IF ALL I HAD WERE A HAMMER…
On the Use of LabVIEW in Teaching Differential Equations

Murat Tanyel
Dordt College

Abstract

The standard computer tool that is used in teaching differential equations is Derive ™, Maple™, Mathematica® or even MATLAB®. However, when I was asked to teach *Differential Equations* (*Diff. Eq.*), my immediate choice for demonstrations did not employ one of these standard packages, but LabVIEW, which I use in my *Digital Signal Processing* and *Communications Systems* courses[1-3]. The first part of this paper will provide an explanation as to why I chose this particular package for my demonstrations in *Diff. Eq.* with specific reference to the aesthetic aspect of the choice[4]. The second part of the paper will describe the toolkit I developed in LabVIEW for the *Diff. Eq.* class. Examples of in-class demonstrations will be provided including a demonstration of "statistical exponential decay" with discussion on the most appropriate statistical analysis of the data generated through this demonstration.

I. Introduction

The title of this paper is taken from the saying, "If all I had were a hammer, everything would look like nails to me." The saying suggests that one's assessment of a problem is influenced by the tools one has available. I have chosen this title because of my almost instinctive reaction to the challenge of providing a numerical solver for the differential equations course I taught in the Fall of 2002. The preface of the textbook[5] that I chose mentions MATLAB, Maple or Mathematica as likely candidates. The proper choice would be a symbolic algebra package. I resort to Maple for my own symbolic manipulation needs. This choice is influenced by my prior experience on a different campus[6] but the standard package on this campus is Derive. The textbook, on the other hand, leaves the choice of the numerical solver to the reader and informs that it would be adequate if it can i) plot direction fields, ii) calculate numerical solutions of differential equations and iii) plot these solutions. As I was contemplating the use of MATLAB as the compromise solution, my personal preference of LabVIEW dared me to use this package for my first numerical example. That started a trend which became hard to break.

This paper will report on the use of this unusual tool in the Diff. Eq. class. Section II will describe LabVIEW briefly and seek to provide an explanation for my preference for LabVIEW from an engineering aesthetic point of view. Section III will survey its mathematical functions.

Section IV will introduce the toolkit I developed for the Fall '02 offering of Diff. Eq. with some examples. Section V will offer a discussion on this first experience.

II. "Beauty Is in the Eye of the Beholder"

LabVIEW is an acronym from the words "Laboratory Virtual Instrumentation Engineering Workbench." This software package is based on the concept of data flow programming and is particularly suited to test and measurement applications. The three important components of such applications are data acquisition, data analysis and data visualization. LabVIEW offers an environment which covers these vital components. One basic components of a LabVIEW *virtual instrument* (*VI* – LabVIEW's term for a program) is the *front panel*, which is a window for the user interface where exchange of data with the end user of the program occurs. The other is the *diagram*, which is another window that describes how inputs from the front panel are to be processed to achieve the outputs using a graphical language similar to block diagrams. LabVIEW front panels offer the user means of input in the shapes of knobs, sliders, switches, and other devices commonly found on laboratory instrumentation as well as means of output in the forms of gauges, meters, oscilloscope-screen type graphs, light emitting diodes and similar output tools, all contributing to a user interface that looks like a typical instrument panel.

In a recent paper[4], Adams and I explore the role of aesthetics in engineering design, particularly in the choice of computer aided tools for signal processing, which is the area I prefer to teach regularly. In that paper, we reflect on the normative principles on the responsible employment of technology and espouse the following principles, after Monsma, et al[7]: *cultural appropriateness*, *openness and communication*, *stewardship*, *delightful harmony*, *justice*, *caring* and *dependability*. Of these principles, *delightful harmony* is the norm that deals with aesthetics. *Delightful harmony* implies that a) the artifact must be effective, or competent; b) it should be "pleasing and satisfying to use;" c) it must promote harmonious relationships[7]. We then establish a link between aesthetics and "playfulness" which incorporates harmony and beauty as distinct qualities after Seerveld, who coins the term "allusivity" to describe it[8]. An allusive object is an object that is suggestive of something else in a subtle manner, with nuance. Seerveld suggests that "allusivity is the central core of aesthetic meaning." [8] We can therefore equate aesthetic quality with the quality of subtle suggestiveness or nuance. Adams incorporates this notion in his definition[9]: "Technological allusivity in engineering design is achieved when the design successfully suggests a (delightfully) harmonious interaction, at the human-technical interface, whereby the product dissolves into an extension of the user." With this background, the paper[4] then demonstrates how the LabVIEW front panel provides a delightful harmony by making allusions to real-life instruments and how its diagram does likewise in processing data in areas in which the processes are best explained by block diagrams, such as digital signal processing, communication systems or control systems.

Another factor to be considered is the programmer's familiarity with the programming environment. Mastery of a programming environment will enhance the "harmonious interaction"

between the programmer and the computer. It is no surprise, then, given a history of immersion into LabVIEW in digital signal processing and communication systems courses[1-3], programming in LabVIEW had "dissolved into an extension of me" and my first reaction was to employ LabVIEW in Diff. Eq., as well.

In short, my choice of software, which may appear unusual for Diff. Eq., derived from the aesthetic aspect of the environment offered by the software. Most of the students in the class were engineering students, trained in the use of laboratory equipment in their engineering and physics classes. They were in a position to appreciate the particular user interface that LabVIEW offered and were thus able to utilize these programs almost instinctively. Preparing demonstrations for the students was also attractive to me because the task of programming alluded to the design of an instrument, appealing to my engineering identity, and also because my familiarity with the software package made the programming effortless.

