© American Association of Physics Teachers 1999, published in *The American Journal of Physics* **61** (1993) 222-232.

Student Programming in the Introductory Physics Course: M.U.P.P.E.T.

Edward F. Redish
Department of Physics and Astronomy
University of Maryland
College Park MD 20742-4111

and

Jack M. Wilson
Anderson Center for Innovation in Undergraduate Education
and Department of Physics
Rensselaer Polytechnic Institute
Troy, NY 12180-3590

May 31, 2001

Abstract

Since 1983, the Maryland University Project in Physics and Educational Technology (M.U.P.P.E.T.) has been investigating the implication of including student programming in an introductory physics course for physics majors. Many significant changes can result. One can rearrange some content to be more physically appropriate, include more realistic problems, and introduce some contemporary topics. We also find that one can begin training the student in professional research-related skills at an earlier stage than is traditional. We learned that the inclusion of carefully considered computer content requires an increased emphasis on qualitative and analytic thinking.

I. Introduction

Since 1983, in an effort we refer to as the Maryland Project in Physics and Educational Technology[1] (M.U.P.P.E.T.), the authors and their colleagues at the University of Maryland have been studying what the impact is of introducing beginning students to programming at the start of the traditional calculus-based introductory physics course.

The computer is more than simply a powerful calculator that allows students to multiply and add more quickly. The computer adds orders of magnitudes to the individual's computational abilities. As we know well in physics, when scales change by orders of magnitude, we have to look carefully for qualitatively new phenomena.

The question we address in this paper is the following:

- *What is the implication of the computer for teaching physics majors at the introductory level?*

To answer this we must considerthe answer to two related questions:

- *What is it we want our students to learn?*
- *In what ways is the current introductory physics course inadequate?*

We assume that the broad general goal of the introductory physics course is to begin to prepare students to be professional physicists as well as to introduce them to the basic physics content. We find that having the computer as part of the course affects significantly both the skills we can begin to train and the specific content.

A. Problems with the traditional physics curriculum

The physics curriculum as presently taught was developed over thirty years ago.[2] Although that change was a significant advance at the time, the curriculum has failed to develop since then.

This might not be a problem if physics were a static field. It is not. In the past thirty years we have seen an explosion of new understanding and power in a variety of subfields of physics ranging in scale from the substructure of the proton to the clustering of galaxies. There have even been major breakthroughs in fields long thought to be understood. Current developments in Newtonian mechanics are evolving into a theory of non-linear systems and chaotic behavior that may produce profound changes in the way we think about physics.[3]

It is not only the content of physics that has changed. The computer has changed the way the physics professional carries out his or her job. The introductory student rarely gets any hint of modern developments or of the excitement of doing physics and learning new things about the world that no one has known before. The absence of active research and exciting new areas in the introductory physics course is a serious distortion of the profession.

A second problem is that, although standard instruction introduces students to the basic content of physics, it provides almost no activities that illustrate how research is done -- the kind of work that professional physicists do day-to-day. This problem has been noted in more general circumstances by education specialists who recommend a professional education that is more like an apprenticeship than current college education.[4] Although many physics departments have programs to "get undergraduates into the research lab", in most cases their goal is to "expose" students to "real" research. That is, the students become part of an existing research project. Unfortunately, since most of them don't know enough physics to actually participate in designing the project or making research decisions, their activities tend to be menial.

To decide what activities an undergraduate needs to do to begin to understand how to become a professional scientist ,we have to analyze the differences between the activities carried out by professional research physicists and introductory students. Some of these are summarized in Table 1.

Students:	Professionals:
Solve narrow, pre-defined problems of no personal interest.	Solve broad, open-ended and often self-discovered problems.
Work with laws presented by experts. Do not "discover" them on their own or learn why we believe them. Do not see them as hypotheses for testing.	Work with models to be tested and modified. Know that "laws" are constructs.
Use analytic tools to get "exact" answers to inexact models.	Use analytic and numerical tools to get approximate answers to inexact models.
Rarely use a computer.	Use computers often.

Table 1: Comparison of students' activities with those of a professional physicist.

To professional physicists, much of the pleasure of doing physics is associated with satisfying curiosity and learning surprising relationships and analogies of structure. Stimulating and satisfying one's curiosity requires starting with broad, naturally formulated questions and refining them successively in response to observation and analysis. Very little of this joy of the profession is present in the traditional introductory physics course. Many students entering college today, even some of those identified as among the brightest and highest achievers, have been trained to associate learning physics with rote memorization and application of memorized laws in narrowly defined situations. The traditional introductory course does little to disabuse them of these views. Students who might have become excellent research physicists change their majors because they see physics as "solved": constraining, routine, and dull.

A critical element in the activity of professional physicists, both theoretical and experimental, is *getting the physics right*. This means understanding which physical principles can be ignored and which must be kept given the accuracy one wants to calculate or measure. In physics, no theory or experiment ever calculates a result or measures an observable *exactly*, in the sense of a mathematical theorem. Physics is not an exact science, rather, it is a science where we believe we understand the accuracy of our approximation.[5] The art and creativity of the profession lie in viewing a complex situation and having the "empathy for the phenomena" to extract the simple principles that dominate the system. Because introductory students perceive that they are working with laws and problems that have exact answers, they are cut off from this fundamental aspect of the profession.

Removing the curiosity and creativity from physics often leaves it dry, obscure, and poorly motivated to the beginning student. Only the most inspiring and dedicated teachers are able to even display the presence of these components of the profession in the introductory course. To actually get the students to feel not only that the field possesses these characteristics, but that they themselves can actually perform innovative investigative physics seems like an unmeetable challenge. Fortunately, many bright students still choose physics as a profession. But fewer physics Ph.D.'s are granted today than two decades ago. Competition from sciences that didn't exist in 1960 is becoming increasingly strong. Can we continue to attract the best students if we hide from them the excitement of contemporary problems and the roles of curiosity and creativity in the profession?

B. Enter the computer

The primary constraint that has kept the profession from introducing more creative science at an early stage is the limited mathematical ability of the introductory student. Creative and open-ended problems using analytical tools require a level of mathematical sophistication not usually obtained by students until their third year of college. In the past decade, however, there has been an immense growth in the power and availability of computer tools and technology. More power is packed into a desktop computer the size of a breadbox than was available in mainframes thirty years ago. Programming environments have been transformed from complex line editing with batch compiling in FORTRAN to systems with full-screen editors, fast compilers, and interactive debuggers in unified, easy-to-use environments in Pascal and structured Basic. These developments open the possibility that students could be given the computer power to solve more interesting problems in the introductory course with little training.

