Jim Lambers
MAT 460
Fall Semester 2009-10
Lecture 1 Notes

Introduction

This course is the first in a two-course sequence on *numerical analysis*, the study of numerical algorithms for solving mathematical problems that arise in science and engineering. These numerical algorithms differ from the analytical methods that are presented in other mathematics courses in that they rely exclusively on the four basic arithmetic operations, addition, subtraction, multiplication and division, so that they can be implemented on a computer.

Error Analysis

First, we will discuss the most fundamental concepts of numerical analysis, which pertain to *error analysis*. Numerical algorithms must not only be *efficient*, but they must also be *accurate*, and *robust*. In other words, the solutions they produce are at best *approximate* solutions because an exact solution cannot be computed by analytical techniques. Furthermore, these computed solutions should not be too sensitive to the input data, because if they are, any error in the input can result in a solution that is essentially useless. Such error can arise from many sources, such as

- neglecting components of a mathematial model or making simplifying assumptions in the model,
- discretization error, which arises from approximating continuous functions by sets of discrete data points,
- convergence error, which arises from truncating a sequence of approximations that is meant to converge to the exact solution, to make computation possible, and
- roundoff error, which is due to the fact that computers represent real numbers approximately, in a fixed amount of storage in memory.

We will see that in some cases, these errors can be surprisingly large, so one must be careful when designing and implementing numerical algorithms.

Nonlinear Equations

The vast majority of equations, especially nonlinear equations, cannot be solved using analytical techniques such as algebraic manipulations or knowledge of trignometric functions. Therefore, *iterative* methods must instead be used to obtain an approximate solution. We will study a variety of such methods, which have distinct advantages and disadvantages. For example, some methods are guaranteed to produce a solution under reasonable assumptions, but they might do so slowly.

On the other hand, other methods may produce a sequence of iterates that quickly converges to the solution, but may be unreliable for some problems.

Polynomial Interpolation and Approximation

Polynomials are among the easiest functions to work with, because it is possible to evaluate them, as well as perform operations from calculus on them, with great efficiency. For this reason, more complicated functions, or functions that are represented only by values on a discrete set of points in their domain, are often approximated by polynomials.

Such an approximation can be computed in various ways. We first consider *interpolation*, in which we construct a polynomial that agrees with the given data at selected points. While interpolation methods are efficient, they must be used carefully, because it is not necessarily true that a polynomial that agrees with a given function at certain points is a good approximation to the function elsewhere in its domain.

One remedy for this is to use *piecewise* polynomial interpolation, in which a low-degree polynomial, typically linear or cubic, is used to approximate data only on a given subdomain, and these polynomial "pieces" are "glued" together to obtain a piecewise polynomial approximation. This approach is also efficient, and tends to be more robust than standard polynomial interpolation, but there are disadvantages, such as the fact that a piecewise polynomial only has very few derivatives.

Numerical Differentiation and Integration

It is often necessary to approximate derivatives or integrals of functions that are represented only by values at a discrete set of points, thus making differentiation or integration rules impossible to use directly. Numerical techniques for these operations make use of polynomial interpolation by (implicitly) constructing a polynomial interpolant that fits the given data, and then applying differentiation or integration rules to the polynomial. We will see that by choosing the method of polynomial approximation judiciously, accurate results can be obtain with far greater accuracy and efficiency than one might expect.

Approximation Theory

Finally, we study the approximation of functions by "simpler" functions such as polynomials and trigonometric polynomials (i.e., sines and cosines). This differs from interpolation, in that the approximation is not intended to agree with the approximated function at any particular point, but rather should be "close", in some sense, to the original function on a given domain. To construct such approximations efficiently, we make use of *orthogonal polynomials*, which facilitate the modification of approximating polynomials after the addition or exclusion of data. We conclude the course with approximation by trigonometric polynomials, which leads to the Fast Fourier Transform, a tool for signal processing that is, without doubt, one of the most influential algorithms in all of scientific computing.