Abstract:

Traditional numerical methods courses face several difficulties. There is a performance problem for students who need to combine different skills. Students spend so much time mastering the details that they often miss the point as to why there is a need for scientific computing.

Using projects drawn from practical situations, and a good high-level package such as MATLAB,\(^1\) can overcome much of this. The difficulty associated with learning scientific programming is greatly alleviated. This in turn frees more time for studying more advanced methods.

The need to combine ideas from different areas makes this an ideal candidate for both team-teaching and student teamwork on the projects.

In this paper, we discuss the use of this approach in a variety of scientific computing courses at different levels.

\(^1\) MATLAB is a registered trademark of The MathWorks, Inc.
I. Introduction

In this paper, we discuss the use of projects as a vehicle for teaching numerical methods and scientific computing. The traditional approach to such courses has often failed for a high proportion of the students for a number of reasons.

Typically, there is a performance penalty for students resulting from the need to combine skills from different, and largely independent, areas of study. Knowledge of, and skill in, modeling, mathematics and programming are all important here. Although not entirely independent, the probabilities of success in the various areas can reasonably be multiplied. A decent B student in each of these now has a combined grade of \((0.85)^3 = 61\%\)!

This grade penalty, which, in reality, is probably not quite as severe as the example suggests, has side-effects which are similarly detrimental to the course. Because a potentially good student struggles, others become disheartened. Students spend so much time and effort mastering the combined details of the methods, their application and their implementation, that they lose all sight of the “big picture”. This often means they fail to see the need for numerical methods at all. “Mathematical problems have an exact right answer” is typical of the student approach to the discipline. This attitude is reinforced by traditional calculus courses, too.

Because of the time and performance penalties, traditional numerical methods courses cannot usually cover state-of-the-art methods. This, in turn, makes it harder to deal with practical problems.

Using projects drawn from practical situations, and a high-level numerical package such as MATLAB, can overcome much of this. Motivation should be enhanced by the use of projects, which should also convince students of the experimental nature of much scientific computing.

Using MATLAB reduces the difficulty associated with learning scientific programming. Even computer science students find the nature of mathematical programs very different from their previous programming experiences. The additional benefit is that, by easing the initial programming difficulties, more time can be spent on more
advanced methods. Moreover, MATLAB has many advanced techniques included. After programming some of the basic methods, the student can experiment with more advanced methods without the need to program them from scratch.

There are new difficulties and challenges that come with this approach. Students tend to treat courses in different disciplines (or even subdisciplines) in isolation. Combining ideas from science, (several branches of) mathematics and computing makes such a course a natural candidate for team teaching and for student teamwork, especially if students from different majors can be teamed up.

In the remainder of this paper we discuss some experiences with this approach in a variety of scientific computing courses, from elementary numerical methods to advanced undergraduate courses.

In the next section we focus on some of the projects used in different courses. The descriptions include a brief summary of the techniques employed in their solution. This will give an idea of the coverage of the courses. Some critique of their success is also included. The presentation of the models could be enhanced in places by team teaching.

In Section 3, we discuss assessment issues. In most of these courses, apart from the most elementary, there are no formal examinations. That leads to some difficulties in the assessment process and these will also be discussed. Related to the question of assessment is the issue of teamwork, which is also discussed. In Section 4, we give a brief summary of student and faculty reaction to this approach. At the U S Naval Academy, all students complete opinion forms at the end of each course. These provide the basis for this summary. Section 5 considers, briefly, the extension of the ideas presented here to a Computation Science major. We conclude with a short summary.

II. Some of the Projects Used

The courses where projects have been used extensively are
Computer Calculus: a two-semester sequence of freshman calculus with computing; all volunteers, any major.

Introduction to Scientific Computing: the introductory course; primarily mathematics major juniors.

Advanced Numerical Analysis (old off-putting name!): mostly mathematics or systems major seniors.

