1 Introduction

The goal of this course is to give a modern introduction to mathematical methods for solving hard mathematics problems that arise in the sciences — physical, biological and social. The toolbox of applied mathematics has changed dramatically over the past fifteen years.

There are two major factors that have contributed to this change. First, the dramatic increases in inexpensive computational speed have made large scale computation much more prevalent. Computers are now sufficiently fast that algorithms with minimal sophistication can perform once unthinkably large computations on a laptop PC. The consequence of this is a dramatic increase in numerical computations in the scientific literature; it is an understatement to say that most theoretical papers in the engineering sciences contain numerical computations. Even in the biological sciences it is becoming more and more fashionable to supplement traditional arguments with simulations of one kind or another.

The second major change in the toolbox of applied mathematics is the advent of fast, reliable and easy to use packages for routine numerical and symbolic computations (Matlab, Mathematica and Maple). These packages have cut the time for writing small scale computer codes dramatically, and likewise have dramatically increased the size and accuracy of analytic computations that can be carried out.

Additionally, they have, as it were, lowered the bar of required knowledge for carrying out numerical calculations. Armed with knowledge of how to run a computer package, it is possible to carry out numerical calculations solving for example a set of coupled highly nonlinear partial differential equations. Although the computer will readily spit out answers, the question then is what do these answers mean?

1.1 Computer Graphics and Mathematical Models

The most dramatic version of this question is to ask what is the difference between the numerical solution to a mathematical model, and a computer graphics animation of the same phenomenon. A computer graphics animation of fire aims to reproduce the salient features of the combustion process and visualize it so that it looks as realistic as possible. But a combustion scientist simulating this same fire does not care if the solution visually looks like fire: she is interested instead in whether the chemical and transport mechanisms are reliably represented, in order to refine understanding of why and how burning occurs. Such a scientist will be more interested in understanding for

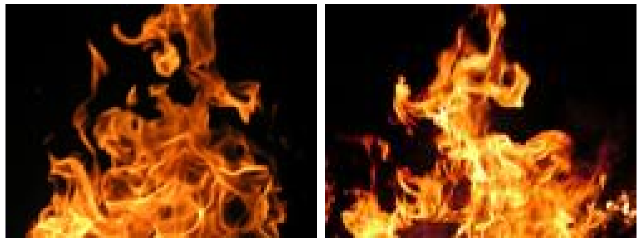

Figure 1.1. A real picture of fire (left, iStockphoto.com) compared with an animation of fire (right, quaife.us).

example the nature of the flame front — what is burning; which chemicals are leading to the color variation; what sets the characteristic scale of the flame, and of the small features in the flame — than in making a simulation that visibly looks like fire.

Somewhat remarkably, a computer graphicist trying to simulate fire might use exactly the same mathematical structure as the scientist, despite their completely different ends. Modern algorithms for computer graphics often solve nonlinear equations motivated by physics.

Another famous example comes from the wonderful movie Finding Nemo. The motions of the fishes in this movie greatly resemble those of real fishes. To achieve this the animators no doubt needed to learn and study fish physiology. However, the actual mechanisms of fish locomotion are quite different than those underlying the animation. Scientists who try to study the mechanisms for fish locomotion also carry out numerical simulations of the motion; instead of focusing on their visual appeal they instead try to create as faithful a representation of the motion as possible, in order to discover how it works.

Whereas a mathematical model aims to understand something animation aims to emulate it. The difference is easy to ascertain when going to the movies. But suppose in the course of your research into some phenomenon you write a computer program to simulate it. You worked hard on your simulation and are proud of it. Is your simulation computer graphics, or does it actually teach you something about the phenomenon in question. And how do you know?

1.2 Calculating while computing

The answer to this question depends both on the model that you have formulated, and the way that you have analyzed it. Creating good models is in itself a fascinating subject, but this is not the topic of the present course. The topic of this course is how to *analyze* the output of a computer simulation to *understand why the output is what it is.* If you solve a horribly complicated mathematics problem, whether a nonlinear partial differential equations, a set of coupled differential equations, the eigenvalues of a large matrix, etc. it is the contention of this course that you should nonetheless be able to understand in explicit terms why the solution is the way that it is. It may be difficult, it may require some approximation, but our contention is both that it is possible in general to do this, and moreover that without doing it you do not really understand what you have done.

Indeed, in recent years, an emerging trend is that while there is more and more interest in inventing algorithms for doing fast computation, or doing them on the computer architectures that have emerged, etc, there is correspondingly less interest in both learning and teaching analytical methods. After all, why should one learn how to carry out a difficult and possibly tedious approximate analytic calculation when there exist all purpose computer programs for solving all problems?

Our answer to this question is that you cannot understand the output of a computer simulation, however sophisticated it may seem, without some analysis to back up the calculation. You must convince yourself that the calculation is correct, and moreover you must understand its essential feature. It is not enough to say "Look, my computer simulation looks like the ocean." You need to explain why it looks like the ocean, which numbers you put into the computer mapped into which numbers characteristic of the ocean, etc. Without being able to do this you simply aren't doing good science and additionally you have little basis to explain why you believe the answer that the computer simulation has given.

Our aim therefore is to teach, within a broad suite of examples, how computer simulations and analytical calculations can be effectively combined. In this course, we will begin with problems that are simple–polynomial equations and first order differential equations – and slowly march our way towards the study nonlinear partial differential equations. We will show that a set of simple ideas provides a framework for developing an understanding all of these problems.