A second aspect of computer use is that the power of the computer has become a major element in modern research physics. Doing physics with the computer is not the same as solving analytic problems and is not the same as learning to use the computer in a computer science course. In order that our majors learn to use this critical tool properly, it is important to give them an early start and specific training in solving physics problems with computers.

II. What are our goals for teaching physics majors?

Our primary goal is to transform the introductory calculus-based physics course so that students who have designated themselves as prospective majors can begin to both get training in a wide range of professional skills and develop an idea of what it would be like to be a physicist. Although there is considerable discussion among physics educators of what content to include, discussion of the general skills we are trying to teach is much less common.

A. Skill analysis

We analyzed the general skills we would want our students to have at the point they begin research on a Ph.D. thesis. Certainly each particular sub-discipline has particular content elements they would like students to have (such as experience with a Camac crate or ability to program fast Fourier transforms), but students who have a good grounding in strong general skills are able to successfully do research in almost any area that interests them. An outline of the general process skills required is given in Table 2.

1. Basic skills
    - A scientific framework
    - Number awareness
2. Theoretical skills
    - Analytic skills
    - Estimation and natural scales
    - Approximation skills
    - Numerical skills
3. Experimental skills
    - Error analysis
    - Mechanical skills
    - Device experience
    - Empathy for the apparatus
4. General skills
    - Intuition
    - Large-problem skills
    - Communication skills

Table 2: Skill analysis for physics students.

We define two basic skills: having a scientific framework and number awareness. By having a scientific framework we mean that the student must understand the "story-line" of science -- that science means observation, hypothesis, analysis, and testing against observation. By number awareness, we mean that the student must understand that aspects of the real world may be quantified by measurement and that the results of our mathematical analysis can tell us about what is happening (or will happen) in the real world. This skill is a *sine qua non* of doing physics. It is often assumed by the introductory physics teacher (incorrectly!) that students have a reasonable scientific framework. Nunmber awareness is stressed in some introductory courses.

We next add a set of theoretical or modeling skills. The first in this group are analytic skills: students should be able to write equations from word problems, to solve a variety of equations, and to interpret their results in terms of the physical world. Some aspects of these skills are stressed in traditional introductory courses. Other theoretical skills needed by physicists tend to be shortchanged in those courses, even when they are restricted to majors: estimation, approximation, and numerical skills.

These skills are essential in learning to model physical systems and to understand the implications of models built by others. Since physics is not an exact science, the "art" in the science is knowing what physical laws to apply under what circumstances and what additional complicating factors can be safely neglected. We call this "getting the physics right". It involves being able to estimate the size of an effect and to calculate corrections by understanding approximations. Today, it often involves putting physical insight into and getting it out of a complex numerical calculation. Yet these skills -- critical for both the professional physicist and the engineer -- are almost completely ignored in traditional introductory courses.

Since physics is a science whose results are continually tested and evaluated against the real world, a physicist needs experimental skills as well as theoretical ones. Majors are often trained in error analysis, mechanical skills, and given experience with a variety of devices. An experimental skill that we hope our students will develop as a concomitant part of their experimental experience is an "empathy for the apparatus". By this we mean an understanding of what is happening in an experiment, what is being measured, and where the information lies.[6] Yet this skill is primary to their understanding of what even basic physics means and how the equations we write down relate to the behavior of the real world.

Finally, there are a number of general skills that all professionals must develop. They must build an intuition for their field -- the ability to understand which tools apply in which circumstances and to have the complex network of internal checks that let them look at a wrong answer and have it not "feel" right. They must learn large-problem skills; the ability to take a significant problem and break it down into component, solvable parts in an appropriate manner while keeping track of the overall goal. They must also build communication skills. In physics, as in any field, it does not suffice to do brilliant work in a notebook or in your head. Physics, as is any field, is a social agreement of what it is we know. To interact with the community, a physicist needs to be able to present his or her results both in oral and written form in a clear and compelling fashion. This cluster of general skills is largely neglected in our professional training of physicists until they begin research in the second or third year of graduate school!

B. Principles of computer use

Our approach to the use of the computer in the classroom is based on three powerful principles:

> 1. It is not enough to use the computer to illustrate examples from the current curriculum. We must rethink the curriculum entirely from the ground up, now assuming the availability of the computer. What can we teach with it that we couldn't teach before?
>
> 2. The computer should not replace anything in the current environment: not the textbook, not the teacher, and not the laboratory.
>
> 3. The student should run the computer, not the other way round.

In our approach the student must play an active role. Students must learn to use the computer in an open ended way. The computer must be a powerful servant for the student, not a master.

III.. The M.U.P.P.E.T. Environment

For physics majors, our view of appropriate computer use implies that they will learn to program the computer themselves directly. In the past, this has represented a formidable obstacle to the use of the computer in physics courses. We could not expect all students to come to the course prepared to develop interesting programs, nor could we afford to devote much time to teaching programming in the physics class.

In order to overcome this barrier, we have developed modular programming materials that can be linked together by students to create sophisticated programs. With these materials built in Pascal, we were able to reduce the programming overhead for the student. Good Pascal programs are easy to read and easy to set up, so the physics and the structure of the analysis is easy to understand. We set up utilities for interactive input and for graphics output, and we provide self-documenting sample programs that allow even non-programmers to learn by example and to begin to build programs themselves without extensive training. (See Fig.1 for an advanced example.) Once they master Pascal, students can make an easy and straightforward transition to other professional languages such as FORTRAN or C.

A. Description and history of the project

We began M.U.P.P.E.T. in 1983. Because physics majors at Maryland are taught in small classes separated from majors in engineering and other sciences, we decided to begin the project with those classes. The introductory course for physics majors at Maryland is a three-semester sequence (Physics 171, 272, and

273). One semester of calculus is a pre-requisite. Many students arrive at Maryland with enough calculus to begin the course as first-semester freshmen. Physics 171 is offered in two sections in the fall and one section in the spring.

The computer and student programming have been used in introductory physics for majors at Maryland since 1985. Our materials development culminated in 1988 with the production of a supplementary manual, *Physics with the Microcomputer*.[7] In the past three years, the M.U.P.P.E.T. materials have been used with more than 200 students in both lecture and lab sections. The materials have been revised several times over the years as a result of student and faculty input.

Informal polling of our students indicates that our physics majors enter college with a broad variety of backgrounds. About one third of our incoming majors have extensive programming experience: they can program comfortably in two or more languages. Another third can program in one language. The rest of the students have little or no programming experience. Even this group has had significant exposure to computers. Almost all the students were able to master the required programming quickly and with enough skill to do computer homework problems and projects.