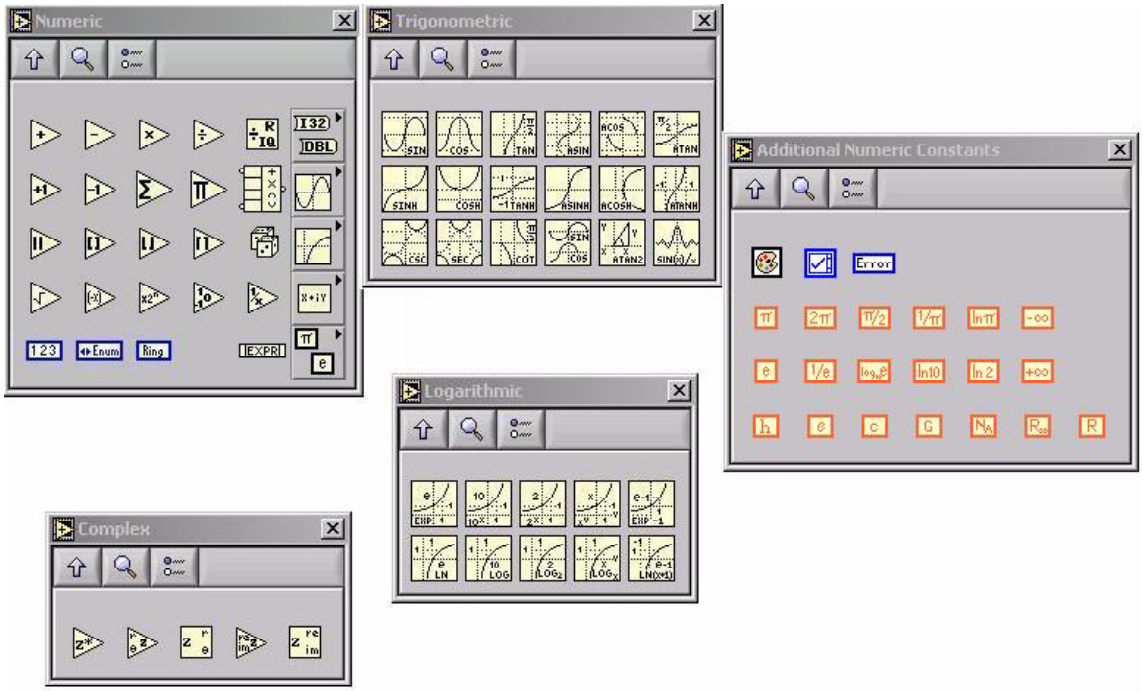

Figure 1: The numeric palette and its sub-palettes, displaying the variety of rudimentary mathematical functions available.

III. LabVIEW's "Mathematical" Functions

LabVIEW programming is accomplished by choosing the appropriate blocks, represented by icons, from the "Tools" palette within the diagram and connecting the appropriate inputs and outputs to these blocks. The most rudimentary mathematical functions in the tools palette are grouped under the category "Numerical". Fig. 1 displays the Numerical palette and its sub-

palettes (Trigonometric, Logarithmic, Complex, Additional Numeric Constants).

It is not my aim to list all the different functions available in the Numeric palette, but Fig. 1 should suffice to convince the reader of the abundance of predefined mathematical functions in LabVIEW. For more complicated algorithms, one can use the "Mathematics" palette, which includes the following subpalettes: Formula (containing blocks that evaluate formulas), 1D and 2D Evaluation (containing blocks that parse and evaluate single variable and 2-variable functions), Calculus (with blocks for numerical integration, differentiation and a subpalette of ODE solvers), Probability and Statistics, Curve Fitting, Linear Algebra, Array Operations, Optimization, Zeroes and Numeric Functions (containing blocks for the Bessel function, Gamma function, Sine integral and others). This kind of computing power certainly enables LabVIEW to deliver the expectations of the textbook listed in the previous section.

IV. The Toolkit

In this section, I am going to introduce highlights from the demonstrations I used in class, or VIs I made available to students for the exercises which called for numerical methods.

Slope Fields: For a differential equation

$$y' = f(t, y) \quad (1)$$

defined for (t, y) in a rectangle R, the slope field is the collection of small line segments with slope $f(t, y)$ attached to every point (t, y) of the rectangle R. Maple, Mathematica, MATLAB and even some hand-held calculators can easily produce these fields5. LabVIEW, on the other hand, does not offer a block for slope fields. The first VI that I developed, `Slopefield.vi`, accomplishes textbook's expectation (i). It draws slope fields (or direction fields) for 1st order differential equations.

1st Order ODE Numerical Solver: The second challenge was to develop a numerical solver that would handle any problem in a typical Diff. Eq. book. ODE solvers in LabVIEW assume the initial value specified occurs at the initial point of the interval of the independent variable. They may proceed in the decreasing (by specifying a negative increment) direction of the independent variable and thus accept initial values at the end of the interval, but in either case, the value is assumed to be at one end of the interval. However, it is not uncommon to find exercises in typical textbooks which call for a sketch of the solution to some diff. eq. in a particular range of the independent variable with the initial condition specified at an arbitrary point in the range. In order to handle such problems, I programmed a generalized ODE solver which can handle the initial condition anywhere in the range of the independent variable.

