Jim Lambers

MAT 460/560

Fall Semester 2009-10

Lecture 6 Notes

These notes correspond to Section 1.3 in the text.

Approximations in Numerical Analysis

Mathematical problems arising from scientific applications present a wide variety of difficulties that prevent us from solving them exactly. This has led to an equally wide variety of techniques for computing approximations to quantities occurring in such problems in order to obtain approximate solutions. In this lecture, we will describe the types of approximations that can be made, and learn some basic techniques for analyzing the accuracy of these approximations.

Sources of Approximation

Suppose that we are attempting to solve a particular instance of a problem arising from a mathematical model of a scientific application. We say that such a problem is *well-posed* if it meets the following criteria:

- The problem has a unique solution.
- A small perturbation in the problem data results in a small perturbation in the solution; i.e., the solution *depends continuously* on the data.

By the first condition, the process of solving a well-posed problem can be seen to be equivalent to the evaluation of some function f at some known value x, where x represents the problem data. Since, in many cases, knowledge of the function f is limited, the task of computing $f(x)$ can be viewed, at least conceptually, as the execution of some (possibly infinite) sequence of steps that solves the underlying problem for the data x. The goal in numerical analysis is to develop a finite sequence of steps, i.e., an algorithm, for computing an approximation to the value $f(x)$.

There are two general types of error that occur in the process of computing this approximation to $f(x)$:

1. *data error* is the error in the data x. In reality, numerical analysis involves solving a problem with *approximate* data $\hat{x}$. The exact data is often unavailable because it must be obtained by measurements or other computations that fail to be exact due to limited precision. In addition, data may be altered in order to simplify the solution process.

2. *computational error* refers to the error that occurs when attempting to compute $f(\hat{x})$. Effectively, we must approximate $f(\hat{x})$ by the quantity $\hat{f}(\hat{x})$, where $\hat{f}$ is a function that approximates f. This approximation may be the result of *truncation*, which occurs when it is not possible to evaluate f exactly using a finite sequence of steps, and therefore a finite sequence that evaluates f approximately must be used instead. This particular source of computational error will be discussed in this lecture. Another source of computational error is roundoff error, which was discussed in the previous lecture.

Basics of Error Analysis

Intuitively, it is not difficult to conclude that any scientific computation can include several approximations, each of which introduces error in the computed solution. Therefore, it is necessary to understand the effects of these approximations on accuracy. The study of these effects is known as *error analysis.*

Error analysis will be a recurring theme in this course. In this lecture, we will introduce some basic concepts that will play a role in error analyses of specific algorithms in later lectures.

Forward Error and Backward Error

Suppose that we compute an approximation $\hat{y} = \hat{f}(x)$ of the value $y = f(x)$ for a given function f and given problem data x. Before we can analyze the accuracy of this approximation, we must have a precisely defined notion of error in such an approximation. We now provide this precise definition.

Definition *(Forward Error) Let x be a real number and let $f : \mathbb{R} \to \mathbb{R}$ be a function. If $\hat{y}$ is a real number that is an approximation to $y = f(x)$, then the* **forward error** *in $\hat{y}$ is the difference $\Delta y = \hat{y} - y$. If $y \neq 0$, then the* **relative forward error** *in $\hat{y}$ is defined by*

$$\frac{\Delta y}{y} = \frac{\hat{y} - y}{y}.$$

Clearly, our primary goal in error analysis is to obtain an estimate of the forward error Δy. Unfortunately, it can be difficult to obtain this estimate directly.

An alternative approach is to instead view the computed value $\hat{y}$ as the *exact* solution of a problem with modified data; i.e., $\hat{y} = f(\hat{x})$ where $\hat{x}$ is a perturbation of x.

Definition *(Backward Error) Let x be a real number and let $f : \mathbb{R} \to \mathbb{R}$ be a function. Suppose that the real number $\hat{y}$ is an approximation to $y = f(x)$, and that $\hat{y}$ is in the range of f; that is, $\hat{y} = f(\hat{x})$ for some real number $\hat{x}$. Then, the quantity $\Delta x = \hat{x} - x$ is the* **backward error** *in $\hat{y}$. If $x \neq 0$, then the* **relative forward error** *in $\hat{y}$ is defined by*

$$\frac{\Delta x}{x} = \frac{\hat{x} - x}{x}.$$

The process of estimating Δx is known as *backward error analysis.* As we will see, this estimate of the backward error, in conjunction with knowledge of f, can be used to estimate the forward error.

As discussed in the previous lecture, floating-point arithmetic does not follow the laws of real arithmetic. This tends to make forward error analysis difficult. In backward error analysis, however, real arithmetic is employed, since it is assumed that the computed result is the exact solution to a modified problem. This is one reason why backward error analysis is sometimes preferred.