Because our group is self-selected as physics majors, they tend to be somewhat better prepared and have higher incoming SAT math scores (average of verbal + math of about 1250) than our broader group of engineering majors. This suggests that our results should be interpreted and extended with some care.[8]

B. The utilities and libraries

The M.U.P.P.E.T. environment provides a set of utilities for handling I/O and menus and a set of libraries to simplify various tasks. The utilities package includes:

data input screen procedures -- These let the programmer include exchange of data with a running program. The program puts up a data window with variable fields. An example is shown on the left side of Fig. 2. The programmer can include default values and the user can modify one or more of the fields using the keyboard or mouse. The user can modify the fields in any order and return to fields previously modified before choosing to go on with the calculation. The code to produce this data screen is given in the procedure `MakeDataScreen` shown in Fig. 1.

graph window procedures -- These let the programmer display output in one or more windows. The programmer can plot any number of curves in a window. The windows are scaled to a unit screen so the user never has to worry about details of pixel count and the variety of graphics displays and resolutions available. Two graph windows are shown on the right side of Fig. 2. The code to produce them is given in the procedures `GraphSetup` and `PlotIt` in Fig.1.

menuing procedures -- These let the programmer give the user choices to branch the program in a variety of ways. A sample of a menu appears at the top of the screen in Fig. 4.

parsing procedures -- These permit the programmer to let the user enter algebraic expressions into a running program and have the program interpret the results as strings of code, rather than just as strings of letters.

The utility procedures are provided to the students in compiled form.

A number of libraries have also been written and are provided to the students. The include a variety of routines for solving Newton's second law in one and two dimensions, a set of binary search procedures for solving equations, and an animation library.[9]

C. Sample programs

About 25 sample programs were developed at Maryland for this project using the M.U.P.P.E.T. utilities and environment. These programs were carefully designed to make them easy to read and to modify. They have been structured to provide examples of clear programming style.

The code for the program PROJ1D is shown in Fig. 1 and the screen it produces is shown in Fig. 2. This is a fairly sophisticated program and is used about halfway through the first semester. Students are asked to fill in the dummy procedure `StepEuler` to provide a solution using the Euler method. A fourth order Runge-Kutta routine is provided later using a similar calling statement.

The sample programs play a critical role in the overall computer use. Not only do they provide the students with concrete examples of computer use, they are flexible enough to permit students with limited programming experience to modify them to explore much more complicated systems as part of their independent project work.

D. Equipment required

The M.U.P.P.E.T. materials are designed to run on low end computer systems so as to provide maximum accessibility. The hardware required is:

- IBM compatible personal computer (XT, AT, or PS/2) with
- two floppy drives or one floppy drive and a hard disk
- at least 384 K of memory
- graphics capability (CGA, EGA, VGA, or Hercules)
- DOS 2.1 or higher

The system will permit larger programs, run faster, and look substantially better if it is used with:

- a full 640 K of memory
- EGA graphics or better
- a math co-processor.

In addition to the hardware, the user needs to supply Turbo Pascal^{TM} in one of the versions 4.0, 5.0, 5.5, and 6.0. (A version for Turbo Pascal for Windows known as *Window on Physics* is also available but has not yet been class tested.) A complete system running all of our M.U.P.P.E.T. programs and environment is currently available for under $600.

IV. What can we do with the computer in an introductory class?

With the computer and the environments described above, we considered a variety of curriculum changes:

- First, the order of the elements in the course may be changed to be more physically appropriate.
- Second, many of the professional skills traditionally short-changed at the introductory level can be introduced.
- Third, more realistic problems may be treated than in the traditional approach.
- Fourth, contemporary topics may be introduced at an early stage.

- And fifth, students may begin designing and carrying out their own research, even in the introductory course.

A. The order of elements may be rearranged to be more appropriate

The order of the traditional curriculum is strongly controlled by the mathematics the student takes in parallel. This often leads to unnatural and unphysical presentations. The computer sometimes allows us to bring the physics to the fore by introducing discrete forms of the fundamental laws. These can be much simpler to explain and understand than the continuous forms traditionally presented.

Example: Newton's Second Law

An example of this is the tradition of teaching uniformly accelerated motion (referred to in M.U.P.P.E.T. as "flat-earth gravity" to emphasize the approximate character of the model) before Newton's second law. A plausible excuse for doing this is that the former does not require any calculus for its solution, while the latter is a differential equation. The student taking calculus at the same time will be able to get some calculus done in math before being required to look at a differential equation.

Unfortunately, this approach runs counter to the underlying physics. Students find it very difficult to understand the motivation for the motion of falling bodies without the concept of force. The confusions developed at this early stage are likely to haunt the students throughout the course.

In the M.U.P.P.E.T. course, because we introduce the computer immediately, Newton's second law is introduced in a very simple and intuitive discrete form first using impulses.[10] The discussion of falling bodies then uses the concept of force and the dynamic form of Newton's second law.

B. Professional skills may be introduced at an earlier stage than is usual

As discussed above, approximation plays a critical role in understanding how physics works. We are forever making simplified models of real-world systems. Yet for our undergraduates, especially at the introductory level, because of their lack of mathematical sophistication, we are forced to treat most problems as mathematics rather than as physics problems.[11] The presence of the computer lets us introduce corrections and their sometimes striking effects.

Example: The large amplitude pendulum

One of the best examples of an approximate equation is the large amplitude pendulum. An excellent model equation for an idealized pendulum is derived in most texts and is within the reach of most of our students:

$$\frac{d^2\theta}{dt^2} + \frac{g}{L}\sin\theta = 0$$

This equation, unfortunately, requires advanced special functions for its analytic solution.[12] For small amplitudes, the equation becomes

$$\frac{d^2\theta}{dt^2} + \frac{g}{L}\theta = 0,$$

directly equivalent to the simple harmonic oscillator equation. Essentially all introductory texts give both these equations. The first is ignored except for the construction of the correct form of the energy. Dynamics problems are done with the second equation.

In the M.U.P.P.E.T. class, we are able to consider the large amplitude equation in more detail. Because we are solving Eq. (1) numerically, we have the situation shown in Fig. 3. Many students assume that, because the analytic expression can be expressed in closed form, it is the "better" solution. We can bring them to a dramatic contradiction of this viewpoint by asking them to consider the analytic and numerical solutions for the cases:

$$\theta_0 = 175\ ^{o} \qquad \omega_0 = 30\ ^{o}/\mathrm{s}$$

$$\theta_0 = 355\ ^{o} \qquad \omega_0 = 0\ ^{o}/\mathrm{s}$$

In the first case, the correct (numerical) solution goes "over the top", spinning round and round the pivot. The analytic solution goes over the top some distance, stops in midair, turns around, goes back over the top. The second case is a small angle oscillation, but the "analytic" solution doesn't recognize this. Instead of falling and oscillating through a small angle, it rises over the top, oscillating back and forth nearly a full circle each time.