1.3 What is an analytical solution?

By and large, the analytic computations that we will emphasize in these notes are quite different than those that are usually taught in mathematical methods text books. Our focus will be on introducing methods which are *structurally stable*, in the sense that they work equally well when the mathematics problem is changed. This is opposite to the type of understanding that is usually taught. For example, in calculus, students are taught how to carry out a series of integrals which can be carried out exactly. For example, every calculus student learns that

$$\int \frac{1}{1+x^2}\, dx = \tan^{-1}(x), \tag{1.1}$$

so that

$$I_1 = \int_0^\infty \frac{1}{1+x^2}\, dx = \frac{\pi}{2}. \tag{1.2}$$

Calculus students pride themselves on learning these formulas; but in truth we must admit that they only have content if you happen to know how to compute $\tan^{-1}(x)$, which is after all defined by the integral!

To drive this point home, here is another example of an integral that has long been taught to (advanced) calculus students, the so-called *elliptic integral*.

$$E(x;k) = \int_0^x \frac{\sqrt{1-k^2t^2}}{\sqrt{1-t^2}}\, dt. \tag{1.3}$$

Probably many of you have never heard of the elliptic integral. But in many circles people still say that a problem has an analytical solution if it can be reduced to elliptic integrals!

Historically, this view of what it meant to derive an analytical solution to a problem was quite a reasonable one. Since computers for carrying out direct solutions to mathematics problems did not exist, the only way to solve a problem was to reduce it to a problem whose solution had already been tabulated numerically. Tables of functions (like the arctangent, the elliptic integral, the logarithm, and whatnot) were collected together–and indeed copies of these tables were included in the back of mathematics textbooks. For example, the classical two-volume series *Methods of Theoretical Physics*, Morse and Feshbach included tables of the following functions:

1. Trigonometric and Hyperbolic Functions
2. Hyperbolic tangent of complex numbers
3. Logarithms and Inverse Hyperbolic functions
4. Spherical Harmonic Functions
5. Legendre functions for large arguments
6. Legendre functions for imaginary arguments
7. Legendre functions of half integral degree
8. Bessel functions for cylindrical coordinates
9. hyperbolic bessel functions
10. spherical bessel functions
11. Legendre functions in spherical coordinates
12. Cylindrical bessel functions
13. Mathieu functions

In 1953 when this book was published, learning the properties of these functions and how to reduce an arbitrary problem you are confronted with to a form that one of these functions can be used was the critical skill required to do calculations. In contrast, most of you have never heard of these functions and will have no need for them at any point in your life. Today, computer calculations have completely replaced the use of tables of special functions and hence the analytical manipulations that accompany their use.

So what does it mean to develop an analytical understanding to a mathematics problem? Leaving aside its outdated pragmatic nature (and the fact that no human being can compute elliptic integrals in their head), the main problem with the historical point of view is that it obscures *why* a value of an integral has the value that it does. Remembering the name of a function does not give any intuition about its properties — why it behaves one way and not another.

For example, what if we perturb our nice arctangent integral to be:

$$I_2(a) = \int_0^{100} \frac{1}{1 + x^2 + ax^7}\ dx \tag{1.4}$$

There is no special function with this name. Thus within the current system of mathematics education there is only one way to solve it — with a computer. A simple computation with Matlab yields that $I_2(a = 0.1) \approx 1.03$. Although this is a very efficient way to solve the problem it does not lead to any real understanding of why the integral has the number that it does. But on the other hand, we note that $I_2(a = 0.1)$ is close to $\pi/2$ (≈ 1.57). Is it an accident or is it a coincidence? Under what circumstances can the integral have a much different value? For example, examining the difference between I_2 and I_1, it seems evident that $I_2(a = 0)$ should be very close to I_1. Indeed, a numerical computation shows that $I_2(0) - \pi/2 \approx -0.01$. What determines the magnitude of this difference? What determines the rate at which $I_2(a) \to I_1$ as $a \to 0$?

Maybe you would like to name it after yourself!

Moreover how do you know that the answer is correct?

The goal of this course is to develop methods and ideas for answering this type of question in the context of the variety of different problems that arise in applications. The ideas we will discuss are weighted roughly equally between learning the mathematics and learning how to use a computer to expose and discover the mathematics. Whereas in 1953 students asking the question "how does $I_2(a)$ depend on a?" were forced to rely solely on their wits, today computers can be used to help prod one's wits into understanding a problem.

It is our belief that developing skill in thinking about mathematics this way is crucial for educating modern students in applying mathematical and numerical methods to the sciences. Despite the relative ease of producing plausible answers to hard problems, learning numerical computation by itself is not enough. First, without having any understanding of *why* a problem has the answer that it does, one does not understand how the answer will change when the problem changes. Simply producing a graph with numbers to be compared to a phenomenon does not lead to any understanding of why the phenomenon behaves as it does. Second, without having an understanding of the answer, it is extremely difficult to determine when the numerical results are erroneous. A numerical method can be erroneous for two different reasons: either the numerical method can not solve the intended equation accurately enough, *or* the equation itself could be inaccurate, due to unjustified approximation. We will illustrate herein that both of these situations can be quite subtle; only with careful understanding can it be debugged and understood. Finally, an understanding of why the answer has the value that it does allows one to design numerical algorithms for much more difficult problems than would be possible if such an understanding did not exist.