Computer Arithmetic: mostly seniors, mathematics and computer science majors

Introduction to Parallel Scientific Computing: mostly seniors, mathematics, computer science, systems and physics majors.

The advanced classes tend to be small and lend themselves to less formal treatment. The introductory scientific computing course is a requirement for the mathematics majors. Typically there are three sections of up to about 15 students each. For the most part, we shall concentrate on this course.

One of the biggest changes that has resulted from teaching the course through projects has been a major shift of emphasis. No longer does the analysis of the numerical methods assume such an important role. The course is much more concerned with solving problems. The projects are chosen to try to give a broad introduction to the subject. Deeper study of the theory behind the methods is left to later courses for those who desire it. This change is certainly appropriate if the course is to become genuinely interdisciplinary, rather than a mathematics course taken by some other students as well.

It is based entirely on projects with some associated homework assignments. There are five projects each occupying about three weeks. This is also the course in which the students are introduced to MATLAB, so some time must be allocated to basic MATLAB programming at the outset. Most of the programming instruction is “in context”, or “just-in-time”. This reduces the time devoted solely to programming. It also reflects the belief that programming, like most things, is best learned in context. Loops or conditional branching in the abstract cause students much more difficulty than, for example, while loops as part of the implementation of a simple iterative scheme.
The projects used most recently, and the methods introduced to solve them, are described below. The students were provided with background notes on the projects and with introductory notes on using MATLAB. These notes [2] and [3] are available from the author.

A. Length of a telephone cable

This entails computing the length of a cable suspended from a sequence of telegraph posts over uneven terrain. We begin with the level ground problem, which entails the solution of a single nonlinear equation.

The shape of a cable hanging under its own weight between two poles at equal heights is described by $y = h_0 + \lambda \cosh(x/\lambda)$ and, if the distance, $L$, between the poles and the amount, $s$, the cable sags in the middle are known, then $\lambda \cosh(L/\lambda) = \lambda + s$. This equation can be solved for $\lambda$. The remaining parameter $h_0$ can then be computed from the known height at the midpoint.

The bisection method and Newton’s iteration are introduced to solve the equation. These also provide an easy introduction to MATLAB loops. The secant method is also studied here as a compromise in case the derivative is unavailable.

For uneven terrain the model has two parameters for each loop of cable because it is no longer symmetric. This results in a pair of nonlinear equations for each loop of the cable, which can be solved using Newton’s method for a pair of equations. Ensuring that the cable is continuous and that it always has a minimum clearance requires some post-processing. A requirement that a minimum clearance is achieved everywhere adds a further “post-processing” consideration.

The project actually seeks the total length of cable. This is a simple computation once the parameters of the curve are known. Its importance lies in the fact that just solving the equations is not the end of the story. The solution must be used to get the important information that would be required by an engineer bidding for the job.
of installing the cable.

The model parameter $\lambda$ is of course related to the density and tension in the cable. A full explanation of the physics is not included in the project. The project is worded in terms of a desired amount of “sag” to allow for extreme weather conditions. This is one of the points at which some team teaching including an engineer or physicist to describe the engineering aspects would be advantageous.

B. Rats in a maze

This project introduces the idea of iterative solution of linear equations. The project is based on the psychological learning experiment in which a rat is inserted in a rectangular mesh of passages with doors at each intersection. Food is placed at certain exits. “Success” is defined as the rat exiting the maze at a point at which it finds food. It is possible to determine whether the rats are learning by a statistical comparison of their success rate with the base level resulting from purely random decisions.

The objective of this project is to determine those base probabilities of success from each potential starting point. This naturally sets up as a Gauss-Seidel iteration.