The explanation of these strange results is fairly simple. The small angle approximation doesn't hold for large angles, so the analytic form shouldn't be applied. But these simple examples illustrate a principle of great importance.

> *When approximate solutions are extrapolated beyond their realm of validity they can give results that are qualitatively wrong.*

Again, this result is obvious to the professional physicist, but there is essentially no example of this important result anywhere in the traditional introductory curriculum. If our students are to learn the art of approximation, they must have simple touchstone examples that clearly illustrate the possible pitfalls.

One additional point is important and illustrative about this case. Note that the mathematics of solving the approximate equation is simpler. But from the students' point of view, the ideal pendulum requires one extra logical step to set up its equation of motion and is therefore conceptually more difficult than the realistic one. Indeed, many students have a poor understanding of the small angle approximation. With M.U.P.P.E.T., we can discuss the exact case first.

C. More realistic problems may be treated than in the traditional approach

The traditional curriculum is largely restricted to those problems and methods that are analytically tractable. The powerful tools that permit the solution of almost any problem are usually touched only in passing since they require numerical methods. In a computer-based course, they can be restored to their rightful importance.

Example: Projectile motion with air resistance

An example of the possibilities opened up by including the power tools is the discussion of air resistance in the M.U.P.P.E.T. course. The importance of including this topic goes far beyond the issue of simply making our description of motion more realistic. With this example, a large number of valuable reasoning tools can be introduced that are usually ignored until much later in the curriculum.

A combination of dimensional reasoning and symmetry principles can be used to construct the Newton drag law:

$$\mathbf{F}_{\text{air res}} = -\,\eta\rho R|v|\mathbf{v} = -b|v|\mathbf{v}$$

where η is a dimensionless parameter. This has interesting consequences[13] and further discussion can be given during the section on kinetic theory.

We then use a M.U.P.P.E.T. program to study the behavior of an object under the influence of this force. The total amount of programming required from the student is to put in the equation for the force law into the program `Proj1D`. The output of this program is shown in Fig. 2.

The students can be asked to carry out an interesting mix of qualitative and quantitative analyses. In studying the qualitative behavior we can ask the student the following questions:

- What is the effect of including air resistance for an object thrown straight up? When there is no air resistance, it takes a projectile the same time to go up as to come down. Does this change when air resistance is added? Give a qualitative argument to show which way it should work and use the program to demonstrate the correctness of your reasoning.

- What is the effect of including air resistance for an object thrown at an angle? When there is no air resistance it travels the same horizontal distance while rising to its maximum height as it does descending. Does this change when air resistance is added? Give a qualitative argument to show which way it should work and use the program to demonstrate the correctness of your reasoning.

- When the object is very light or the air resistance coefficient is large, something strange happens to the velocity of a falling object. Use the program to determine what this is.

Once the student has observed the phenomenon of terminal velocity qualitatively on the computer, we can ask them to derive the expression for terminal velocity analytically. With the computer program in hand, we can ask the students to study some interesting realistic cases. Here are two sample homework problems that can be done at the end of the unit:

Sample problem 1: A 10 gm sheet of paper is crumpled up into a compact ball with a radius of 4 cm. When dropped, it takes 1.0 sec to fall a distance of 2 meters. Use this to determine the air resistance coefficient b in the force law $\mathbf{F}_{\text{air res}} = - b|v|\mathbf{v}$. If a wooden and a steel ball are dropped from the same height, how long would they each take to fall? What accuracy would you need in your measurements to see the difference in the rates of fall between the wooden and steel balls?

Sample problem 2: A ball of mass 0.14 kg is thrown straight up with a speed of 20 m/s. It comes down 1 second earlier than expected, if air resistance is ignored. Find the air resistance coefficient b for this object if the force has the form

$$F = - b\ v\ |v|.$$

Find the coefficient γ if the force has the form

$$F = -\gamma\ v.$$

Design a simple experiment (with numbers!) using this ball to determine which force gives a better description of the real world.

The example shows that the actual computational work and the programming involved is a fairly small part of the unit. But having it present permits us to bring in scale analysis, dimensional analysis, and to demonstrate approximation techniques and ways of extracting physics from computer programs. These are all skills that the professional must know, but which we have had little opportunity to teach in undergraduate courses.

D. Contemporary topics may be introduced

Once power tools are put in the student's hands, a much wider variety of problems can be addressed. These include more realistic problems than are usually handled, as well as ones of contemporary interest.

Example: Chaos theory

Recent developments in the theory of classical mechanics have stressed the sensitivity of non-linear classical problems to initial conditions. Although this sensitivity has been known to workers in the field, especially Lagrange and Poincare, when we teach Newtonian mechanics we tend to tell only half the story. We traditionally emphasize that:

> *In principle, classical systems are totally predictable once starting conditions are specified.*

However, it is equally important that the student understand the contemporary lesson of chaos theory which emphasizes that:

> *In practice, it is usually impossible to predict the long-term motions of any classical system with a finite calculation since they are highly sensitive to the starting conditions.*

In the M.U.P.P.E.T. course, the presence of the computer allows us to include a segment on chaos theory as a natural extension of Newtonian dynamics at the end of the first semester. The students find this topic of great interest, and many of them choose some non-linear problem as a research topic in the second semester. A screen from the M.U.P.P.E.T. sample program Iterate is shown in Fig. 5. This program illustrates the phenomena of bifurcation, period doubling, and repeatable randomness. The code for this program contains only 60 lines, many of which can be clipped from the basic sample programs.

E. Students may begin research at an early stage

Introducing the computer to students at an early stage gives them the power to investigate a wide variety of complex problems in an open-ended inquisitive fashion and thereby begin to get a real exposure to how science is done in practice. We have had considerable success with first and second year physics majors performing independent projects in the M.U.P.P.E.T. course.

When one of us (EFR) taught the M.U.P.P.E.T. course, he required project work of every student in every semester. An earlier attempt to require projects of sophomore physics majors in 1970-72 had not been successful. Only about 15% of the students were able to do projects that had any characteristics of normal scientific research. Most students were severely hampered by not having sufficiently strong analytic and mathematical skills to carry through an open-ended investigation. However, in the M.U.P.P.E.T. environment with freshman majors in 1986-1989, when each student had access to a computer and the M.U.P.P.E.T. tools, the results were strikingly different. About two thirds of the students were able to do valuable and interesting projects.