Direction Field with Solution: The third VI combines the first two VIs and superimposes the

numerical solution to the slope field. Fig. 2 is a depiction of this VI, showing the slope field on a 12 × 12 matrix of the differential equation $y' = y$ with $y(0) = 1$ in a rectangle ($-2 \le t \le 2$ and $4 \le y \le 8$) with the numerical solution superimposed.

Guessing Game for Initial Condition: Fig. 3 shows another VI that makes use of the ODE solver in LabVIEW. This VI is meant as an exercise to demonstrate the dependence of solutions on initial conditions. The diff. eq. in question is

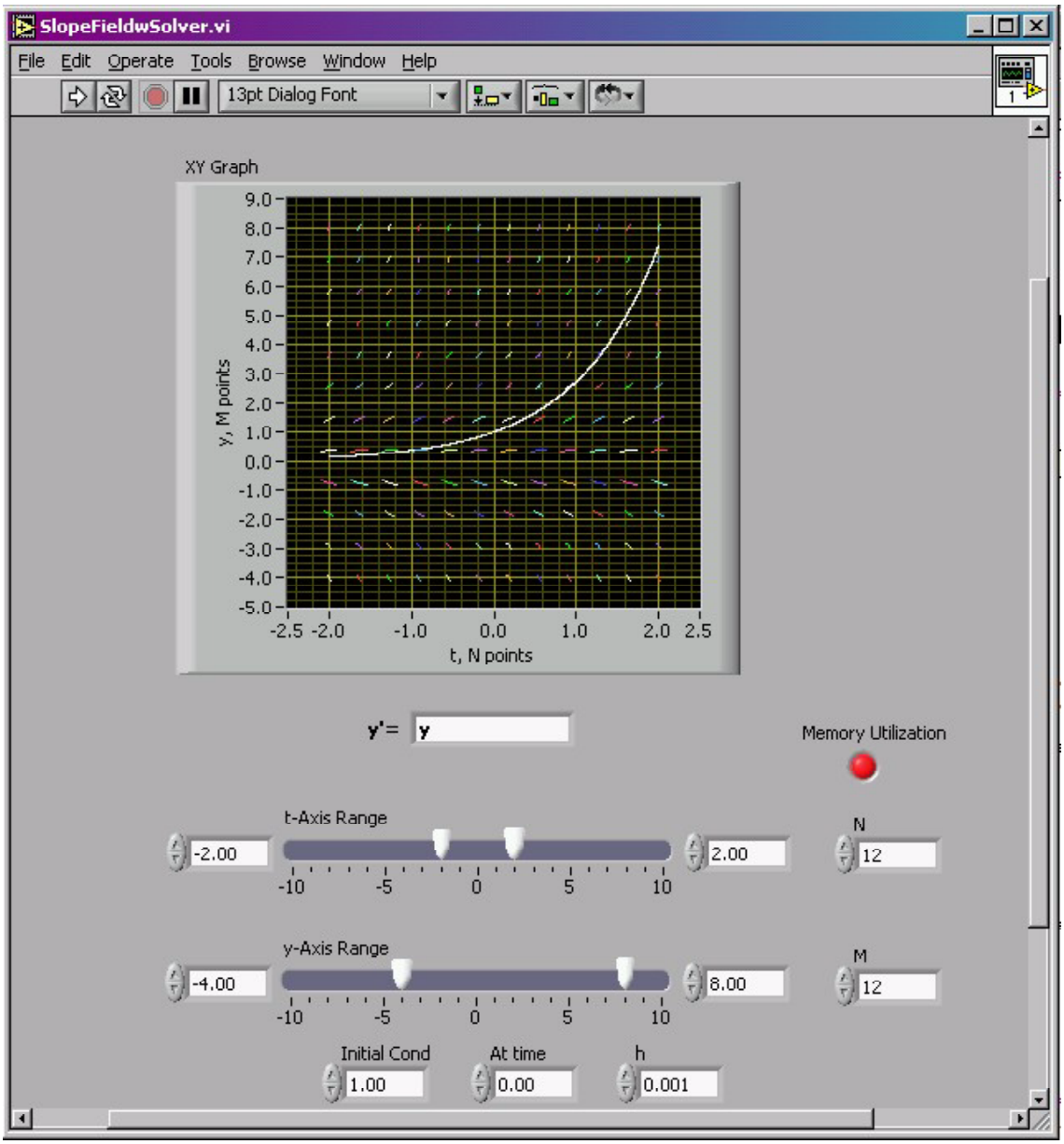

Figure 2: The slope field superimposed with the numerical solution for a 1st order diff. eq.

$$y' = y - t \quad (2)$$

The user is asked to guess the initial condition (at $t_0 = 0$) such that the final value of the curve will converge to (6, 0). The VI displays the target point (the white circle in the graph of Fig. 3), the curve resulting from the initial guess as well as the four previous initial guesses with the curves resulting from those guesses. Fig 3 displays that an initial guess of 0.9825 came close to the target, with the destination point (6, -0.03887). The previous guesses were 0.95, 0.975, 0.98, 0.985 with the corresponding colors of resulting curves.

