Teaching DSP First with LabVIEW

Mark A. Yoder, Bruce Black
Mark.A.Yoder@Rose-Hulman.edu, Bruce.Black@Rose-Hulman.edu
812-877-8291, 812-877-8327
812-877-8895, FAX
Electrical and Computer Engineering
Rose-Hulman Institute of Technology

Abstract

The proponents of graphical programming (that is using graphics to program a computer, not programming a computer to do graphics) claim graphical programming is better than text-based programming; however text-based programmers far out number graphics-based programmers. This paper describes the preliminary developments of comparing the use of LabVIEW (a graphical programming language) to MATLAB (a text-based language) in teaching an introductory discrete-time signal processing (DSP) class.

This paper presents the results of using both methods in a junior-level introduction to DSP class. The students who enter this class have had a course in continuous-time signals and systems but no DSP theory background. The class uses the text "Signal Processing First", by McClellan, Schafer, and Yoder, published by Prentice Hall, to introduce discrete-time signal processing. In the past, a series of MATLAB based mini-projects were used in addition to homework to reinforce the DSP concepts. The new version of the class uses the same mini-projects except that they are based on LabVIEW.

Six quarters of concept inventory data have been collected on the MATLAB version of the class. The same inventory was used with three quarters of the LabVIEW version of the class and the results compared. The author does not expect this study to answer the "which is better?" question. Rather it will give experience in assessing what the tradeoffs are in choosing between two very different types of programming languages to teach DSP.

Introduction

When *DSP First*[1] was published in 1998, it introduced several new approaches to teaching discrete-time signal processing. One new approach was teaching DSP early in the curriculum. DSP has traditionally been taught after signals and systems, which is taught after circuits. *DSP First* showed that DSP could be taught **first**, even before circuits[2]. Another new approach was the heavy use of MATLAB[3] in demonstrating DSP concepts in class and in the laboratories[4]. In 2003, its derivative work *Signal Processing First*[5] added four chapters on continuous-time signal processing while continuing the approach of DSP First.

These texts have been used in the junior-level introduction to discrete-time signal processing class at Rose-Hulman Institute of Technology. Strangely, this class is taught after continuous-time signal processing which is taught after circuits. For several quarters, the Discrete Time Signal and Systems Concept Inventory[6-9] has been used for both pre- and post-testing of students in the class.

The combination of a prerequisite class that relies on a computer technology, MATLAB, and several quarters' worth of base-line concept inventory data provides a nice environment for experimenting with the type of programming language used. Thus we decided to see what would happen if we switched from the text-based MATLAB, to the graphics-based LabVIEW[10].

The next section presents the Mini Projects that are used to reinforce the concepts in the class. The section after that shows how programming in LabVIEW differs from programming in MATLAB. The following section presents how the course previously used MATLAB and how that was changed to LabVIEW. Finally, the preliminary results of making the changes are presented.

The Mini Projects

The DSP class as taught at Rose-Hulman does not have a lab with it. Rather a series of Mini Projects use MATLAB to reinforce the concepts of the class. The projects start with a music synthesis project in which students are asked to find sheet music for a given tune (Jingle Bells is popular in the winter quarter, The Little Fugue and Minuet in G by Bach have also been used) and then write a program to synthesize it, producing a *.wav* file that plays the song.

In the second project, the students are given a *.wav* file containing a spoken message that has been masked by some loud additive sinusoids. Their task is to filter out the sinusoids. The weekly projects progress until the students are given a *.wav* file containing a recording of some instrument (an ocarina and violin have been used in the past; in the future a tuba or trombone may be used). Their task is to produce a text file that tells what notes have been played, when, and for how long. This is just the reverse of the first mini project. Table 1 has a complete list of the projects that have been used. The actual projects can be seen at[11].

Table 1: The Mini Projects

mp01:	Music Synthesis
mp02:	Discrete Convolution GUI
mp03:	Tone Removal
mp04:	Tone Removal via poles and Zeros
mp05:	Note Detection
mp06:	Simple Song Detection
mp07:	Swiss Army Knife

We chose to use the same mini projects with LabVIEW as were used with MATLAB, so if there were any changes in the concept inventory results they would most likely be a consequence of the change in language. The students enter the DSP class having used MATLAB in at least one other class. Therefore it was necessary to add two "labs" to introduce them to signal processing using LabVIEW. These labs are listed in Table 2.

Table 2: Additional LabVIEW Labs
Lab 01: Introduction to LabVIEW
Lab 02a: Introduction to Complex Exponentials

From MATLAB to LabVIEW

So how does programming in LabVIEW differ from programming in MATLAB? Many associate LabVIEW with data acquisition and testing. Indeed its origins are there, but it has grown to be able to do much more. LabVIEW has a rather impressive collection of signal processing blocks that make it quite acceptable for teaching signal processing. An added advantage is that block diagrams developed when programming LabVIEW look very much like the block diagrams in textbooks. The mapping from one to the other is rather natural.

