Slides on Numerical Analysis

Chapter 1

Mathematical Preliminaries

1.1 Review of Calculus

In this section, we will review some important concepts of calculus.

- Limit of a function

Definition 1.1 Let f be a function defined on a set X of real numbers. Then f has the **limit L** at x_0, written

$$\lim_{x \to x_0} f(x) = L,$$

if, given any real number $\varepsilon > 0$, there exists a real number $\delta > 0$ such that

$$|f(x) - L| < \varepsilon$$

whenever $x \in X$ and $0 < |x - x_0| < \delta$.

Definition 1.2 Let f be a function defined on a set X of real numbers and $x_0 \in X$. Then f is **continuous** at x_0 if

$$\lim_{x \to x_0} f(x) = f(x_0).$$

The function f is continuous on the set X if it is continuous at each number in X.

Especially, let $C(X)$ denote the set of all functions that are continuous on the set X. When X is an closed interval $[a, b]$, the set of all functions that are continuous on the interval $[a, b]$ is denoted by $C[a, b]$.

♦ Limit of a sequence

Definition 1.3 Let $\{x_n\}_{n=1}^{\infty}$ be an infinite sequence of real or complex number. The sequence converges to a number x (Limit) if, for any $\varepsilon > 0$, there exists a positive integer $N(\varepsilon)$, such that implies

$$|x_n - x| < \varepsilon,$$

whenever $n > N(\varepsilon)$.

Noted by

$$\lim_{n\to\infty} x_n = x,$$

or $x_n \to x$ as $n \to \infty$.

Theorem 1.4 If f is a function defined on a set of real numbers and $x_0 \in X$, then the following statements are equivalent:

a. f is continuous at x_0;

b. if $\{x_n\}_{n=1}^{\infty}$ is any sequence in X converging to x_0, then

$$\lim_{n\to\infty} f(x_n) = f(x_0).$$

Continuity: the minimal requirement for predictable behavior.

Definition 1.5 If f is a function defined in an open interval containing x_0, then f is differentiable at x_0, if

$$f'(x_0) = \lim_{x \to x_0} \frac{f(x) - f(x_0)}{x - x_0}$$

exists.

The number $f'(x_0)$ ia called the **derivative** of $f(x)$ at x_0.

Notes: Let $C^n(X)$ denote the set of all functions that have n continuous derivatives on X, and the set of all functions that have derivatives of all orders on X is denoted by $C^\infty(X)$.

Some Important Theorems:

Theorem 1.6 If the function f is differentiable at x_0, then f is continuous at x_0.

Theorem 1.7 (Rolle's Theorem) Suppose $f \in C[a, b]$ and f is differentiable on (a, b). If $f(a) = f(b)$, then a number c in (a, b) exists with $f'(c) = 0$.

Theorem 1.8 (Mean Value Theorem) Suppose $f \in C[a, b]$ and f is differentiable on (a, b), then a number c in (a, b) exists with

$$f'(c) = \frac{f(b) - f(a)}{b - a}.$$

Theorem 1.9 (Extreme Value Theorem) If $f \in C[a, b]$, then $c_1, c_2 \in [a, b]$ exist with $f(c_1) \leq f(x) \leq f(c_2)$ for each $x \in [a, b]$. If, in addition, f is differentiable on (a, b), then the numbers c_1 and c_2 occur either at the endpoints of $[a, b]$ or where f' is zero.

Definition 1.10 **Riemann Integral** of a function on an interval $[a, b]$ is the following limit, provided it exists:

$$\int_a^b f(x)dx = \lim_{\max \Delta x_i \to 0} \sum_{i=1}^{n} f(z_i)\Delta x_i,$$

where the numbers $x_0, x_1, x_2, \cdots, x_n$ satisfy $a = x_0 \leq x_1 \leq x_2 \leq \cdots \leq x_n = b$, and where $\Delta x_i = x_i - x_{i-1}$ for each $i = 1, 2, \cdots, n$ and z_i is an arbitrarily chosen point in the interval $[x_{i-1}, x_i]$.