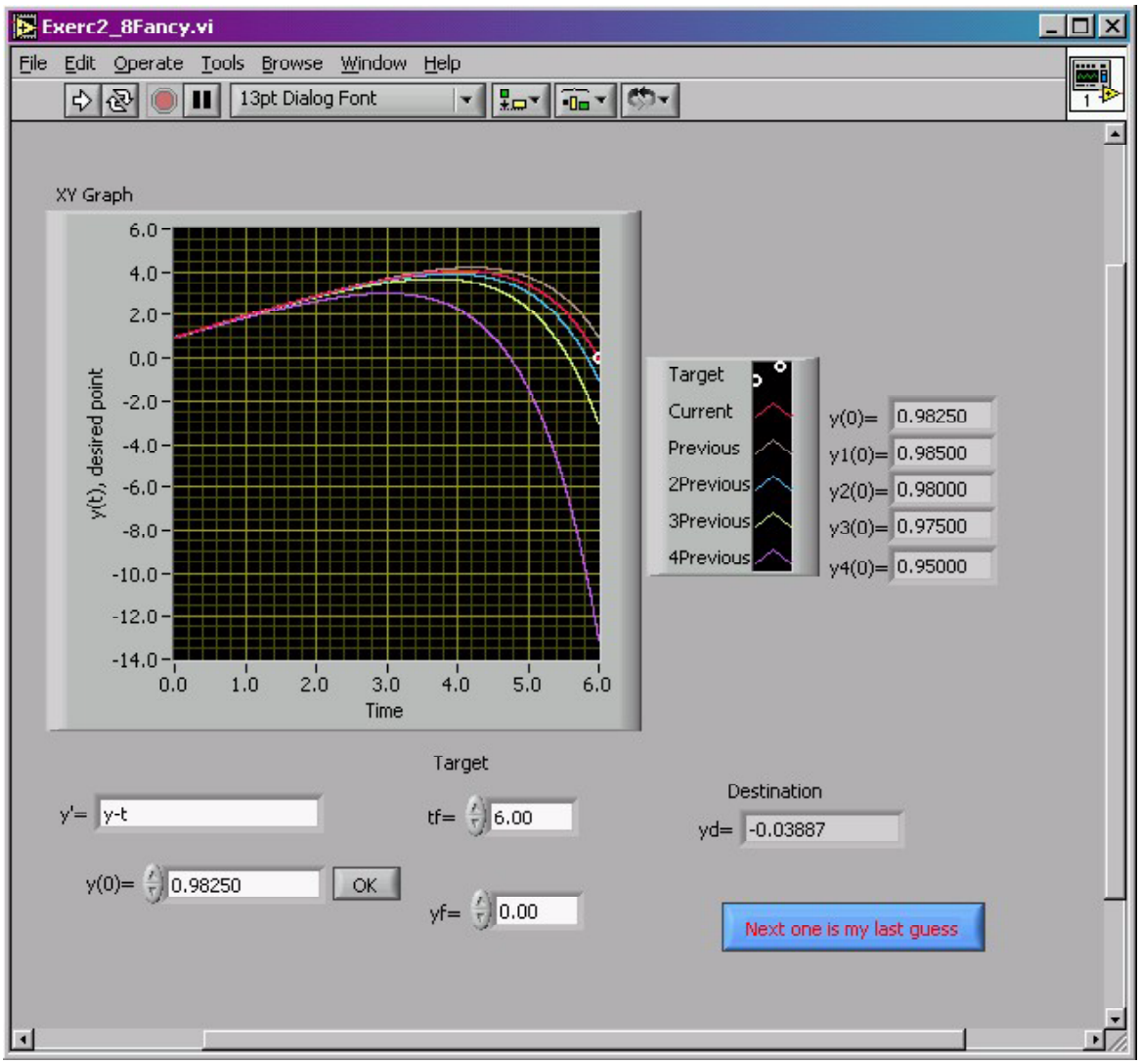

Figure 3: An exercise to demonstrate the dependence of solutions on initial conditions.

Damped Unforced Spring System: In the introduction to second order equations, the textbook uses the example of a mass attached to a spring with friction, suspended from the ceiling and with non-zero initial conditions. The differential equation describing the system in the absence of a forcing function is:

$$my'' + \mathrm{m}y' + ky = 0 \quad (3)$$

where m is the mass, m is the coefficient of friction and k is the spring constant and y is the vertical displacement of the mass. This example is brought up to illustrate the concept of overdamped, underdamped and critically damped systems. The same example is also used to introduce various ways of viewing the system: a graph of displacement as a function of time (y vs. t), velocity as a function of time [v (= y') vs. t], the phase-plane plot (of v vs. y) and a 3D plot of v vs. y with respect to t. The exercises at the end of the section ask for these plots under varying parameters and initial conditions. Fig. 4 is the front panel of a VI where students can change parameters and observe the behavior of the system. The plots suggest that the parameters chosen represent an underdamped system.

