INTRODUCTION TO NUMERICAL ANALYSIS WITH C PROGRAMS

Attila M´at´e
Brooklyn College of the City University of New York

INTRODUCTION TO NUMERICAL ANALYSIS WITH C PROGRAMS

Attila Máté

Brooklyn College of the City University of New York

July 2004

Last Revised: August 2014

CONTENTS

PREFACE

These notes started as a set of handouts to the students while teaching a course on introductory numerical analysis in the fall of 2003 at Brooklyn College of the City University of New York. The notes rely on my experience of going back over 25 years of teaching this course. Many of the methods are illustrated by complete C programs, including instructions how to compile these programs in a Linux environment. These programs can be found at

`http://www.sci.brooklyn.cuny.edu/~mate/nml_progs/numanal_progs.tar.gz`

They do run, but many of them are works in progress, and may need improvements. While the programs are original, they benefited from studying computer implementations of numerical methods in various sources, such as [AH], [CB], and [PTVF]. In particular, we heavily relied on the array-handling techniques in C described, and placed in the public domain, by [PTVF]. In many cases, there are only a limited number of ways certain numerical methods can be efficiently programmed; nevertheless, we believe we did not violate anybody's copyright (such belief is also expressed in [PTVF, p. xvi] by the authors about their work).

New York, New York, July 2004
Last Revised: August 25, 2014

Attila Máté

1. FLOATING POINT NUMBERS

A floating point number in binary arithmetic is a number of form $2^e \cdot 0.m$, where e and m are integers written in binary (that is, base 2) notation, and the dot in $0.m$ is the binary "decimal" point, that is, if the digits of m are $m_1, m_1, \ldots, m_k$ ($m_i = 0$ or 1), then $0.m = m_1 \cdot 2^{-1} + m_1 \cdot 2^{-2} + \ldots m_1 \cdot 2^{-k}$. Here e is called the exponent and m is called the mantissa. In the standard form of a floating point number it is assumed that $m_1 = 1$ unless all digits of m are 0; a nonzero number in nonstandard form can usually be brought into standard form by lowering the exponent and shifting the mantissa (the only time this cannot be done is when this would bring the exponent e below its lower limit – see next). In the IEEE standard,[1] the mantissa m of a (single precision) floating point is 23 bits. For the exponent e, we have $-126 \le e \le 127$. To store the exponent, 8 bits are needed. One more bit is needed to store the sign, so altogether 40 bits are needed to store a single precision floating point number. Thus a single precision floating point number roughly corresponds to a decimal number having seven significant digits. Many programming languages define double precision numbers and even long double precision numbers.

As a general rule, when an arithmetic operation is performed the number of significant digits is the same as those in the operands (assuming that both operands have the same number of significant digits). An important exception to this rule is when two numbers of about the same magnitude are subtracted. For example, if

$$x = .7235523 \qquad \text{and} \qquad y = .7235291,$$

both having seven significant digits, then the difference

$$x - y = .0000232$$

has only three significant digits. This phenomenon is called loss of precision. Whenever possible, one must avoid this kind of loss of precision.

When evaluating an algebraic expression, this is often possible by rewriting the expression. For example, when solving the quadratic equation

$$x^2 - 300x + 1 = 0,$$

the quadratic formula gives two solutions:

$$x = \frac{300 + \sqrt{89996}}{2} \qquad \text{and} \qquad x = \frac{300 - \sqrt{89996}}{2}.$$

There is no problem with calculating the first root, but with the second root there is clearly a loss of

Problems

1. Calculate $x - \sqrt{x^2-1}$ for $x = 256,000$ with 10 significant digit accuracy. Avoid loss of significant digits.

Solution. We cannot use the expression given directly, since x and $\sqrt{x^2-1}$ are too close, and their subtraction will result in a loss of precision. To avoid this, note that

$$x - \sqrt{x^2-1} = \left(x - \sqrt{x^2-1}\right) \cdot \frac{x + \sqrt{x^2-1}}{x + \sqrt{x^2-1}} = \frac{1}{x + \sqrt{x^2-1}}.$$

To do the numerical calculation, it is easiest to first write that $x = y \cdot 10^5$, where $y = 2.56$. Then

$$\frac{1}{x + \sqrt{x^2-1}} = \frac{1}{y + \sqrt{y^2 - 10^{-10}}} \cdot 10^{-5} \approx 1.953,125,000,0 \cdot 10^{-6}.$$

2. Calculate $\sqrt{x^2+1} - x$ for $x = 1,000,000$ with 6 significant digit accuracy. Avoid the loss of significant digits.