The probability $P_{i,j}$ of success starting from a point with grid coordinates $(i, j)$ is just the average of the probabilities of success at the four immediate neighbors, since, for random moves, there is an equal probability of going next to each of these neighbors. At each internal intersection in the maze, we have

$$P_{i,j} = \frac{1}{4} \left( P_{i-1,j} + P_{i+1,j} + P_{i,j-1} + P_{i,j+1} \right)$$

This system is easily solved by iterative methods such as Gauss-Seidel, but is difficult to set up in a conventional matrix-vector formulation. (The difficulties arise from both the size of the system and taking careful account of the edges of the mesh.) For the iterative system, the edges are incorporated by fixing probabilities of 1 or 0, depending on whether the corresponding exit has food or not.
The project is then made more open-ended by adding diagonal passageways, adjusting the “random” decision to make sharper turns less likely than others, or by adding “trap-doors” in certain locations. The basic Gauss-Seidel approach remains appropriate but needs modifications varying from minor to quite substantial, when directional information is used.

This project is usually done in teams, which allows greater scope for students to pursue their own variations on the basic experiment. However the initial problem is assigned as a homework exercise so that each individual must first show the ability to solve the simple case.

C. Reproduce a picture

The directions for this project are simple. Choose a picture and reproduce it as a line drawing, using interpolation. The methods introduced are polynomial and cubic spline interpolation. The difficulties of polynomial interpolation become quickly apparent in this context!

Specifically, the students take measurements on their pictures relative to some axes of their choosing and then plot interpolation polynomials or splines to try to reproduce the original picture.

An appreciation for the problems of knot placement is quickly developed, as well as the basic ideas of interpolation. The distinction between curves with a “vertical” orientation, and the traditional “horizontal” $y = f(x)$ is also quickly understood.

This project is somewhat open-ended in that, students get involved in continually trying to improve their pictures with either more detail or smoother, more accurate reproduction of the curves.

Another aspect, which is often added by the students, is the use of cubic spline interpolation of parameterized curves. Some parts of a picture typically lend themselves to being plotted as a single curve, but neither as a function of $x$, nor one of $y$. This gives students a better insight into the basic ideas of computer aided design.

A pleasing outcome of this project this year is that one student elected to pursue her “capstone” writing project
studying exponential splines. The student observed the tendency for cubic splines to “wiggle”, and read about exponential splines in an *American Mathematical Monthly* article [1]. (This capstone project is a component of our Advanced Calculus course. It is usually on a pure mathematics topic.)

As background for this topic it would be helpful to bring in an engineer who works with splines in computer aided design, or a computer graphics expert to help with the motivation of this area of study. In this instance, a single guest lecture might suffice.

D. The Gamma function

Motivated by the gamma distribution of statistics (our mathematics majors have a course in applied statistics concurrent with this course) we are faced with computing a table of values of the gamma function and plotting the results. The motivation of this project can benefit significantly from a guest lecture by a statistician about the importance of the gamma distribution.

This is our venture into numerical integration. Simpson’s rule with its error formula is sufficient for the definite integrals involved – but we need to evaluate improper integrals. All have infinite range and most have discontinuity of important derivatives at 0.

For the infinite range of integration, the student has to find a way of bounding the tail and then using Simpson’s rule to obtain the required accuracy. The singularity at zero is severe for small values of the argument. A similar approach can be used to that of bounding the tail – but for even moderate accuracy, it is easy to show that the resulting definite integral would require so many nodes that the computation would take several *years* – a convincing argument that other approaches are needed!

The recurrence relation for the gamma function can be used to get round this problem. The students see the power of the combination of mathematical analysis of the problem and good algorithm design. Again, in this project the student faces the inadequacy of the basic method for the current objective.
One of the key objectives in teaching scientific computing is to overcome the “black box” mentality. For this it is important to show the limitations of the basic methods – and, of course, to see that they can be used to solve the problem with a little extra thought and care in the algorithm. When time allows, and for students who are interested, this project provides a good introduction to more advanced methods for infinite or singular integrals.

E. The human cannonball

This project began as a conventional two-dimensional projectile with quadratic air-resistance. The objective of hitting a target at a given position was renamed the human cannonball by students who fantasized about escaping from the course by being shot through a window in their fortress wall!