The students were told to seek a topic they were interested in and would like to know more about. It had to have some relation to the content of the course, although we tended to be flexible if a student showed strong interest in some other topic. They were told that this was not to be a project where they read, organized, and replayed other people's materials. They were supposed to design their own project, carry out an investigation, and write a report.

In an ideal project, we believe that the student should carry out the following activities:

- Formulate a question of broad general interest.
- Perform library research on the subject.
- Reformulate the question more sharply so it is amenable to modeling or an experiment.
- Set up the calculation or experiment and run it.
- Check the results for consistency, accuracy, and correctness.
- Extract some physical insight from the results.
- Propose further research based on the results.
- Present the results both in written and oral form.

The professional researcher does all of these. Consider for your own undergraduate majors' curriculum: At what point in their undergraduate training do each of your students get experience with each of these kinds of tasks? Of course few of our freshman projects succeeded in accomplishing *all* these goals. We were surprised and delighted that some did, but we considered a project successful if a student demonstrated five of the eight activities.

Some of the subjects investigated by students in conjunction with our calculus-based physics course include:

Colliding galaxies
Tethered satellites
The flight of the frisbee
Pumping a pendulum swing
The Lyapunov exponent in
    the Sinai billiard problem
The effect of the backboard
    in basketball
The motion of a golf ball
The evolution of light from galaxies
The motion of a spinning ping-pong ball

Shepherd moons and Saturn's rings
Capture of a planet by a wandering star
Grain boundary growth in crystals
Diffractive lens design
Period doubling and chaos in the
    van der Pol oscillator
The planet Nemesis and its effect
    on the Oort cloud
Designing an airplane wing

It would have been a substantial burden on the teacher of the class to advise semester long research projects for even a class of 25 students. Fortunately, we had the support of the Maryland physics department. In most cases, students were sent to other faculty with particular relevant expertise for advice.[14] This turned out to have a substantial benefit. The undergraduate majors met faculty on a personal basis at an early stage in their careers. This resulted in students making valuable contacts that often developed into research projects when they became juniors and seniors. Since the start of project work in the introductory courses, the number of upperclass majors seeking to do independent research projects has grown from essentially zero (1-2 per year) to a substantial fraction of the class (10-12 per year).[15]

At universities with very high admission standards, it is well known that freshman can do independent research (viz. winners of the Westinghouse competition). What was not widely appreciated before M.U.P.P.E.T. was that freshmen with a wide range of abilities and backgrounds can begin to design their own research projects and carry them out successfully if the physics course is tied to the computer and the students empowered in its use.

Perhaps the most surprising result was the distribution of good projects. Students who would have been identified as mediocre students by their exam grades occasionally did outstanding projects. A careful analysis of these students showed that they had "stylistic" rather than content problems. They did not perform well under exam pressure, but preferred to work slowly and carefully. Some of them had extraordinary intensity and persistence when they were interested and involved in a project. Others who did well on exams could find no topic at all to interest them and turned in very poor research projects. These observations raise the question whether it is a good idea to use a student's performance on traditional timed hour exams as a "first cut filter" to weed out those who should not be physicists. This automated hoe may be chopping some valuable flowers!

F. M.U.P.P.E.T. can be used in association with the Laboratory

The M.U.P.P.E.T. approach has been used by one of us (JMW) to provide modeling and analysis tools in conjunction with the laboratory that accompanies the first semester introductory physics course (mechanics).[16] (M.U.P.P.E.T. can also be used in the laboratory directly to serve as a user interface with the computer to accumulate data directly from analog to digital converters.[17]) The laboratory was taught by alternating between a traditional "hands-on" laboratory one week and a computer modeling/data analysis session the next. The students used M.U.P.P.E.T. to calculate means and standard deviation and performing linear least square fits. They also developed computer models and compared the model's predictions with experimental observations.

For example, in one of the laboratories, students observed and took data on the motion of a pendulum using a stroboscope and a Polaroid camera. They then used their graphing and data analysis package to plot the observed angle, and to calculate and plot the angular velocity, angular acceleration, kinetic energy, potential energy, and total energy vs. time. They then used M.U.P.P.E.T. to build a mathematical model of the system on a template we provided. With their program, they were able to compare the prediction of a mathematical model with their observations and make an estimate of the damping. With this model, students were able to extend their analysis of their investigations to consider large angle corrections, driving forces, and resonance.

Instead of listening to lectures about modeling and error analysis (as had been the previous practice -- even in the lab section of the class!) the students performed the activities themselves using our computer tools. The students' projects were significantly improved in quality and showed a better understanding of the phenomena than when they worked in the traditional mode.

V. What did we have to leave out?

We are certain that many of our readers will respond to our claims with skepticism. Everyone who has taught the traditional introductory calculus-based physics course has had the problem of having too much material to cover. How could we possibly teach programming, add more realistic problems, include the large amplitude pendulum, air resistance, random walks, and integrals over non-simple charge distributions (among others)? What did we leave out?

Indeed, some materials were left out that are included in the traditional course. Rigid body motion was suppressed, as was fluid dynamics and much discussion of sound. In our three semester sequence we did not do any modern physics or relativity, in part since the sequence is followed immediately by a full two semester sequence on modern physics.

The additional materials, however, did not take very much time to include. The programming handouts were read in parallel with the standard reading, a few pages per week. Less than 5% of the lecture time was used to discuss programming. Two to three lecture hours per semester were actually spent in the microcomputer laboratory with the students getting started on some of the more computer oriented homework assignments.

From this small basis of computer instruction, we were able to include one to two homework problems per week that were somehow associated with the computer. (These were occasionally estimation, analytic, or qualitative problems.) These were done at the cost of reducing the number of standard "plug-and-chug" problems the students were assigned.

VI. Is M.U.P.P.E.T. transferable?

One question to consider is whether the M.U.P.P.E.T. environment is transferable. Many educational developments work well as long as they are taught by their creators, but do not succeed when taught by anyone else. There is good evidence that M.U.P.P.E.T. is usable elsewhere.

The University of Maryland is a reasonably typical university environment.

The College Park campus of the University of Maryland is a large state university with a large student body having a wide range of interests, backgrounds, and levels of ability. Materials developed and tested at Maryland should be usable at many campuses across the nation.

M.U.P.P.E.T. at Maryland has not been restricted to its developers.

The M.U.P.P.E.T. course for physics majors has been and is being taught at Maryland by faculty not involved in the development of M.U.P.P.E.T.

New courses have been developed with M.U.P.P.E.T. materials in Australia.