Especially, if we choose $z_i = x_i$ and $\Delta x_i = (b-a)/n$, then in this case

$$\int_a^b f(x)dx = \lim_{\max \Delta x_i \to 0} \frac{b-a}{n} \sum_{i=1}^{n} f(x_i),$$

Theorem 1.11 (Weighted Mean Value Theorem for the Integral) If $f \in C[a,b]$, the Riemann Integral of g exists on the $[a,b]$, and $g(x)$ does not change sign on $[a,b]$, then there exists a number c in (a,b) with

$$\int_a^b f(x)g(x)dx = f(c)\int_a^b g(x)dx.$$

When $g(x) \equiv 1$, this theorem give the **average value** of the function f over the interval $[a,b]$.

$$f(c) = \frac{1}{b-a}\int_a^b f(x)dx.$$

Theorem 1.12 (Generalized Rolle's Theorem) Suppose $f \in C[a, b]$ is n times differentiable on (a, b). If $f(x)$ is zero at the $n + 1$ distinct numbers $x_0, x_1, x_2, \cdots, x_n$ in the $[a, b]$, then a number c in the (a, b) exists with

$$f^{(n)}(c) = 0.$$

Theorem 1.13 (Intermediate Value Theorem) If $f \in C[a, b]$ and K is any number between $f(a)$ and $f(b)$, then there exists a number c in (a, b) for which $f(c) = K$.

Theorem 1.14 (Taylor's Theorem) Suppose $f \in C^n[a,b]$, that $f^{(n+1)}$ exists on $[a,b]$, and $x_0 \in [a,b]$. For every $x \in [a,b]$ there exists a number $\xi(x)$ between x_0 and x with $f(x) = P_n(x) + R_n(x)$. where

$$\begin{aligned} P_n(x) &= f(x_0) + f'(x_0)(x-x_0) + \frac{f''(x_0)}{2!}(x-x_0)^2 \\ &+ \cdots + \frac{f^n(x_0)}{n!}(x-x_0)^n \\ &= \sum_0^n \frac{f^{(k)}(x_0)}{k!}(x-x_0)^k, \end{aligned}$$

and

$$R_n(x) = \frac{f^{(n+1)}(\xi(x))}{(n+1)!}(x-x_0)^{n+1}.$$

here $P_n(x)$ is called the n**th Taylor polynomial** for f about x_0, and $R_n(x)$ is called the **remainder term (truncation error)** associated with $P_n(x)$. The infinite series obtained by taking the limit of $P_n(x)$ as $n \to \infty$ is called the **Taylor series** for f about x_0. In the case $x_0 = 0$, the Taylor polynomial is often called a **Maclaurin polynomial**, and the Taylor series is called a **maclaurin series**.

See $(\sqrt{3})^2 = 3$.

This is true in traditional arithmetic in algebra or calculus, but if we use calculator or computer to do, what will happen?

In our traditional mathematical world, we permit number with an infinite number of digits;

We define $\sqrt{3}$ as an unique positive number, so when it is multiplied by itself, we can get 3.

But in computer computation, $\sqrt{3}$ first is represented with a fixed, finite number of digits, which may be very closed to its exact value. This means only rational (有理数) number can be presented exactly.

Since $\sqrt{3}$ is not rational, it's square will not be precisely 3, although it will likely be sufficiently close to 3.

Roundoff error (舍入误差):

For computer storage, one standard is made by IEEE, which is Binary Floating Arithmetic Standard 754-1985.

Format: single, double, or extended precision

64-bit(binary digit) representation for a real number:

Representation: The first bit is a sign indicator, denoted s, This is followed by an 11-bit exponent(指数), c, called the characteristic, and a 52-bit binary fraction, f, call the mantissa(尾数).The base for the exponent is 2.

Using this system, a floating-point number can be shown with the form:

$$(-1)^s 2^{c-1023}(1+f)$$

Consider the machine number

$$0\ 10000000011\ 1011100010001\underbrace{00\cdots00}$$

s=0

$c = 1\cdot 2^{10}+0\cdot 2^{9}+\cdots+0\cdot 2^{2}+1\cdot 2^{1}+1\cdot 2^{0} = 1024+2+1 = 1027$

$$f = 1\cdot\left(\frac{1}{2}\right)^{1}+1\cdot\left(\frac{1}{2}\right)^{3}+1\cdot\left(\frac{1}{2}\right)^{4}+1\cdot\left(\frac{1}{2}\right)^{5}+1\cdot\left(\frac{1}{2}\right)^{8}+1\cdot\left(\frac{1}{2}\right)^{12}$$

