An Interactive Learning Environment for Adaptive Systems Instruction

Jose C. Principe, Neil Euliano, Curt Lefebvre

Computational NeuroEngineering Laboratory
NEB 486, Electrical and Computer Engineering Department
University of Florida, Gainesville, FL 32611, USA
{principe,neil}@cnel.ufl.edu, curt@nd.com

ABSTRACT

We have developed a new computer based learning environment to teach adaptive systems in the EE undergraduate curriculum. The learning environment is based upon an electronic book containing a hypertext document linked with a software simulator which runs interactive examples on-line, and is fully controlled by the student.

This paper presents the concepts behind the project, its implementation, and discusses the teaching methodology and an evaluation of the first course offering.

1. Introduction

Computer based learning has been heralded as the next generation of teaching tools, but this promise is largely unfulfilled. The difficulty comes from the lack of a clear understanding of what computers bring to the human learning process and how to develop environments which use them effectively. The first question is a research area in itself, so we started by observing how computers are currently being used for classroom instruction.

We see two basic levels in the utilization of computers in education: computer based presentation and computer based instruction. They differ in the level of modifications imposed in the teaching methodology. Multimedia presentations (slide shows) and Web searches belong to the first category since they just make conventional classroom delivery more efficient and appealing, but they leave the teaching methodology largely unchanged. On the other hand, computer based instruction implies a change in the traditional way of thinking about the discipline. It should exploit the computer as an added dimension to bridge the gap between the professor's knowledge and the student's solipsistic attitude.

We believe that computer based presentation alone will not greatly improve the learning environment. In fact it simply speeds the presentation of material and increases the distance between the student and professor -- often putting the student in a passive mode which hinders conceptual understanding and retention. Computer based instruction is becoming more common but in a primitive sense. Computer programs (mostly MATLAB m files) are already complementing DSP textbooks, but they are still add-ons to the text. The presentation of the material has not changed significantly due to the presence of the software. The examples at the end of each section are simply substituted with code segments without integration between the text and the computer exercises. This format is, however, a step in the right direction, since the student gets familiar with the code and the methodology to simulate problems in the environment.

In order to address computer based instruction, we have to ponder both the advantages and disadvantages of computer based learning. One major advantage is the power of simulation that comes from the universal nature of digital computation. Digital computers are in fact formal systems [1], so they can simulate any mathematical solution to a problem. Thus, a computer program is practically and theoretically as powerful a representation as an equation. This is particularly appealing in engineering and digital signal processing, which typically require a strong mathematical background.

The main disadvantage is that the computer is still not as easy to use as a book. Some contributing factors include: small displays which limit the amount of information that can be conveyed at one time, a lack of content based search, a lack of efficient browsing, and possibly the fact that people aren't comfortable using computer based books. These shortcomings are going to be with us for many years, so successful applications of computer based instruction should emphasize their advantages.

Our goal was to teach EE undergraduates adaptive systems. Adaptive systems are not normally taught at the undergraduate level due to the demanding mathematics that are necessary to explain the theory. So the challenge was to explain adaptive systems without using a strong mathematical formalism.

2. A model for the integration of text with a simulator

Our original model for the learning environment was borrowed from the everyday life of the research laboratory where the professor interacts with the graduate student at a high conceptual level. Instead of learning through lectures, the students learn through the results of their research and experimentation, with guidance from their advisor. This environment tends to be unstructured, but is highly effective (we learn much more when we are active players). This implies that the new format should be based on the laboratory model, not upon the classroom model. Software simulators are excellent to recreate real situations, but the problem is that the undergraduate student must be guided very closely and must have a theoretical understanding of the material.

We have devised a way to present the theory of adaptive systems by blending short explanations (including what we call "conceptual equations") with a software simulator that runs "live" examples. The students learn the concepts through both the text and the simulator. Each reinforces the other. Mathematical equations are useful because they precisely condense the information. However, from a practicioner's point of view, the equations do not need to be derived at the same time the concepts are taught.

Soon we realized that we had to reorder the presentation of the material if we wanted a tight integration between the simulator and the text. The reason is that a simulation starts with the basic building blocks, while in the traditional mathematical approach we tend to present the most general case first and then particularize to each situation. If we keep the traditional flow unchanged, the simulator would be used only at the end of the chapter as we find in most present books with MATLAB code. *The need for a different flow in the material presentation and a different teaching style are probably the most valuable lessons learned from our experience.*

3. A methodology for adaptive systems instruction

Adaptive systems are inductive machines. Their parameters are obtained from the data and a set of constraints, using a learning algorithm. The most interesting adaptive systems are nonlinear and distributed. So their analytical study requires sophisticated mathematics, well beyond the level of the typical engineering undergraduate.