Prof. Ian Johnston at the University of Sydney became acquainted with the M.U.P.P.E.T. idea at the Raleigh Conference on *Computers in Physics Instruction* in the summer of 1988.[18] Since then, he and his colleagues have developed materials for second year physics majors on numerical methods and quantum mechanics. In 1989-90, he used the M.U.P.P.E.T. environment to integrate computational physics into the undergraduate courses at Sydney University.[19]

Johnston tested M.U.P.P.E.T. in three successive semesters. In the first test, 18 volunteers, chosen from a class of 200 second-year students, were given six four-hour microlab sessions in addition to the normal work in a course in quantum mechanics. In the second test, this was repeated with 92 students out of a class of 202. The third test involved 24 students in the third year class. These students were asked to work through four computer modeling problems in diverse areas of physics including solid state, kinetic theory, plasma physics, and Fourier transforms. Johnston and McPhedran conclude:

(1) Students do not need to be able to program before they can handle these materials. Students who had no previous programming experience (about 25% and 16% of the students in the first two trials) had to work harder at first, but had little trouble once they got started.

(2) The students' understanding of a number of traditional subjects was significantly improved by adding computer modeling problems as shown by grades in a comparison of the students in the test and

traditional groups on traditional tests. This was because the computer programs allowed students to explore many more cases than they could by hand. For example, because the students in these trials had seen the shapes of many different wave functions, they could easily answer questions in ordinary texts dealing with the geometrical property of eigenfunctions and performed significantly better than students in the traditional group on such questions in exams.

(3) In the last microlab, students were asked to investigate broadly posed problems. They were able to successfully design and complete projects in a one week period, thanks in a large part to the ease of building models with the M.U.P.P.E.T. software package. This confirms our experience with projects at Maryland.

We conclude from these experiences that the M.U.P.P.E.T. environment is robust and survives being transferred to other users.

VII. Conclusions

Summary

We have reported on the development of the M.U.P.P.E.T. utilities -- a flexible and powerful computing environment that permits introductory students to add programming to their tools for solving physics problems quickly and easily and with minimal overhead. When these tools are added to the traditional calculus-based introductory physics course, the students' power to solve problems expands enormously. This opens many possibilities for changing the curriculum. Elements may be rearranged in a more natural order; professional skills may be introduced at an earlier stage than is traditional; contemporary topics such as chaos and quantum theory may be introduced; and students may begin research immediately.

Our conclusion is that M.U.P.P.E.T. works well for majors in small classes. We have not yet tested whether these methods can be extended to large classes with other scientists such as chemists and engineers. It may be possible if the infrastructure exists to provide students with sufficiently accessible networked computer resources. Success in this environment could also be aided substantially by good coordination with other departments.

Future developments

As a result of its strength as an open environment capable of growing as the student's strength grows (and as the power of the computer grows), the M.U.P.P.E.T. utilities have been adopted as the basis for two multi-university projects of national scale -- CUPS and CUPLE.

The Consortium for Undergraduate Physics Software (CUPS) is a project based at George Mason University and funded by the NSF to add computers to upperclass physics courses. A group of 27 physics faculty with software design experience are developing six manuals to accompany upperclass physics courses. Each manual contains nine simulations, each of which will add an element of new physics, not easily includable without the computer.

The Comprehensive Unified Learning Environment[20] (CUPLE) is a project to bring together in a single unified computer environment some of the successful attempts to reach more introductory physics students and to train them more effectively and professionally. CUPLE is bringing together sophisticated tools for handling graphing, student programming, laboratories, and video with modularized text materials and a database of information. The M.U.P.P.E.T. environment is being upgraded for this project to an object-oriented approach now called *Window on Physics* (or WinPhys for short). WinPhys is built on Turbo Pascal for Windows™ and takes full advantage of the Graphical User Interface (GUI) Microsoft Windows™ 3.

Acknowledgments

M.U.P.P.E.T. was supported by the Fund for the Improvement of Postsecondary Education. The project has involved a large number of faculty, students, and visitors who played various roles throughout the years. We particularly acknowledge the collaboration and efforts of those faculty members who participated in the development of the original conception of M.U.P.P.E.T. and who have contributed extensively to its development: Profs. Charles Misner, Bill MacDonald, and Jordan Goodman for concepts and fundamental ideas; James Harold, Ken Hennacy, Gerhard Norkus and Madhura Nirke for developing programs and utilities. Visitors to the program who made important contributions include Ian Johnston (Sydney), Gordon Aubrecht (Ohio State), Ed Taylor (MIT), Pat Cooney (Millersville), Steve Hanzely (Youngstown), and Gunther Kurz (Esslingen).

Endnotes

[1] W. M. MacDonald, E. F. Redish, and J. M. Wilson, "The M.U.P.P.E.T. Manifesto", *Computers in Physics*, **2**(4), 23-30 (July/Aug 1988).

[2] F. Verbrugge, "Conference on Introductory Physics Courses", *Amer. J. of Phys.*, **25,** 127-128 (1957) ; "Improving the Quality and Effectiveness of Introductory Physics Courses", *ibid.* 417-424; F. Bitter et al., "Report of Conference on the Improvement of College Physics Courses", *ibid.*, **28**(1960) p. 568-578.

[3] R. L. Devaney, *An Introduction to Chaotic Dynamical Systems* (Addison-Wesley, 1989).

[4] J. S. Brown, A. Collins, and P. Duguid, "Situated cognition and the culture of learning", *Educational Researcher*, p. 32-42 (Jan-Feb 1989) .

[5] The most precise comparison of theory and experiment occur in quantum electrodynamics, where the g-2 value of the muon and the binding energy of the helium atom can be calculated to more than 10 significant figures. At greater than this level of accuracy, one runs into the problem of virtual production of strongly interacting particles where the theory does not yet exist to permit further improvements.

[6] We are greatful to John Risley for a discussion of this idea.

[7] E. F. Redish, J. M. Wilson, and I. P. Johnston, *Physics with the Microcomputer*, to be published.

[8] We have focussed in this work on the course for physics majors. Some preliminary testing of the use of student programming in large classes with engineering students was begun in the fall of '91. This effort is to focus more on conceptual problems and building

up a strong view of how one does physics than on developing professional skills.

[9] These last two libraries were developed by I. P. Johnston at the University of Sydney.

[10] E. F. Redish and Edwin Taylor, "Impulse Mechanics", *AAPT Announcer* **17**(4), 82 (Dec. 1987)

[11] This tends to hold largely for textbook problems. The classic laboratory in which the student measures the value of g to a high accuracy with a pendulum and calculates many corrections provides one of a number of excellent counter-examples to this statement.