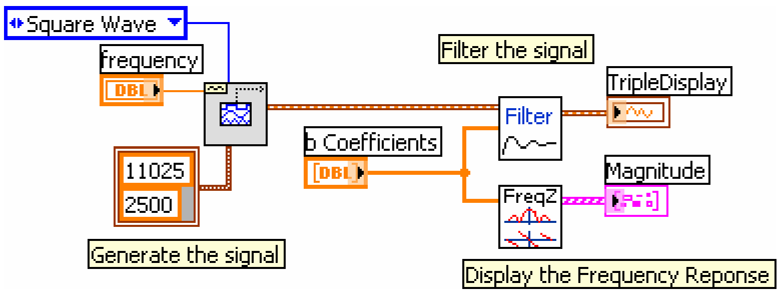
Here's one example that captures the flavor of each program. The goal is to generate a 1kHz square wave signal, sampled at f_s=11025 samples/s. The signal is to be passed through a three-point sliding sum FIR filter and the spectrum of the signal is to be displayed. The frequency response of the FIR filter is also to be displayed.
Here's the MATLAB code

```
fs = 11025;                % Set the sampling rate
tt = 0:1/fs:1/4;           % Generate the time scale
f0 = 1000;                 % Frequency of square wave
xx = square(2*pi*f0*tt);       % Generate the signal

% create the filter
bb = [1 1 1];              % A three point summer
yy = filter(bb, 1, xx);
figure(1)
specgram(yy, [], fs)

figure(2)
ww = -pi:pi/100:pi;
HH = freqz(bb, 1, ww);     % Compute the frequency response
plot(ww, abs(HH))
```

Here is how it is done in LabVIEW:



Which of these programs is faster to program in? A very unscientific study showed that the creation of these two examples by the author each took the same time, 6 minutes. Two of those minutes were spent waiting for the programs to launch.

The Experiment

The Mini Projects were all converted to LabVIEW and two introductory labs were created. In both the winter and spring quarters of the 2005-2006 and 2006-2007 school years, the discrete-time signal processing class was taught exclusively with LabVIEW. In the Winter 2005 quarter there were two sections with 30 and 32 students. Each section was taught by a different professor. A mid-quarter survey was given to the students to see how they were feeling about using LabVIEW. There were some surprising results. Of the 64 students that took the survey, only 3 had learned LabVIEW prior to learning MATLAB. This was no surprise since MATLAB is used in a previous required course. Tables 3-5 shows the responses to some of the questions.

Table 3: Some results of the W*inter 2005-2006* student survey.

	MATLAB	Either/ Neither	LabVIEW
Which language did you learn first?	60		3
Average number of quarters of experience	2.5		1.1
Which language was easier to learn?	11	12	40
Suppose you had some simple task to do, which language would be quicker to do it in?	9	8	47
Which language is better for solving signal processing problems?	14	6	44
Which language do you prefer to use?	7 – strongly 9 – somewhat	7	13 – somewhat 28 – strongly

Table 4: Some results of the S*pring2006* student survey.

	MATLAB	Either/ Neither	LabVIEW
Which language did you learn first?	35		0
Average number of quarters of experience	3		1.05
Which language was easier to learn?	10	8	18
Suppose you had some simple task to do, which language would be quicker to do it in?	15	5	16
Which language is better for solving signal processing problems?	13	8	15
Which language do you prefer to use?	10 – strongly 10 – somewhat	6	5 – somewhat 5 – strongly

Table 5: Some results of the Winter 2006-2007 student survey.

	MATLAB	Either/ Neither	LabVIEW
Which language did you learn first?	10		0
Average number of quarters of experience	3		1.4
Which language was easier to learn?	2	2	6
Suppose you had some simple task to do, which language would be quicker to do it in?	3	2	5
Which language is better for solving signal processing problems?	2	3	5
Which language do you prefer to use?	2 – strongly 3 – somewhat	0	2 – somewhat 3 – strongly

Students generally like to stick with what they learned first and what is commonly used on campus. The surprise was that on the average, students had twice as much experience with MATLAB, but by almost 3 to 1 preferred to use LabVIEW the first quarter; however the 2nd quarter the preference shifted to 2 to 1 in favor of MATLAB and the last quarter it was even. The written comments from the students give some insight. Many like LabVIEW's visual/graphical interface. They say it is easier to learn and more understandable. Some say they are more able to focus on concepts rather than syntax. One student said "LabVIEW often gives a better 'birds eye view' of what is happening…".

There seems to be another camp of students who could be labeled as traditional programmers. When you know what you are doing, it's much faster to type a program than to select icons from menus and point and click to connect them. This group's view could be summed up by the student that shuddered "Graphical programming languages scare me".

There appears to be a third group out there who do not necessarily like LabVIEW, but also dislike MATLAB. A student noted "I was thrown into MATLAB and sunk, …". Another said "I strongly dislike MATLAB".