So the machine number precisely represents the decimal number(十进制数)

$$(-1)^{s}2^{c-1023}(1+f) = (-1)^{0}\cdot 2^{1027-1023}\left(1+\frac{1}{2}+\frac{1}{8}+\frac{1}{16}+\frac{1}{32}+\frac{1}{256}+\frac{1}{4096}\right) = 27.56640625.$$

However, this original machine number represents not only 27.56640625, but also any real number in an interval. Since the next smallest machine number is

$$0\ 10000000011\ 101110010000\underbrace{11\cdots11}$$

and the next largest machine number is

$$0\ 10000000011\ 101110010001\underbrace{00\cdots01}$$

The smallest normalized positive number that can be represented is

$$0\underbrace{00\cdots01} \rightarrow 2^{-1023}\cdot(1+2^{-52}) \approx 10^{-308},$$

and the largest one can be represented is

$$0\underbrace{11\cdots11} \rightarrow 2^{1024}\cdot(2-2^{-52}) \approx 10^{308}.$$

underflow (下溢): number less than $2^{-1023} \cdot (1 + 2^{-52})$; cause to zero.

overflow (上溢): number greater than $2^{1024} \cdot (2 - 2^{-52})$, cause to halt.

Normalized decimal floating-point form:

$$\pm 0.d_1 d_2 \cdots d_k \times 10^n,$$

where $1 \leq d_1 \leq 9$,and $0 \leq d_i \leq 9$ for each $i = 1, 2, \cdots, k$.

Numbers of this form are called k-**digit decimal machine numbers**. The left digits $d_{k+1} d_{k+2} \cdots$ can be treated by **chopping** or **rounding** methods.

Example 1 Five-digit chopping or rounding of π.

Definition 1.15 If p^* is an approximation to p, the **absolute error** is $|p - p^*|$ (denoted by δp), and the **relative error** is $\dfrac{|p - p^*|}{|p|}$ (denoted by $\delta_r p$), provided that $p \neq 0$.

Example 2 shows that the relative error is more meaningful, since the relative error takes into consideration the size of the value.

Definition 1.16 The number p^* is said to approximate p to t **significant digit**(有效位数) (or figures) if t is the largest nonnegative integer for which

$$\frac{|p - p^*|}{|p|} < 5 \times 10^{-t}.$$

What is the relationship between relative error and significant digits?

Denote by $fl(y)$ the k decimal digits chopping floating-point form of $y = 0.d_1 d_2 \ldots d_k d_{k+1} \ldots \times 10^n$, then

$$\begin{aligned}
\left|\frac{y - fl(y)}{y}\right| &= \left|\frac{0.d_{k+1}d_{k+2}\ldots \times 10^{n-k}}{0.d_1 d_2 \ldots \times 10^n}\right| \\
&= \left|\frac{0.d_{k+1}d_{k+2}\ldots}{0.d_1 d_2 \ldots}\right| \times 10^{-k} \\
&\leq \frac{1}{0.1} \times 10^{-k} = 10^{-k+1}.
\end{aligned}$$

Similarly, the bound for the relative error when using k-digit rounding arithmetic is $0.5 \times 10^{-k+1}$.

In addition to inaccurate representation of numbers, the arithmetic performed in a computer is not exact.

Assume that the floating-point representations $fl(x)$ and $fl(y)$ are given for the real numbers x and y, the symbols $\oplus$, $\ominus$, $\otimes$, $\oslash$ represent machine addition, subtraction, multiplication and division operations, respectively. Then

$$x \oplus y = fl(fl(x) + fl(y)), \quad x \ominus y = fl(fl(x) - fl(y)),$$
$$x \otimes y = fl(fl(x) \times fl(y)), \quad x \oslash y = fl(fl(x)/fl(y)).$$

Example 4

Let $p = 0.54617$ and $q = 0.54601$. The exact value of $r = p - q$ is $r = 0.00016$. Suppose the subtraction is performed using four-digit arithmetic. Rounding p and q to four digits gives $p^* = 0.5462$ and $q^* = 0.5460$, respectively, and $r^* = p^* - q^* = 0.0002$ is the four-digit approximation to r. Since

$$\frac{|r - r^*|}{|r|} = \frac{|0.00016 - 0.0002|}{|0.00016|} = 0.25,$$

the result has only one significant digit, whereas p^* and q^* were accurate to four and five significant digits, respectively.