[12] E. T. Whittaker and G. N. Watson, *A Course of Modern Analysis*, *Fourth Edition* (Cambridge U. Press, 1952).

[13] E. F. Redish, "The impact of the computer on the physics curriculum", in **Computers in Physics Instruction**, E. F. Redish and J. S. Risley, eds. (Addison-Wesley, 1990), p. 15-22.

[14] In the sections taught by EFR, students were instructed: "Your paper has to teach the teacher something he doesn't know in order to earn an A." This had the effect of encouraging them to seek advising elsewhere in the department.

[15] The total number of upperclass majors has remained constant at about 50.

[16] J. M. Wilson, "Combining computer modeling with traditional laboratory experiences in the introductory mechanics laboratory for physics majors", AAPT Announcer, **17**(2), 80 (May, 1987).

[17] Ian Johnston, Sydney University, private communication.

[18] **The Conference on Computers in Physics Instruction, Proceedings**, E. F. Redish and J. S. Risley, eds. (Addison-Wesley, 1990)

[19] I. D. Johnston and R. C. McPhedran, "Computational Physics in the Undergraduate Curriculum", submitted to the The Australian Physicist.

[20] J. M. Wilson and E. F. Redish, "The Comprehensive Unified Physics Learning Environment: Part I. Background and system operation", *Computers in Physics*, **6**(2) (Mar/April 1992), 202-209; "..: Part II. The basis for integrated studies", **ibid. 6**(3) (May/June 1992), 282-286 .

Figure Captions

Fig. 1: The Pascal source for the program PROJ1D.

Fig. 2: The input and graphics screens produced by the program PROJ1D.

Fig.3: The structure of the solutions to the large amplitude pendulum equation

Fig. 4: The screen displaying large amplitude "over-the-top" motion for a pendulum. From the M.U.P.P.E.T. sample program `Pendulum`.

Fig. 5: Result of iterations of the logistic function in the chaotic regime. From the M.U.P.P.E.T. sample program `Iterate`.

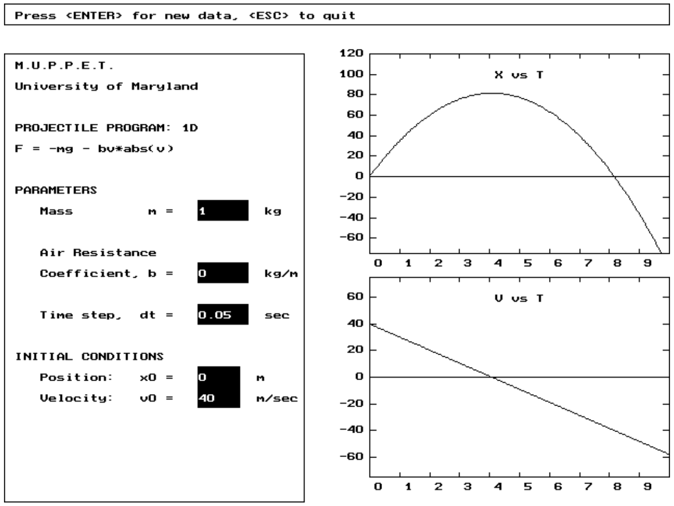
```
PROGRAM Projectile1D;              { proj1d.pas }

{***********************************************}
{*                                             *}
{*        Program to calculate motion of       *}
{*        a particle in 1D with gravity        *}
{*        and air resistance using RK2.        *}
{*                                             *}
{***********************************************}

USES
   Crt,Dos,Graph,Printer,MUPPET;

CONST
   numData : Integer = 200;    {Number of points to
plot}
   g       : Real    = 9.8;    {m/sec/sec}

VAR
   t,x          : DataVector;   { time, position    }
   v,a     : DataVector;   { velocity, accel   }
   x0,v0        : Real;         { initial conds.    }
   m            : Real;            { mass              }
   b            : Real;            { air resis. coeff. }
   dt           : Real;         { time step         }
   i            : Integer;      { loop variable     }
   IC           : Screen;       { data screen       }
   act          : Char;         { control character }
 { The types 'DataVector" and "Screen" are defined  }
 { inside the unit MUPPET.                          }


{*------------ Physics Procedures -------------*}

FUNCTION Force(x,v,t:Real) : Real;
   BEGIN
      Force := -m*g - b*v*abs(v)
   END;

{*---------- Mathematics Procedures -----------*}

{ Second order Runge-Kutta routine for stepping }
{ from variables at time t (In variables) to    }
{ variables at time t+dt (Out variables).       }

PROCEDURE StepRK2(xIn, vIn, tIn, aIn,tStep:Real;
              VAR xOut,vOut,tOut,aOut:Real);
   VAR
      xHalf,vHalf : Real;
      tHalf,aHalf : Real;
   BEGIN
      tHalf := tIn + 0.5*tStep;
      xHalf := xIn + 0.5*vIn*tStep;
      vHalf := vIn + 0.5*aIn*tStep;
      aHalf := Force(xHalf,vHalf,tHalf)/m;
      tOut  := tIn  + tStep;
      xOut  := xIn + vHalf*tStep;
      vOut  := vIn + aHalf*tStep;
      aOut  := Force(xOut,vOut,tOut)/m;
   END;

{*--------- Data Screen Procedures ------------*}

PROCEDURE MakeDataScreen;
   BEGIN
      DefineInputport(0,0.45,0,0.9);
      _A[01]:='"M.U.P.P.E.T."                      ';
      _A[02]:='"University of Maryland"            ';
      _A[03]:='                                    ';
      _A[04]:='"PROJECTILE PROGRAM: 1D"            ';
      _A[05]:='"F = -mg - bv*abs(v)"               ';
      _A[06]:='                                    ';
      _A[07]:='"PARAMETERS"                        ';
      _A[08]:='   "Mass         m = " 0.14++ "kg"  ';
      _A[09]:='                                    ';
      _A[10]:='   "Air Resistance"                 ';
      _A[11]:='   "Coefficient, b = " 0+++++ "kg/m"';
      _A[12]:='                                    ';
      _A[13]:='   "Time step,  dt = " 0.050+ "sec" ';
      _A[14]:='                                    ';
      _A[15]:='"INITIAL CONDITIONS"                ';
      _A[16]:='   "Position:   x0 = " 0++++ "m"    ';
      _A[17]:='   "Velocity:   v0 = " 30+++ "m/sec"';
      LoadScreen(IC,17);
   END;

PROCEDURE GetScreenData(VAR m,b,x0,v0,dt:Real);
   BEGIN
      ClearMUPPETport;
      Message('Press <ENTER> to plot, <ESC> to quit');
      Accept(IC);                    {displays screen}
      m  := Valu(IC,1);  {puts 1st entry on IC into m}
```

```
      b  := Valu(IC,2);  {puts 2nd entry on IC into b}
      dt := Valu(IC,3);
      x0 := Valu(IC,4);          {etc...}
      v0 := Valu(IC,5);
   END;

{*----------- Graphics Procedures -------------*}

PROCEDURE GraphSetUp;
   BEGIN
      GraphBackColor:=DarkGray;
      DefineViewport(1, 0.55,1, 0.5,0.9); {Define
ViewPort 1}
      DefineViewport(2, 0.55,1, 0.05,0.45);
{ViewPort 2}
      DefineScale(1, 0, 10, -75.0, 75);       {Define
Scale 1}
      DefineScale(2, 0, 10, -75.0, 75);      {Define
Scale 2}
   END;

PROCEDURE PlotIt(viewPort,color:Integer; x,y:DataVector;
                 nameLabel:BigStr);
   BEGIN
      Setcolor(color);
      SelectScale(viewPort);
      OpenViewport(viewPort);
      Axis(0,0,1,20);
      PlotData(x,y,numData);
      PutLabel(Inside,nameLabel);
   END;


{*--------------- Main Program ----------------*}

BEGIN
   MUPPETinit;
   MakeDataScreen;
   GraphSetUp;
   REPEAT
      GetScreenData(m,b,x0,v0,dt);
      IF EscapedFrom(IC) THEN
         BEGIN
            MUPPETdone;
            EXIT
         END;
```

```
      t[1] := 0;                   {initializes first
step}
      x[1] := x0;
      v[1] := v0;
      a[1] := -g - b*v0*abs(v0)/m;

      FOR i := 2 to numData DO        {solve the
equation}
         StepRK2(x[i-1],v[i-1],t[i-1],a[i-1],dt,
                 x[i],  v[i],  t[i],  a[i]);

      Message('Press <ENTER> for new data, <ESC> to
quit');
      PlotIt(1, lightGreen, t, x, 'X vs T');
      PlotIt(2, lightRed,   t, v, 'V vs T');

      act := ReadKey;
   UNTIL ord(act) = 27;

   MUPPETdone;
END.
```