The concept inventory post-test was given. The normalized gain [defined as 100% *(post-pre)/(perfect score-pre)] [9] from the pre-test to the post-test for this class is shown in Table 6. Table 6 also shows the gain for previous offerings of the same course which used roughly the same mini projects implemented with MATLAB.

Table 6: Normalized Gain from Discrete-Time Signals and System Concept Inventory

	Win 02-03	Spr 02-03	Win 03-04	Spr 03-04	Win 04-05	Spr 04-05
Language	MATLAB	MATLAB	MATLAB	MATLAB	MATLAB	MATLAB
Number of Students	38	40	56	29	61	42
Normalized Gain	37	36	32	29	41	41

	Win 05-06	**Spr 05-06**	**Win 06-07**
Language	**LABVIEW**	**LABVIEW**	**LABVIEW**
Number of Students	**64**	**34**	**43**
Normalized Gain	**46**	**37**	**42**

Although the LabVIEW sections in the first quarter showed the highest gain so far, it would be risky to draw many conclusions. The LabVIEW experiment did not seem to hurt the students, and maybe it even helped.

Conclusions and Future Work

It has been an interesting experiment switching from one language to a completely different one for teaching DSP. The results of our survey showed that many of our students actually preferred using LabVIEW. The real gain may be in doing some sort of computer-based exercise rather than the choice of languages used.

On the anecdotal side, both instructors agree that it is much easier to grade a simple program written in a graphical language than in a text-based one. It is easy to tell at a glance whether the student did the program correctly.

The question of which programming paradigm is best for what purpose may never be answered. However, the graphic approach is a viable paradigm. If our students are to be exposed to a breadth of ideas, then they should be exposed to a variety of way to program.

An interesting future project would be to adapt the labs and mini-projects to use Simulink[3] a graphical interface for MATLAB. This would allow one to use a graphical interface but not be as removed from MATLAB.

Bibliography

[1] Jim McClellan, Ron Schafer and Mark A. Yoder, “DSP First: A Multimedia Approach”, Prentice-Hall, January 1998, in 8th printing.

[2] Mark A. Yoder, James H. McClellan, Ronald W. Schafer, “Crystal Radios or DSP First?,” Frontiers in Education Conference, Tempe, Arizona, November 4-7,1998, Session F2D.

[3] The Mathworks, http://www.mathworks.com/.

[4] J.B. Schodorf, Mark A. Yoder, J. H. McClellan, R.W. Schafer, “Using Multimedia to Teach the Theory of Digital Multimedia Signals”, *IEEE Trans. On Education, special issue on Applications of Technology in Education*, Vol. 39, No. 3, August 1996.

[5] Jim McClellan, Ron Schafer and Mark A. Yoder, “Signal Processing First” Prentice-Hall, March 2003.

[6] http://www.foundationcoalition.org/home/keycomponents/concept/SSCI.html

[7] Evans, D.L., and Hestenes, David, 2001. "The Concept of the Concept Inventory Assessment Instrument," *Proceedings, 2001 Frontiers in Education Conference*, Reno, Nevada, 10–13 October 2001.

[8] Wage, Kathleen E., and Buck, John R., 2001. "Development of the Signals and Systems Concept Inventory (SSCI) Assessment Instrument," *Proceedings, 2001 Frontiers in Education Conference*, Reno, Nevada, 10–13 October 2001.

[9] Wage, K.E., Buck, J.R.,Wright, C.H.G., and Welch, T.B., “The Signals and Systems Concept Inventory,” *IEEE Transactions on Education*, Aug. 2005,Vol. 48, No. 3, pp 448- 461.

[10] LabVIEW, www.ni.com.

[11] The Mini Projects www.Rose-Hulman.edu/DSPFirst. Click on the ‘Labs’ link on the upper left. Please contact the author for a username and password.

[12] Signal Processing with LabVIEW, http://zone.ni.com/devzone/conceptd.nsf/webmain/ef0c3c8a145ef2df8625706e007f281a.

Biography

Mark A. Yoder - Professor of Electrical and Computer Engineering at Rose- Hulman Institute of Technology in Terre Haute, Indiana. Co-authored the books **DSP First: A Multimedia Approach** and **Signal Processing First** with Jim McClellan and Ron Schafer which were published by Prentice Hall in 1998 and 2003. And he has also co-authored **Engineering Our Digital Future** with Geoffrey Orsak, et. al. published 2004 by Prentice Hall. KA9ELG.

Mark's biography isn't complete without some mention of his family. His wife Sarah has her Ph.D. in Electrical Engineering from Purdue University, and they have ten wonderful children ages 23, 23, 20, 18, 16, 14, 12, 10, 7, and 4. Three boys and seven girls.

Bruce A. Black completed his Ph.D. at the University of California at Berkeley, in electrical engineering. Since 1983 he has been on the faculty of the Department of Electrical and Computer Engineering at Rose-Hulman Institute of Technology in Terre Haute, Indiana, where he is also advisor to Tau Beta Pi and to the Amateur Radio club (W9NAA). His interests are in communications, wireless systems, and signal processing.