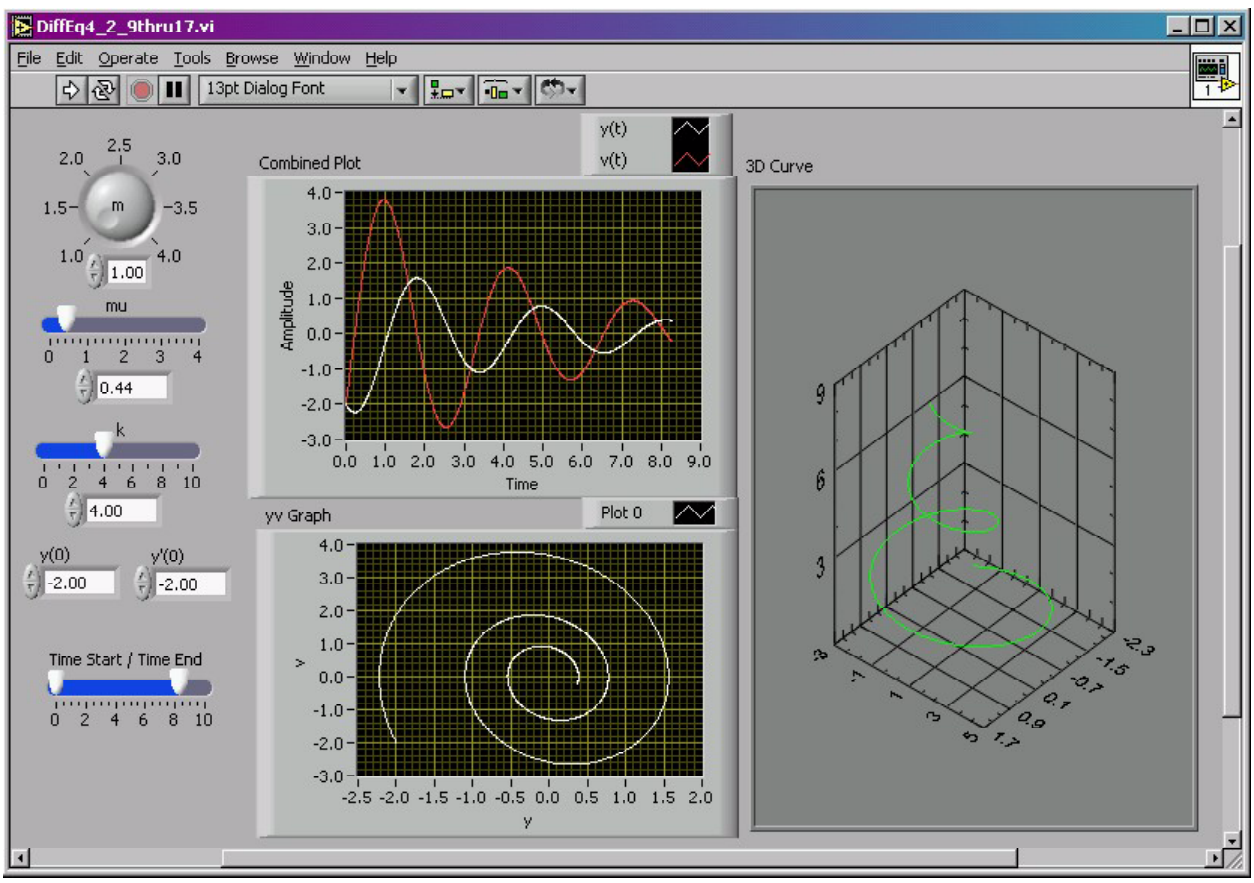

Figure 4: Various ways of studying a 2nd order system.

Statistical Exponential Decay: Some processes that are probabilistic in nature exhibit behavior that can be modeled by exponential decay10. At a faculty development workshop at the University of Prince Edward Island, I took part in an "experiment" which simulated population decay under scarcity of food. We were all given dice and we all got up and rolled our dice to see who would "starve". Those that rolled a certain number sat down, pretending "death". We counted those still standing at the end of each roll, and fitted an exponential curve to the data. The parameters of the exponential curve fit should estimate the number of people that participated and the number of faces of the dice used. We made the remark that one run was not sufficient and performing a number of "experiments" and averaging the data would yield better estimates. This "experiment" was a good candidate to be included when we discussed population (growth or decline), in my diff. eq. class. The mathematical model for the process can be developed as follows:

$$\frac{dp}{dt} = bp - dp \qquad (4)$$

where p stands for population as a function of time, b is the birth rate and d is the death rate. In other words, the change in the classroom's population is equal to the "birth" rate times the current population minus the "death" rate times the current population. Since we do not introduce any new students while the experiment is going on, the birth rate, $b = 0$. The death rate, on the other hand, is statistically 1/6 for six-sided dice. So eq. 4 becomes:

$$\frac{dp}{dt} = -\frac{1}{6}p \qquad (5).$$

This is a separable equation, which may be rearranged as:

$$\frac{dp}{p} = -\frac{1}{6}dt \qquad (6).$$

Integrating both sides, we get

$$\ln|p| = -\frac{1}{6}t + C \qquad (7)$$

or,

$$|p(t)| = e^{-\frac{1}{6}t + C} \qquad (8).$$

Finally,

$$p(t) = Ae^{-\frac{1}{6}t} \quad (9)$$

where $A = \pm e^{C}$. At time 0, the population is the class size, so an estimate of A should reveal the class size and an estimate of the time constant should yield the number of faces (= 6 in this case) of the dice.