Fig. 1

© American Association of Physics Teachers 1999, published in *The American Journal of Physics* **61** (1993) 222-232.



```
Press <ENTER> for new data, <ESC> to quit

M.U.P.P.E.T.
University of Maryland

PROJECTILE PROGRAM: 1D
F = -mg - bv*abs(v)

PARAMETERS
   Mass          m =  1      kg

   Air Resistance
   Coefficient, b =  0      kg/m

   Time step,  dt =  0.05   sec

INITIAL CONDITIONS
   Position:   x0 =  0      m
   Velocity:   v0 =  40     m/sec
```

Fig. 2

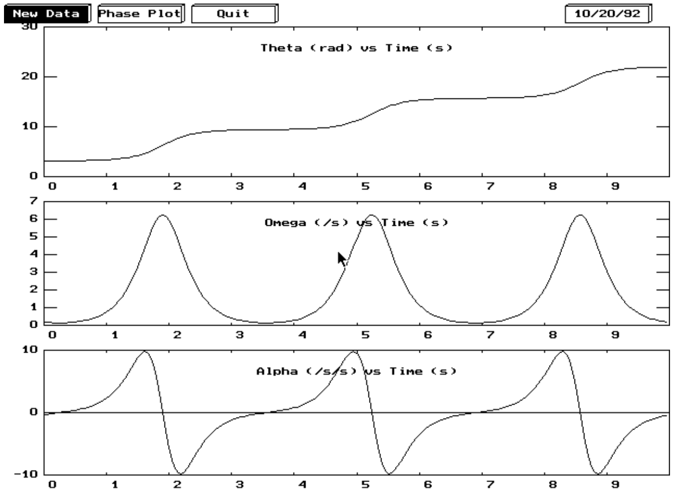


Fig. 4

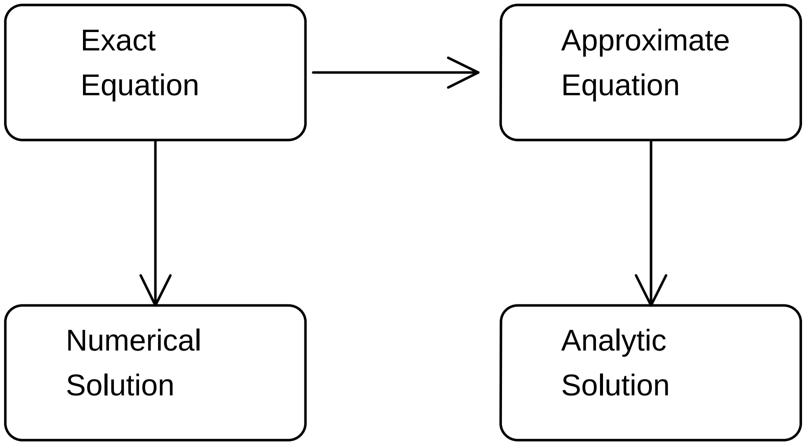
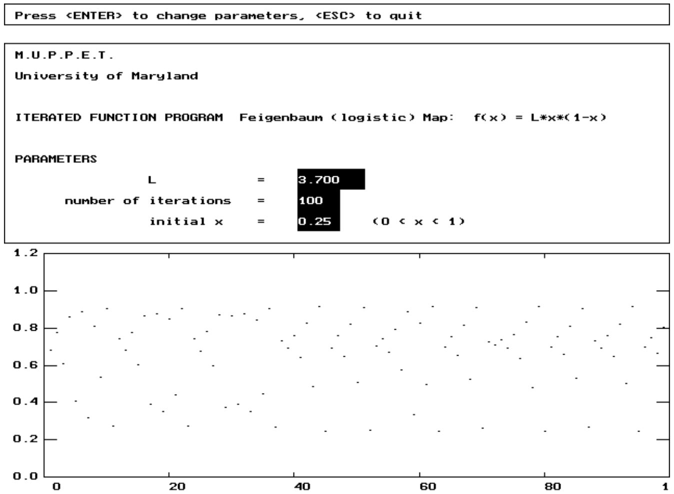


Fig. 3



```
Press <ENTER> to change parameters, <ESC> to quit

M.U.P.P.E.T.
University of Maryland

ITERATED FUNCTION PROGRAM  Feigenbaum (logistic) Map:   f(x) = L*x*(1-x)

PARAMETERS
                L               =    3.700
      number of iterations      =    100
                initial x       =    0.25     (0 < x < 1)
```

Fig. 5