If chopping is used to obtain the four digits, the four-digit approximations to p, q, and r are $p^* = 0.5461$, $q^* = 0.5460$, and $r^* = p^* - q^* = 0.0001$. This gives

$$\frac{|r - r^*|}{|r|} = \frac{|0.00016 - 0.0001|}{|0.00016|} = 0.375,$$

which also results in only one significant digit of accuracy. ■

- One of the most common error-producing calculations involves the cancellation of significant digits due to the subtraction of nearly equal numbers, see Example 4.
- If a finite-digit representation or calculation introduces an error, further enlargement of the error occurs when dividing by a number with small magnitude (or, equivalently, when multiplying by a number with large magnitude).

Let a be an approximation of x, b be an approximation of y. For given functions $f(x)$ and $f(x, y)$, consider the error estimates of the approximations $f(a)$ and $f(a, b)$.

Suppose $f(x)$ is continuously differentiable in a neighborhood of a, then

$$f(x) \approx f(a) + f'(a)(x - a),$$

so

$$\delta f(a) \leq |f'(a)|\delta a,$$
$$\delta_r f(a) \leq \left|\frac{f'(a)}{f(a)}\right| \delta a.$$

Similarly, suppose that $f(x, y)$ is continuously differentiable in a neighborhood of a, b. From

$$f(x, y) \approx f(a, b) + f'_x(a, b)(x - a) + f'_y(a, b)(y - b),$$

we have

$$\begin{aligned}
\delta f(a, b) &\leq |f'_x(a, b)|\delta a + |f'_y(a, b)|\delta b, \\
\delta_r f(a, b) &\leq \left|\frac{f'_x(a, b)}{f(a, b)}\right| \delta a + \left|\frac{f'_y(a, b)}{f(a, b)}\right| \delta b.
\end{aligned}$$

Define

$$f(x, y) = x \circ y,$$

where $\circ$ represents $+, -, \times$ or $\div$.

We have

$$\begin{aligned}
\delta(a \pm b) &\leq \delta a + \delta b, \\
\delta_r(a \pm b) &\leq \frac{\delta a + \delta b}{|a \pm b|}, \\
\delta(ab) &\leq b\delta a + a\delta b, \\
\delta_r(ab) &\leq \frac{\delta a}{|a|} + \frac{\delta b}{|b|} = \delta_r a + \delta_r b, \\
\delta\left(\frac{a}{b}\right) &\leq \frac{1}{|b|}\delta a + \frac{|a|}{|b^2|}\delta b, \\
\delta_r\left(\frac{a}{b}\right) &\leq \frac{\delta a}{|a|} + \frac{\delta b}{|b|} = \delta_r a + \delta_r b.
\end{aligned}$$

Then it is clear that we should avoid

- the subtraction of two nearly equal numbers
- dividing by a number with small magnitude, or multiplying by a number with large magnitude

The loss of accuracy due to roundoff error can often be avoided by a **reformulation** of the problem.

Example 5 The roots of $ax^2 + bx + c = 0$, when $a \neq 0$, are

$$x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}, \quad x_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}.$$

If $b^2 >> 4ac$, then the numerator(分子) in the calculation for x_1 (x_2) involves the subtraction of nearly equal numbers when $b > 0$ $(b < 0)$, which brings a poor approximation to x_1 (x_2). See the example $x^2 + 62.10x + 1 = 0$ for the case $b > 0$.

Using four-digit rounding arithmetic, consider this formula applied to the equation $x^2 + 62.10x + 1 = 0$, whose roots are approximately

$$x_1 = -0.01610723 \quad \text{and} \quad x_2 = -62.08390.$$

In this equation, b^2 is much larger than $4ac$, so the numerator in the calculation for x_1 involves the *subtraction* of nearly equal numbers. Since

$$\sqrt{b^2 - 4ac} = \sqrt{(62.10)^2 - (4.000)(1.000)(1.000)} = \sqrt{3856. - 4.000} = \sqrt{3852.}$$
$$= 62.06,$$