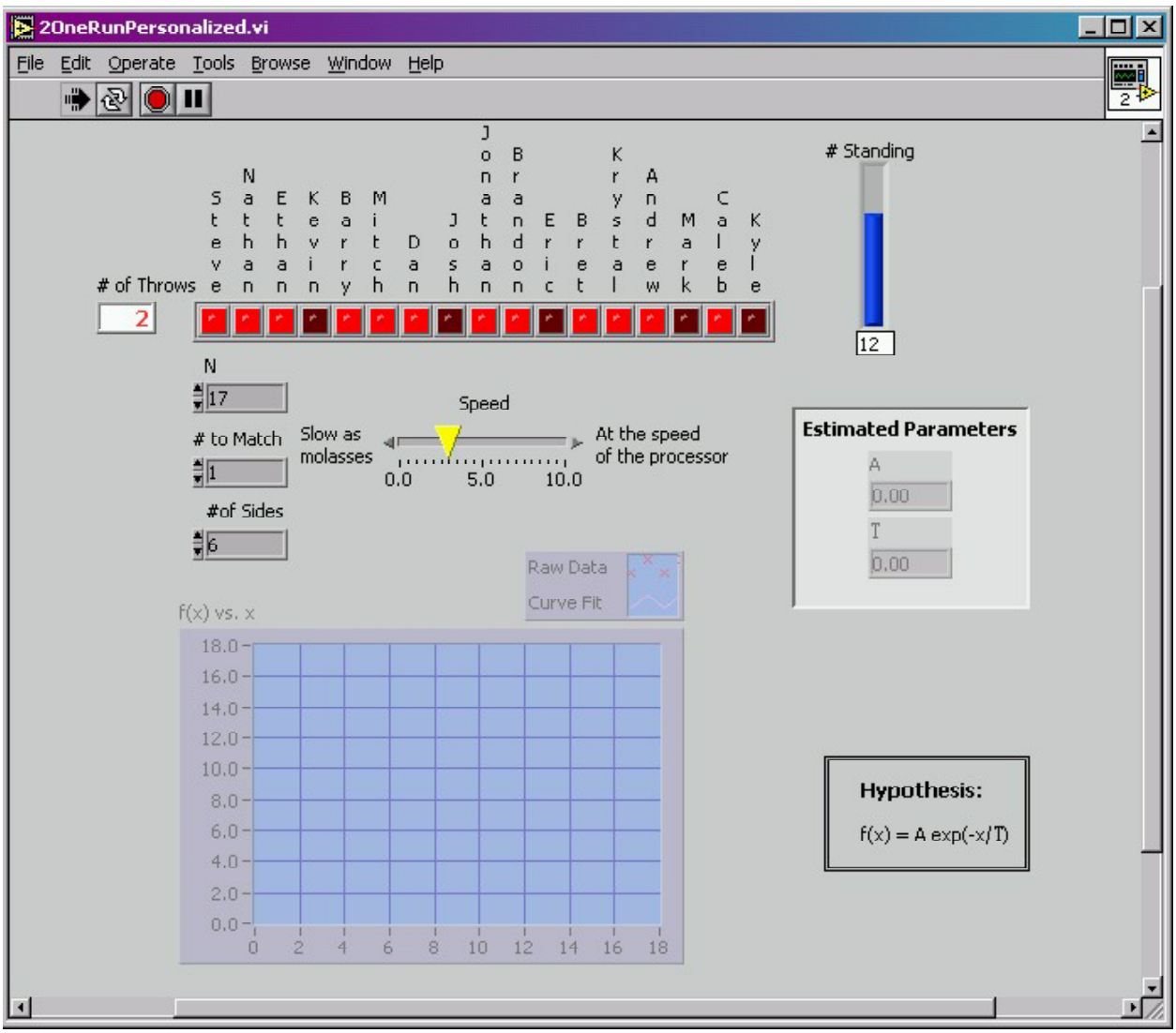

Figure 5: The statistical exponential decay experiment in progress.

The "experiment" was a good candidate, but it was too time consuming. I decided to simulate the experiment on the computer (once slowly, to let the students experience it) and then to repeat the simulation many times almost instantaneously to achieve the better estimates. Fig. 5 is a snapshot of the front panel of the VI that I projected on the screen for the statistical exponential decay experiment. Each student in the class is represented by a LED on the panel. If the LED is on, the student is standing. If it is off, the student has "starved". I gave the students the opportunity to act it out as the simulation ran on the projector screen and soon the simulation turned into a game with the last ones standing receiving enthusiastic cheers.

At the end of the experiment, the VI performs the curve fitting and the parameter estimation (Fig. 6). In Fig. 6, we notice that while the estimate for the class size is pretty accurate (17.14 as opposed to the class size of 17), the dice have too many sides (7.65)!