Solution. We cannot use the expression given directly, since $\sqrt{x^2+1}$ and x are too close, and their subtraction will result in a loss of precision. To avoid this, note that

$$\sqrt{x^2+1} - x = \left(\sqrt{x^2+1} - x\right) \cdot \frac{\sqrt{x^2+1} + x}{\sqrt{x^2+1} + x} = \frac{1}{\sqrt{x^2+1} + x}.$$

To do the numerical calculation, it is easiest to first write that $x = y \cdot 10^6$, where $y = 1$. Then

$$\frac{1}{\sqrt{x^2+1} + x} = \frac{1}{\sqrt{y^2 + 10^{-12}} + y} \cdot 10^{-6} \approx 5.000,000,000 \cdot 10^{-7}.$$

3. Show how to avoid the loss of significance when solving the equation

$$x^2 - 1000x - 1 = 0.$$

4. Evaluate $e^{0.0002} - 1$ on your calculator. Hint: The best way to avoid the loss of precision is to use the Taylor series approximation of e^x. Using only two terms and the Lagrange remainder term

(Note that angles are measured in radians.) Hint: Rather than evaluating the sines of the angles given, it may be better to use the formula

$$\sin x - \sin y = 2\cos\frac{x+y}{2}\sin\frac{x-y}{2}$$

with $x = 0.245735$ and $y = 0.245712$. Then sine of a small angle may be more precisely evaluated using a Taylor series than using the sine function on your calculator. Note, however, that this approach does not avoid the loss of precision that results from calculating $x - y$. From what is given, this cannot be avoided.

6. Find $1 - \cos 0.008$ with 10 decimal accuracy.

Solution. Of course 0.008 means radians here. Using the value of $\cos 0.008$ here would lead to unacceptable loss of precision, since to value is too close to 1. Using the Taylor series of $\cos x$ gives a more accurate result:

$$\cos x = \sum_{n=0}^{\infty}(-1)^n \cdot \frac{x^{2n}}{(2n)!} = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots$$

For $|x| \le 1$ this is an alternating series, and so, when summing finitely many terms of the series, the absolute value of the error error will be less than that of the first omitted term. With $x = 0.008$, we have

$$\frac{x^6}{6!} < \frac{0.01^6}{720} = 10^{-12} \cdot \frac{1}{720} < 10^{-12} \cdot .00139 < 1.39 \cdot 10^{-15},$$

and so this term can safely be omitted. Thus, writing $x = 8 \cdot 10^{-3}$, with sufficient precision we have

$$1 - \cos x \approx x^2/2! - x^4/4! = 32 \cdot 10^{-6} - \frac{512}{3} \cdot 10^{-12} \approx 0.0000319998293$$

7. Find $1 - e^{-0.00003}$ with 10 decimal digit accuracy.

Solution. Using the value of $e^{-0.00003}$ would lead to an unnecessary and unacceptable loss of accuracy. It is much better to use the Taylor series of e^x with $x = -3 \cdot 10^{-5}$:

$$1 - e^x = 1 - \sum_{n=0}^{\infty}\frac{x^n}{n!} = -x - \frac{x^2}{2} - \frac{x^3}{6} - \dots .$$

For $x = -3 \cdot 10^{-5}$ this becomes an alternating series:

$$3 \cdot 10^{-5} - \frac{9 \cdot 10^{-10}}{2} + \frac{27 \cdot 10^{-15}}{6} - \dots .$$

When summing finitely many terms of an alternating series, the error will be smaller than the first omitted term. Since we allow an error no larger than $5 \cdot 10^{-11}$, the third term here can be safely omitted. Thus,

$$1 - e^{-0.00003} \approx 3 \cdot 10^{-5} - \frac{9 \cdot 10^{-10}}{2} = .000,029,999,55.$$

8. Calculate

$$\sum_{n=1}^{9999} \frac{1}{n^2}$$