We begin with a review of the simple projectile equations in the absence of air resistance – the usual calculus or elementary physics model. The linear air resistance case is also introduced and solved. The equation for the launch angle must be solved numerically. This introduces the basic idea behind shooting methods – and serves to remind students of their earlier experience with such methods.

A more realistic quadratic air-resistance term is then included. The basic problem becomes a two-point boundary value problem in two-dimensions. This is converted readily to a system of four first-order differential equations, which can be solved numerically for given initial conditions. Single step methods are employed with the classical RK4 method being the technique of choice. The study of single step methods begins with Euler’s method and progresses to the Runge-Kutta methods. In each case some elementary problems are used to check the order of the truncation error for the different methods.

Converting the scalar (single first-order equation) MATLAB code to a system of differential equations is made especially simple by MATLAB’s vector notation. Therefore the step from a single differential equation to the system under consideration is not usually a stumbling block.

Shooting methods are used to solve the boundary-value problem. The resulting equation for the launch-angle is
then solved. Newton’s method cannot be used since we have no access to the derivative. The secant method is re-introduced and used – bringing us full circle to another application of the solution of a single (if complicated) equation.

The biggest difficulty most students have here is conceptual. The idea of creating a function whose value is the “amount by which the target is missed” always seems to cause some heartache. Essentially students are still unfamiliar with functions that are defined by a complicated algorithm, rather than an algebraic formula. Once this hurdle is cleared, this project goes smoothly.

Students often find ways to add to this project, too. Some try to come up with a firing table based on target locations – sometimes distinguishing between the high and low trajectories. Others experiment with the parameter used in the air resistance term. The initial value is determined from considerations of an arbitrary terminal velocity for free-fall in air.

The physical model for air resistance for different shaped bodies is an opportunity for team-teaching involving aeronautics engineers or physicists.

F. Projects in other courses

We conclude this section with a brief description of some of the scientific computing projects which have been used in other courses. In the computer calculus sequence, we are primarily concerned with projects that support the calculus curriculum. Topics covered include equation solving (and evaluation of implicit and inverse functions), elementary numerical integration including improper integrals and power series. The motivation is usually drawn from the calculus rather than practical application, although some practical motivation is always included for the topics themselves. In the more advanced courses, the projects have often been chosen to reflect student interests rather than a fixed syllabus.

In the Computer Arithmetic course, these have included simulation of hardware algorithms for binary fixed and
floating-point arithmetic, residue number systems and the level-index arithmetic systems.

Having the students program a simulation of floating-point addition or multiplication, for example, certainly guarantees a much clearer understanding of the issues than conventional lectures and tests. It also provides a sense of “ownership” which traditional courses lack.

The parallel scientific computing course introduces parallel architectures and then concentrates on basic linear algebra operations on a massively parallel computer and the solution of linear systems. The course usually culminates in a freely chosen project. A recent one concerned parallel computation of coefficients of a wavelet transform by using each processor to compute one component by Romberg integration. Others studied linear algebra on either very large systems or on smaller systems so that multiple copies of matrices can be stored on the processor array.

The most recent version of the advanced numerical analysis course was devoted to function approximation, including least squares, uniform approximation and rational approximation. An interesting overall project for this course was to produce a Greek alphabet font by combining the various techniques.

III. Assessment issues

As was previously mentioned, the introductory scientific computing course has no formal examinations. The assessment is based solely on the projects and associated homework assignments. The upper level classes are treated similarly. In the Computer Calculus course, formal examinations are included for the conventional calculus parts but the computational aspects are assessed entirely through projects and related homework. For this discussion, we concentrate on the junior level introductory course.

Several of the projects are done by teams (usually pairs) of students. Some are individual work. In particular project 3 on reproducing a picture using splines is an individual project. The associated homework is all (intended to be) individual.
There are both problems and advantages to teamwork. Most of the advantages are fairly obvious: encouraging teamwork itself, learning from each other, learning how to allocate the work and to use different people’s strengths are all worthwhile – perhaps especially so in the Naval Academy setting.