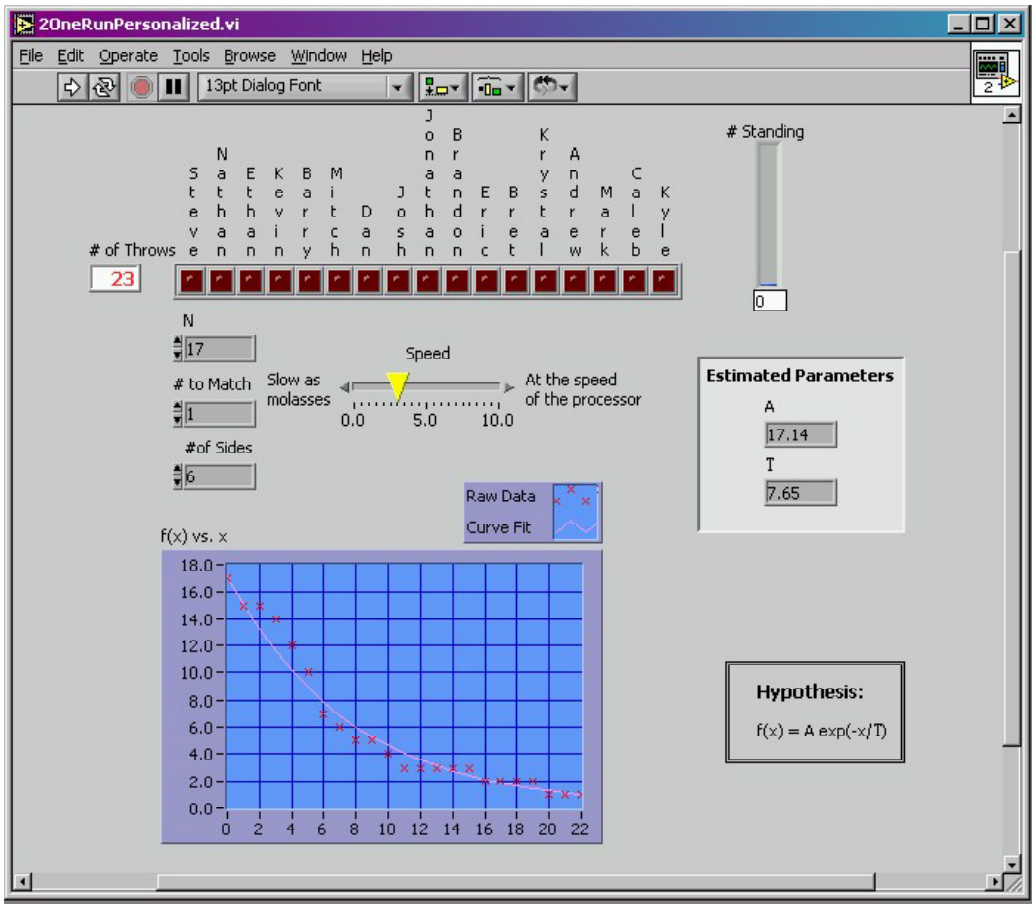

Figure 6: The result of one experiment.

Running this VI even "at the speed of the processor" will still take a long time to complete many experiments. So, I wrote another VI that is free of all visual aids, which repeats the experiment

without any exchange of information and then reports on the aggregate results.

Figure 7 displays the result of 1000 experiments. In this case, the number of students is 32. The question that arises is how to average the data. Does one average all the data from the 1000 experiments and then estimate the parameters, or does one estimate the parameters at each run and then average them? The VI of Fig. 7 performs both of these estimations. On that panel, A and T are the estimates from the averaged data while A' and T' are the averaged parameters from the estimates of individual runs. The results in Fig. 7 suggest that the estimate of $p(0)$ is better when one averages the data first. For an initial size of 32, $A = 32.02$ (.06 % error) and $A' = 29.11$ (9 % error). Estimate for the time constant is slightly more accurate if calculated individually and then averaged. Fig. 7 reveals that $T = 5.46$ (9 % off) and $T' = 6.37$ (6.2 % off).