Alternatively, a student can learn how to use adaptive systems as they use a CAD tool for VLSI design. They are manipulating a vast knowledge base of topologies and training algorithms which they learn how to use in practical situations along with a methodology to check their results. In essence they learn by contrasting the knowledge contained in the conceptual equations with the observation of the behavior of the system through a set of visualization tools such as scopes, meters, plots, etc. This environment totally exploits the power of the computer since software can provide a repeatable step by step simulation of any mathematically derived algorithm. *We have to remember that a computer algorithm encapsulates knowledge as precisely as a mathematical equation, and through visualization we can use our senses to apprehend the concepts in a more natural way.*

In order to achieve these goals the material has to be presented in a different style and sequence. The style selected includes a motivation (typically a real-world example which grabs the student's attention), a description of the problem, an outline of the methodology and theory, and many examples. The examples illustrate the key aspects of the solution using both fabricated data to elucidate the concepts and real data to assess the applicability of the method. Derivations are included as "hot links" that the student may click for further understanding.

Alternating between the theory and the examples is also crucial. It not only cements concepts while they are still fresh but also keeps the interest level high. After teaching the course for one semester, we believe that each topic should be explained in about 15 minutes and then the students should take at least another 15 minutes to run the examples and change parameters or data files. It is important to point out that the examples are not just "demonstrations", they are interactive simulations which include experimentation by the student.

4. Constructing the textbook

Integrating a hypertext document with a simulator requires many skills that typically do not reside in a single individual. Besides the knowledge and practice of teaching the material, there is a need to have access to a good simulator

and an intimate knowledge of computers. Other practical aspects can not be forgotten such as availability of the software, and universality of the operating system/computer platform.

We chose the Windows 95 operating system, the defacto standard in the PC world. We decided to use the Windows Help format for the hypertext document since it can be used on any PC without additional support files. We created the hypertext document using RoboHelp, the most widely used software to create help files in Windows 95.

The simulator was a less clear choice. Although MATLAB is widely used for DSP instruction it is not an open system and it is very slow for adaptive system simulation. One of the top requirements for the electronic book is the seamless integration of the text with the simulator, which in the Windows environment means compatibility with protocols of parameter passing between different programs (as implemented by the Object Linking and Embedding standard - OLE). MATLAB is weak in this respect. Speed is also crucial. If we have to wait 10 minutes to adapt a filter, the student's curiosity will erode and the impact will be lost.

Therefore we decided to use another program called NeuroSolutions [2]. NeuroSolutions is an object-oriented program that has an icon based user interface which is very appealing pedagogically (systems are constructed Lego style). Icons are organized in families, and the mathematics for each family are well documented. It contains a complete set of algorithms necessary for adaptive system instruction. Additionally, NeuroSolutions has the following useful features: it is OLE compatible which allows for tight integration with other windows programs (including the hypertext document); it has many good visualization tools; it allows for the creation of custom components (DLLs) which allows for unique methods of demonstrating concepts; and it includes a macro language which greatly facilitates the creation of the examples (you can execute commands for the student).

One word of caution is needed at this stage, which is the second big lesson learned from our experience. Due to the interface with a simulator, writing an electronic textbook is far different from writing a typical textbook. A deep knowledge of the simulator and the material is necessary to find ways to best illustrate the concepts. Painstaking detail is required to create the examples, find the parameters that work best, and then integrate them with the text. *Due to the additional work and skills required, a team of authors is typically required to write an electronic book.*

5. Classroom Format

From the start we envisaged students using the computer during every lecture. Each student has access to one PC which runs the hypertext and the simulator. A multimedia Pentium 133 MHz with SVGA performs adequately. Most institutions have computer laboratories, but most are not set up for classroom instruction. We were fortunate to have secured an NSF Instrumentation and Laboratory Improvement (ILI) grant that enabled the purchase of PC equipment to set up a classroom specifically designed to meet our needs.

In this new learning environment, the professor's job is very different. The professor's role is more like the conductor of an orchestra: highlighting topics, providing the timing for instruction, and answering questions. A computer projector is very important to let the professor organize and coordinate the student activities. Additionally, an electronic white board (smart board) makes the presentation even more efficient since the professor can control and demonstrate the use of the computer while standing at the front of the class.