Some of the problems are associated with fairness. It is difficult to be certain that members of the team all deserve the same grade. In their own assessment, students almost invariably say that everyone contributed at a similar level. This is one reason for trying to use teams of two. It is much less likely that one will be a passenger in such a situation – especially if they choose their own partners, usually friends. Where possible, using teams whose members have different majors helps, too. They then can recognize each other’s different strengths and learn more from each other.

On rare occasions, when it appears that a student is being carried by his or her partner, it is desirable to enforce a change in the teams partway through. More frequent changes are certainly worth considering. The individual homework can often provide clues to problems, and help to separate members of a team to ensure fair grading.

Working with projects, it is important to demand more than just an answer, of course. The students are asked to produce a “Lab Report” summarizing the problem, the solution techniques, the results and their conclusions. This should be a formal report with everything in the same computer file. They may use whatever document preparation software they choose. Scientific Workplace is available to all in addition to their standard word processing packages.

One initial difficulty they often have is finding the right level of detail in explaining the methods without just reproducing course notes. The paradigm is that they should prepare the report in order to explain what is happening to a fellow student who is not taking the course, typically an engineering major. This model seems to encourage them to think about the overall process rather than getting into too much detail. It remains one of the main challenges to get the reports to present an appropriate balance between “big picture” and detail. Writing
these project reports provides excellent opportunity for the students to improve their technical writing and communication skills as well as their scientific skills.

Pacing projects so that every student has a reasonable chance is also a challenge in this style of teaching. One technique used is to require partial results at different times – usually by starting with a simple, special case of the project. This gives weaker students a decent chance to get started in good time. Instructor solutions to the simple cases can also be made available if necessary. I also allow revised (improved) projects to be resubmitted for improved grades.

Another aspect of pacing the projects is timing their due dates relative to the material. I have found it helpful to leave about a week for completion of a project after all the material is covered. Some of that week can be used for questions, while some of the material for the next project is also introduced.

IV. Student and faculty reaction

All students at USNA complete Student Opinion Forms on all courses. These vary among departments. Ours rely on written comments rather than checking boxes so it is difficult to analyze them in general.

However, there are certain very strong impressions from those for the project courses. I taught the introductory scientific computing course in traditional fashion up to four years ago. Since then it has been taught in the manner described.

Previously the course was not well received by most of the students. Many mathematics majors especially had difficulty seeing the need for such a course or the methods presented there. The failure to see the wood for the trees was a major difficulty.

The new course is much better received, with most students claiming to have enjoyed the experience. There are, of course, still some dissatisfied customers but most now seem to gain an appreciation both for why such material is important and for some of the intricacies of using numerical approximations.
The most common criticisms now focus on three aspects:

1. There is no standard text. (Course notes are supplemented by a text used as reference.)

2. Often students think the first project is the hardest, and ask that the projects be reordered. (I suspect that the first project is “hardest” because it is first, and they are still acclimatizing.)

3. Some would like the project problems to be more realistic. (Some team-teaching would help here.)

Last year (1998), for the first time, there was another faculty member teaching a section of this course. His reactions were entirely positive. He felt his students enjoyed the course, and he did, too. He had some reservation about how much was expected of the students. (He feared his demands were too light.) He judged the overall demand of writing and computing and mathematics to be sufficient however.

In the freshman computer calculus sequence, other faculty have been involved for many years. (The course was conceived by Ted Benac in the early 1970’s.) Projects were introduced over ten years ago to replace examinations on the computing part of the course. They have been universally well received by other faculty and have been strongly preferred by the students involved. All the side benefits mentioned for the scientific computing course apply there, too.