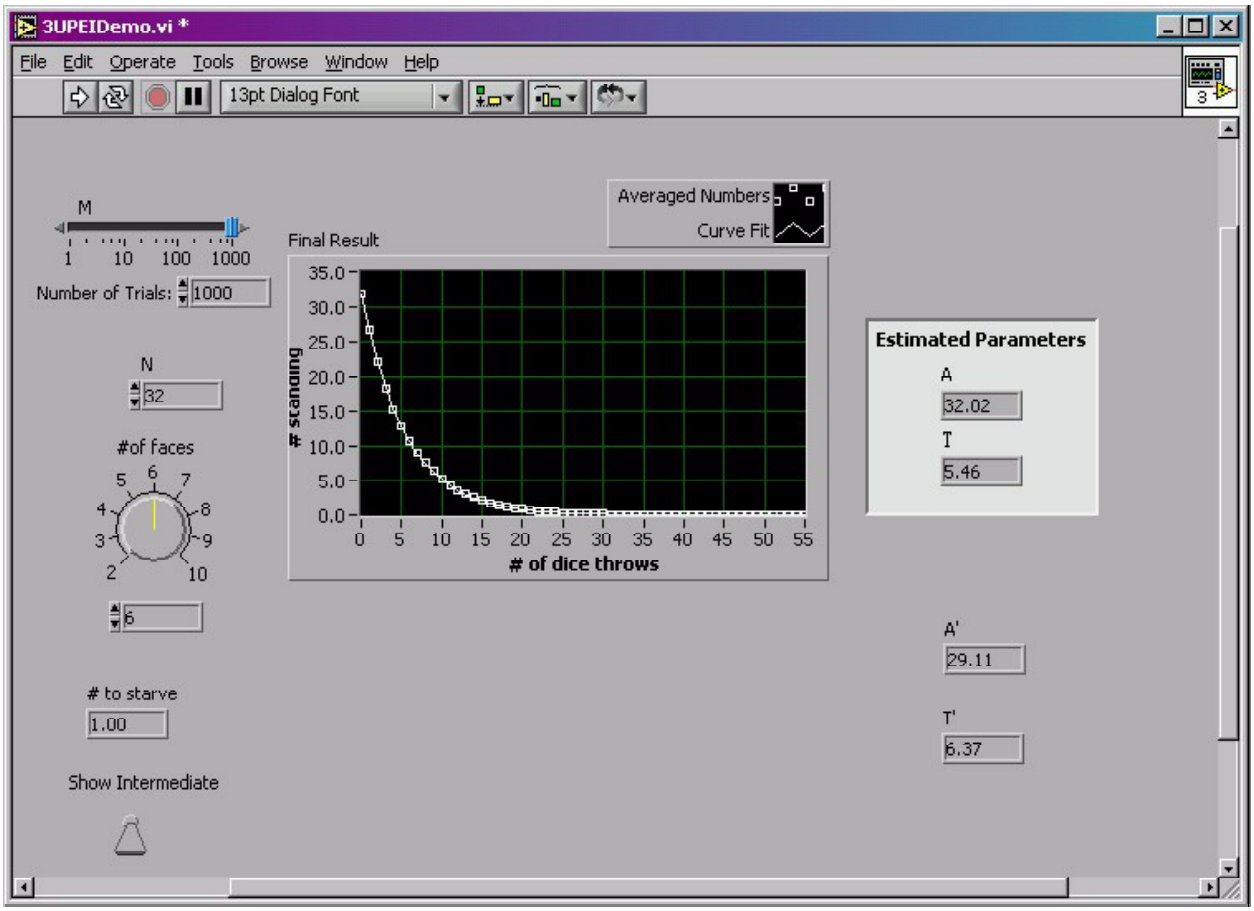

Figure 7: The VI that performs multiple experiments.

V. Discussion

Students in MATH 204 were not familiar with the LabVIEW programming environment or even

LabVIEW programs (VIs). However, they were able to use the VIs I provided without any difficulty. Since the front panel simulates an instrument, carefully labeled knobs and switches render the program much more user friendly than routines offered in more conventional environments. It is easier to guess the parameters of a program when they are labeled than to remember which variable controls which parameter in a list of function arguments, as is the case with many text-based programming environments.

I found that routines provided by LabVIEW were sufficient to meet any computational requirement that the textbook brought. The VI that displays slope fields is very slow and memory intensive because every little line segment is a separate graph. There must be a more efficient way of drawing these lines. Other than that, the utilities that LabVIEW provided were on par with any textual computational environment. I had fun developing these tools and I would improve and add on to these tools if I had to teach differential equations again.

The students, who were the end-users of these programs, were able to run these routines without any official introduction to LabVIEW. The execution of some routines (such as the exponential decay and the guessing game for initial conditions) created enough excitement to encourage the utilization of the entire toolkit. I enjoyed programming these routines in LabVIEW and found that the programming environment was more than adequate to meet my needs. In other words, the artifact (the programming environment) was effective *and* competent; it was "pleasing and satisfying to use;" and it promoted harmonious relationships between the computer and the students, all of which suggests that this alternative environment would score highly on the aesthetic scale among the normative principles on the responsible employment of technology as mentioned in section II.

ACKNOWLEDGEMENTS: The LabVIEW software used in this course was purchased through a grant from Johnson Controls in Holland, MI. I would like to thank Johnson Controls for the upgrade of the electronics lab, which also benefited the course mentioned in this paper.

Bibliography

1. Viss, M. and Tanyel, M. "From Block Diagrams to Graphical Programs in DSP," *2001 ASEE Annual Conference & Exposition Proceedings*, Albuquerque, NM, June 24-27 2001.

2. Tanyel, M., "Enhancing the DSP Toolkit of LabVIEW", *2002 ASEE Annual Conference and Exposition Proceedings*, Montréal, QC, June 16-19 2002.

3. Tanyel, M., Nguru, K., "Preparation of a Virtual Toolkit for Communication Systems", *2002 ASEE Annual Conference and Exposition Proceedings*, Montréal, QC, June 16-19 2002.

4. Tanyel, M., Adams, C., "On the Aesthetics of Computer Aided Tools for Signal Processing", *Proceedings of 64th Annual ASEE North Midwest Section Meeting*, Madison, WI, Oct. 2002.

5. Polking, J., Boggess, A., Arnold, D., *Differential Equations*, Upper Saddle River, NJ: Prentice Hall (2001).

6. Carr, R., Tanyel, M., *Engineering Explorations with Maple*, Philadelphia, PA: Harcourt Brace College Publishing / Harcourt Brace Custom Publishing (1994)

7. Monsma, S. V., Leegwater, A., Christians, C., Schuurman, E., Dykema, E. R. and Van Poolen, L., *Responsible Technology*, Grand Rapids, MI: W. B. Eerdmans Publishing Company (1986).

8. Seerveld, C., *Rainbows for the Fallen World: Aesthetic Life and Artistic Task*, Toronto, ON: Toronto Tuppence Press (1980).

9. Adams, C. C., *An Analysis and Solution to the Two-Cultures Problem in Undergraduate Engineering Education*, Doctoral Dissertation, The University of Iowa, Iowa City, IA, July 1996.

10. Wagner, B., Unpublished notes on Active Student Participation in Chemistry Classes, 2002 UPEI Summer Institute on Faculty Development, University of Prince Edward Island, Charlottetown, Prince Edward Island, Canada, 07/29-08/02/2002.

MURAT TANYEL
Murat Tanyel is a professor of engineering at Dordt College. He teaches upper level electrical engineering courses. Prior to teaching at Dordt College, Dr. Tanyel taught at Drexel University where he worked for the *Enhanced Educational Experience for Engineering Students* (*E4*) project, setting up and teaching laboratory and hands-on computer experiments for engineering freshmen and sophomores. For one semester, he was also a visiting professor at the United Arab Emirates University in Al-Ain, UAE where he helped set up an innovative introductory engineering curriculum. Dr. Tanyel received his B. S. degree in electrical engineering from Boğaziçi University, Istanbul, Turkey in 1981, his M. S. degree in electrical engineering from Bucknell University, Lewisburg, PA in 1985 and his Ph. D. in biomedical engineering from Drexel University, Philadelphia, PA in 1990.