One of the aspects to consider is the heavy utilization of facilities. Since the textbook and the examples are on the computer, students must have access to the classroom beyond the lectures. We schedule 2 hour daily periods of access to the classroom. Office hours are also in the classroom. Students who own computers (which will soon be required at our institution) will not require as much laboratory time.

6. Materials

The electronic book is divided into 11 chapters with one appendix that covers the principles of vector spaces and random processes. The chapters are the following:

- I- Linear Regression
- II - Principles of Pattern Recognition
- III - Classification with Multilayer Perceptrons
- IV- Applying Multilayer Perceptrons
- V- Function Approximation
- VI- Hebbian Networks and PCA
- VII- Competitive Networks
- VIII- Principles of Digital Signal Processing
- IX- Adaptive filters
- X- Time Lagged Recurrent Networks
- XI- Recurrent Networks

Chapter I covers the concept of data fitting with the linear model and more importantly the concept of gradient descent learning. Chapter II presents the concepts of statistical pattern recognition. Chapter III studies the multilayer perceptron (MLP) and how to adapt its parameters. Chapter IV delves into the details of how to use MLPs in real world applications (initialization, topology selection, alternate search method, stopping criterion). Chapter V presents the unifying view that regression and classification are special cases of the more general problem of function approximation. This is the way adaptive systems should be ultimately interpreted. Chapter VI covers the principles of estimating correlation with local rules and emphasizes the engineering applications of principal component analysis. Chapter VII presents competitive networks and their applications. Chapter VIII covers the fundamentals of extracting (and quantifying) information in time. Filtering is motivated in vector spaces, but then we cover the principles of time and frequency signal analysis. Chapter IX brings together the concept of regression and time processing in the form of adaptive filters. This chapter contains over 20 practical examples. Chapter X brings together the concept of time processing mixed with nonlinearities. However the adaptation is still static. Many problems can be solved with these intermediate topologies that extend the linear model. Finally, Chapter XI teaches the training of recurrent topologies in time and presents the general case of first order distributed dynamical systems.

This is too much material to be covered in a semester course, but allows for great flexibility in choosing different slants in teaching the material. For EE students we start with linear regression, we move to pattern recognition and MLPs, then to adaptive filters and finally to time lagged recurrent networks (and eventually to recurrent networks). Since EE students have little exposure to statistics we cover chapter Chapter II but skip Chapter VIII. An alternate track more neural network oriented would be Chapter I, III, IV, V, VI,VII,VIII, IX.

7. Teaching Experience

The course was taught on an experimental basis in the Spring ‘97 term as a senior elective. Ten students were enrolled. The course was taught in a PC lab that was not setup for instruction (less than perfect conditions), but the reaction was still very positive. In fact 3 of the students went on to select adaptive systems for their senior project.

One of the highlights of the course is that it is really a blend of theory with design. The integration of a hypertext and simulator allows us to teach problem solving rather than just theory. Three weeks into the semester we were browsing the WEB in search of real data to download to the simulator and perform linear regression. We setup the problem formulation in an adaptive system framework and found excellent examples that highlighted the power of the technique.

The first project was an “open” assignment. The students were requested to find an interesting problem, state a hypothesis, and validate their hypothesis using regression. I received excellent projects from every student. Likewise for the topic of classification. The second project involved a real optical character recognition data set (10 digits).

A final exam was given testing the conceptual understanding of the material. Here we found that the student performance had two clear clusters (the top mark and the average). This was interpreted as meaning that there was a clear set of students that were not only able to master the simulator and the application of the technology to practical problem solving, but were also able to understand the fundamental concepts of the material. The rest mastered the use of adaptive systems (and the simulator) but did not absorb the concepts and theory as well.

We believe that this is one of the dangers of the interactive learning environments. If the student is interested and has a good background, he/she can very rapidly absorb large amounts of information, well beyond what we might expect from an undergraduate student (one of the undergraduate students submitted a paper to a prestigious conference with the experience developed in class). On the other hand, average students may struggle with the extra level of difficulty and never reap the benefits of their effort. Hence the “dynamic range” of student progress should be constantly monitored. The lab periods and the office hours are very important because there the professor (or TA) has an opportunity to coach students in a one-on-one basis and “equalize” progress.

Acknowledgments: This work was partially supported by NSF grant DUE 9751290.

References

1. Haugeland J., “Artificial Intelligence: the very idea”, MIT Press, 1985.

2. NeuroSolutions User’s Manual, NeuroDimension Inc., Gainesville, Florida, 1995.