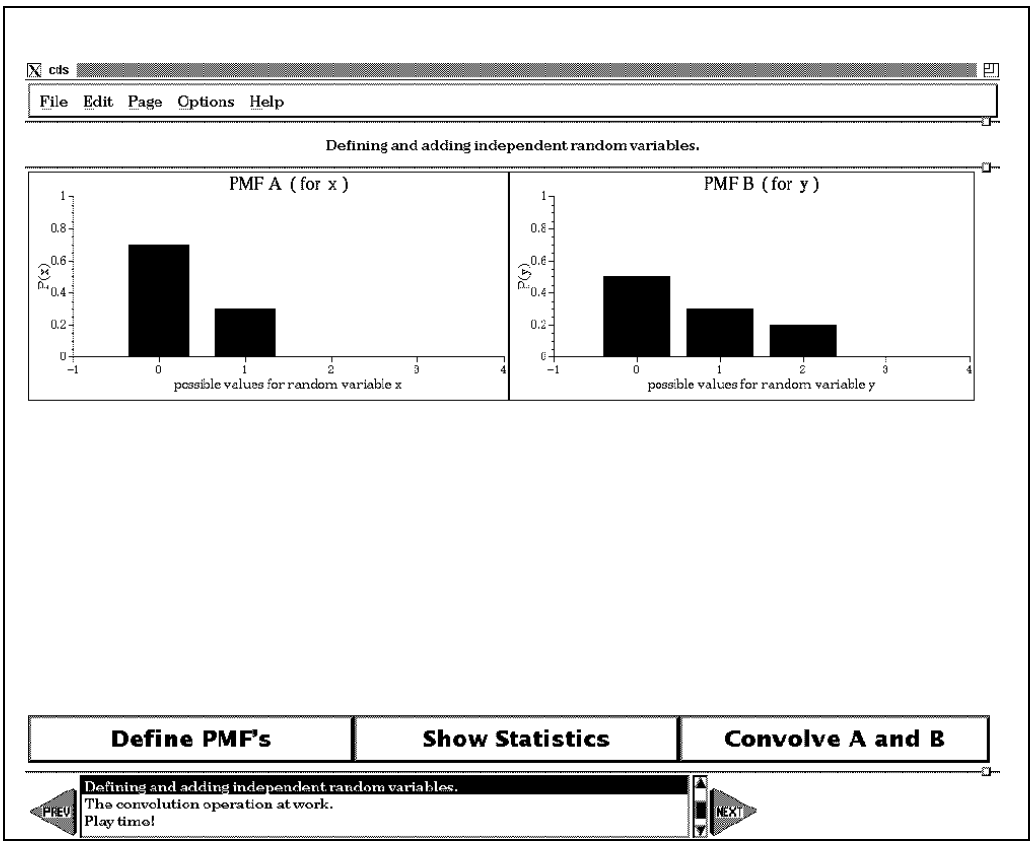

Figure 3-1: Adding Independent Random Variables

Defining/Adding independent variables When this screen is first brought up, it can be somewhat confusing to figure out what to do, especially for students who might use the application for classes other than probability. There are several methods that could help make this easier to use:

- A paragraph introducing random variables and adding them would be useful.
- Definitions of terms, including sample space and PMF, would be very helpful, especially in the form of hypertext.

As shown in Figure 1, there is also a lot of wasted space when this screen is first brought up. There are 3 buttons to click: to define PMFs, to "Show Statistics", and to convolve the two functions.

directly to any of the five screens. For each of the screens, there is help available from the "Help" menu option.

Chapter 3

Design Specifications

3.1 Critique of Previous Application

The old convolution application was set up as four different screens, plus a title page and a credits page. Each screen shows a different lesson or aspect of adding independent variables.

1. The first screen is a display of the equations that describe the convolution algorithm.

2. The second screen shows two PMFs and demonstrates how adding random variables can be derived, using sample space.

3. The third screen shows how convolution provides a simple mechanism that keeps track of all the statistics of adding independent variables. The algorithm is portrayed with a graphical demonstration.

4. The fourth screen consists of the convolution of several predefined functions, to show interesting effects of convolving two functions, including a smoothing of the functions.

The screens can be navigated by using "Prev" and "Next" buttons at the bottom. There is also a list of the screens next to the buttons, with which the user may jump

Chapter 2

Purpose of the Project

2.1 Overview

- intended lesson of application
- intended users of application
- port of the old convolution demo to a new platorm
- why maintainability is so important, history
- background of project

2.2 Theory of Convolution

explain theory, formulas for convolution; uses of convolution

Chapter 1

Introduction

For the past three years, the course on Probabilistic Systems Analysis at MIT (6.041) has used an application developed by John Piscitello to demonstrate how to add random independent variables. This application was written in the Courseware Development System (CDS), which is an authoring tool designed at MIT written for Xwindows. It was designed to make writing lecture and educational programs extremely easy, by providing a simple graphical interface to many powerful backends. The language used by CDS aplications would consist of only a few high-level functions, to maximize the ease of programming in CDS. [Piscitello 8] Piscitello's application was intended for the lecturer, to give an easy way to teach random variables and convolution. Although this application has been extremely useful, CDS is no longer supported and now only runs on VAXstation 3100s. As the VAXes are becoming obsolete, this project was required to port the random variable application to a platform that is still supported. Many of the features of Piscitello's application are incorporated into RVadd, and several new features have been added to make the application even easier to use. In addition, the intended audience has been widened to allow students to actually use the program to learn convolution.

Contents

1 Introduction **2**

2 Purpose of the Project **3**

2.1 Overview 3

2.2 Theory of Convolution 3

3 Design Specifications **4**

3.1 Critique of Previous Application 4

3.2 Design issues 13

3.3 Usage Requirements 14

4 Choice of Platform **16**

4.1 Ease of Programming 16

4.2 Portability 17

4.3 User Interface 17

4.4 Program Maintenance 18

4.5 Choice of cT 18

5 Design of the Implementation **19**

5.1 Layout of the Program 19

5.2 The Screens 19

6 Future Improvements **20**