V. Extending the Idea: the Computation Science Major

In discussions with a member of our Computer Science department, we have been considering extending the ideas here to a Computation Science major. This would not be a computer science major, nor would it be a mathematics major. Rather, the ideas of learning through projects would be extended in specially designed courses which would be team-taught using faculty from both those departments and from engineering, science, and economics. The intention would be to take real problems in those areas, model them, solve them, examine the solution, refine the model, re-solve the problem, … repeating this iterative loop as necessary. Mathematical modeling, computer science, numerical analysis, applied mathematics and science would all combine in a
multidisciplinary whole.

One of the initial difficulties in mounting such a major is defining the discipline. Computation science is not computer science, nor is it just mathematics, nor is it just conventional science, and perhaps most importantly, nor is it an exact science. Of course it has aspects of all of these. Certainly it involves programming in some high-level language, typically nowadays a package such as MATLAB rather than the more traditional languages. Java programming is becoming ever more important for web-based packages and applications.

A basic understanding of computer architectures – especially advanced architectures and the principles of parallel computing – is important for the modern computation scientist. The computer science and mathematical backgrounds overlap in the need for a basic understanding of computer arithmetic and its effects on computational processes. The mathematical fundamentals will also include all the standard core mathematics found in good applied science programs: calculus, differential equations, statistics, linear algebra, etc.

The mathematical packages used will almost certainly include a symbolic package such as Maple as well as “traditional” numerical computing. The mathematical content must also include the analysis of the results of the computation to assess their validity, and to refine the process when necessary.

These aspects of computer science and mathematics would be of little value without mathematical modeling, which should be derived from science courses rather than mathematics courses.

But computation science is not just a careful selection of courses from these three disciplines. It is by its very nature interdisciplinary. Of course much of the core material could be taught in conventional courses. The basic mathematics and science, and computer architecture and programming courses would remain valuable initial courses.

Once this core is developed, projects should be used extensively in courses which have an interdisciplinary, team-taught nature. The development from practical problem, to mathematical model, to computational
algorithm, to numerical (or numeric-symbolic) computation is not a one-way street. At each stage of the process there should be feedback and checks to see whether the algorithm solves the mathematical model, whether this model really models the true situation and, whether the final results make scientific sense.

A typical course within this major should blend an actual scientific problem with a study of some possible mathematical and numerical techniques for its solution, including a discussion of convergence and error, and the associated computational implementation – and the (repeated) feedback loop just discussed.

It is important that throughout such a course, students get an appreciation for the experimental nature of computation science and do not expect to see closed form solutions to all problems.

A computation science major along these lines is likely to be expensive in resources initially, and a headache for university schedulers. I believe those are prices worth paying to produce graduates with a potential for immediate impact in a number of areas.

VI. Summary

This paper has described an approach to teaching scientific computing by the use of projects. Inevitably some of the theoretical mathematical aspects are reduced, but the understanding of the place of scientific computing in the scientific spectrum is much enhanced.

There are still problems, and new projects are often hard to find. There is significant burden on the instructor to provide class notes on the projects – perhaps a web-based repository of good projects could be established to help instructors in this area.

Extensive interdisciplinary team-teaching is a desirable future development both within conventional scientific computing courses, and, in a more ambitious extension of the ideas to a computation science major unifying some of the strengths of science and engineering, and mathematics and computer science.
References


Bio Sketch:

**Peter R Turner** has been a member of the Mathematics Department at the U S Naval Academy for 13 years. He obtained his PhD (in Pure Mathematics) from the University of Sheffield in 1973 and then was Lecturer in Numerical Analysis at the University of Lancaster, UK for 14 years. His primary research interests are in computer arithmetic and massively parallel computing. Peter Turner is author (or co-author) of some 40 refereed articles, as well as four undergraduate texts. He is a member of SIAM, the MAA, IEEE Computer Society and the ACM. His long time interest in computational science education has recently led to his proposal for a Center for Computational Science and Engineering at the naval Academy. Away from the office his interests focus on his family, his dogs, and the occasional dram of single malt scotch whisky!