In analysis of roundoff error, it is assumed that $\mathrm{fl}(x \;\texttt{op}\; y) = (x \;\texttt{op}\; y)(1+\delta)$, where $\texttt{op}$ is an arithmetic operation and δ is an unknown constant satisfying $|\delta| \leq \epsilon_{\text{mach}}$. From this assumption, it can be seen that the relative error in $\mathrm{fl}(x \;\texttt{op}\; y)$ is $|\delta|$. In the case of addition, the relative backward error in each operand is also $|\delta|$.

Sensitivity and Conditioning

In most cases, the goal of error analysis is to obtain an estimate of the forward relative error $(f(\hat{x}) - f(x))/f(x)$, but it is often easier to instead estimate the relative backward error $(\hat{x} - x)/x$. Therefore, it is necessary to be able to estimate the forward error in terms of the backward error. The following definition addresses this need.

Definition *(Condition Number) Let x be a real number and let $f : \mathbb{R} \to \mathbb{R}$ be a function. The* **absolute condition number***, denoted by κ_{abs}, is the ratio of the magnitude of the forward error to the magnitude of the backward error,*

$$\kappa_{abs} = \frac{|f(\hat{x}) - f(x)|}{|\hat{x} - x|}.$$

If $f(x) \neq 0$, then the **relative condition number** *of the problem of computing $y = f(x)$, denoted by κ_{rel}, is the ratio of the magnitude of the relative forward error to the magnitude of the relative backward error,*

$$\kappa_{rel} = \frac{|(f(\hat{x}) - f(x))/f(x)|}{|(\hat{x} - x)/x|} = \frac{|\Delta y/y|}{|\Delta x/x|}.$$

Intuitively, either condition number is a measure of the change in the solution due to a change in the data. Since the relative condition number tends to be a more reliable measure of this change, it is sometimes referred to as simply the *condition number.*

If the condition number is large, e.g. much greater than 1, then a small change in the data can cause a disproportionately large change in the solution, and the problem is said to be *ill-conditioned* or *sensitive.* If the condition number is small, then the problem is said to be *well-conditioned* or *insensitive.*

Since the condition number, as defined above, depends on knowledge of the exact solution $f(x)$, it is necessary to estimate the condition number in order to estimate the relative forward error. To that end, we assume, for simplicity, that $f : \mathbb{R} \to \mathbb{R}$ is differentiable and obtain

$$\begin{aligned}
\kappa_{rel} &= \frac{|x\Delta y|}{|y\Delta x|} \\
&= \frac{|x(f(x+\Delta x) - f(x))|}{|f(x)\Delta x|} \\
&\approx \frac{|xf'(x)\Delta x|}{|f(x)\Delta x|} \\
&\approx \left|\frac{xf'(x)}{f(x)}\right|.
\end{aligned}$$

Therefore, if we can estimate the backward error Δx, and if we can bound f and f' near x, we can then bound the condition number and obtain an estimate of the relative forward error.

Of course, the condition number is undefined if the exact value $f(x)$ is zero. In this case, we can instead use the absolute condition number. Using the same approach as before, the absolute condition number can be estimated using the derivative of f. Specifically, we have $\kappa_{abs} \approx |f'(x)|$.

Conditioning, Stability and Accuracy

Determining the condition, or sensitivity, of a problem is an important task in the error analysis of an algorithm designed to solve the problem, but it does not provide sufficient information to determine whether an algorithm will yield an accurate approximate solution.

Recall that the condition number of a function f depends on, among other things, the absolute forward error $f(\hat{x}) - f(x)$. However, an algorithm for evaluating $f(x)$ actually evaluates a function $\hat{f}$ that approximates f, producing an approximation $\hat{y} = \hat{f}(x)$ to the exact solution $y = f(x)$. In our definition of backward error, we have assumed that $\hat{f}(x) = f(\hat{x})$ for some $\hat{x}$ that is close to x; i.e., our approximate solution to the original problem is the exact solution to a "nearby" problem.

This assumption has allowed us to define the condition number of f independently of any approximation $\hat{f}$. This independence is necessary, because the sensitivity of a problem depends solely on the problem itself and not any algorithm that may be used to approximately solve it.

Is it always reasonable to assume that any approximate solution is the exact solution to a nearby problem? Unfortunately, it is not. It is possible that an algorithm that yields an accurate approximation for given data may be unreasonably sensitive to perturbations in that data. This leads to the concept of a *stable algorithm*: an algorithm applied to a given problem with given data x is said to be *stable* if it computes an approximate solution that is the exact solution to the same problem with data $\hat{x}$, where $\hat{x}$ is a small perturbation of x.

It can be shown that if a problem is well-conditioned, and if we have a stable algorithm for solving it, then the computed solution can be considered accurate, in the sense that the relative error in the computed solution is small. On the other hand, a stable algorithm applied to an ill-conditioned problem cannot be expected to produce an accurate solution.