we have

$$fl(x_1) = \frac{-62.10 + 62.06}{2.000} = \frac{-0.04000}{2.000} = -0.02000,$$

a poor approximation to $x_1 = -0.01611$, with the large relative error

$$\frac{|-0.01611 + 0.02000|}{|-0.01611|} \approx 2.4 \times 10^{-1}.$$

On the other hand, the calculation for x_2 involves the *addition* of the nearly equal numbers $-b$ and $-\sqrt{b^2 - 4ac}$. This presents no problem since

$$fl(x_2) = \frac{-62.10 - 62.06}{2.000} = \frac{-124.2}{2.000} = -62.10$$

has the small relative error

$$\frac{|-62.08 + 62.10|}{|-62.08|} \approx 3.2 \times 10^{-4}.$$

To obtain a more accurate approximation for x_1 when $b > 0$, we change the form of the quadratic formula by **rationalizing the numerator:**

$$x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \cdot \frac{-b - \sqrt{b^2 - 4ac}}{-b - \sqrt{b^2 - 4ac}} = \frac{-2c}{b + \sqrt{b^2 - 4ac}},$$

which greatly improves the approximation of x_1 and leads to a small relative error for the equation $x^2 + 62.10x + 1 = 0$.

For the case of $b < 0$, the similar rationalization technique can also be applied to give the following alternative quadratic formula for x_2:

$$x_2 = \frac{-2c}{b - \sqrt{b^2 - 4ac}}.$$

An alternative approach reads

$$x_1 = \frac{-b - \text{sign}(b)\sqrt{b^2 - 4ac}}{2a}, \quad x_2 = \frac{c}{ax_1}.$$

Accuracy loss due to roundoff error can also be reduced by rearranging calculations.

Example 6 Evaluate $f(x) = x^3 - 6.1x^2 + 3.2x + 1.5$ at $x = 4.71$ using 3-digit arithmetic.

- Method 1: directly computing.
- Method 2: computing the **nested** form

$$f(x) = ((x - 6.1)x + 3.2)x + 1.5.$$

It is clear that nesting has reduced the relative error of direct method.

Polynomials should always be expressed in nested form before performing an evaluation, because this form minimizes the number of arithmetic calculations.

1.3 Algorithms and Convergence

Algorithm: an algorithm is a procedure that describes, in an unambiguous or clear manner, a finite sequence of steps to be performed in a specified order.

Key techniques for algorithm: looping and condition-control method

Description: pseudo-code method.

Example 1: Compute $\sum_{i=1}^{n} x_i = x_1 + x_2 + \cdots + x_n$.

The algorithm can be described as

INPUT: $N, x_1, x_2, \cdots, x_n$.

OUTPUT: SUM=$\sum_{i=1}^{n} x_i$.

Step 1 Set SUM=0

Step 2 For $i = 1, 2, \cdots, N$ do

set SUM=SUM+x_i

Step 3 OUTPUT (SUM);

STOP.

Example 2: The Nth Taylor polynomial for $\ln x$ expanded about $x_0 = 1$ is

$$P_N(x) = \sum_{i=1}^{N} \frac{(-1)^{i+1}}{i}(x-1)^i,$$

compute the minimal value of N required for

$$|\ln 1.5 - P_N(1.5)| < 10^{-5}.$$

The algorithm can be described as

INPUT: value x, tolerance TOL, maximum number of iterations M.

OUTPUT: degree N of the polynomial or a message of failure.

Step 1 Set $N = 1$; $y = x - 1$;

$SUM = 0$;

$POWER = y$;

$TERM = y$;

$SIGN = -1$.

Step 2 While $N \leq M$ do Steps 3-5.

Step 3 Set $SIGN = -SIGN$;

$SUM = SUM + SIGN \cdot TERM$;

$POWER = POWER \cdot y$;

$TERM = POWER/(N + 1)$;

Step 4 If $|TERM| < TOL$ then

OUTPUT (N);

STOP.

Step 5 Set $N = N + 1$.

Step 6 OUTPUT ('Method Failed');

STOP.

Some important concepts on algorithm:

Stability:

An algorithm is said to be **stable** imply that small changes in the initial data can produce correspondingly small changes in final results. Otherwise it is **unstable**. Some algorithm are stable only for certain choices of initial data, this case are called **conditionally stable**.

Definition 1.17 Suppose that E_0 denotes an initial error and E_n represents the magnitude of an error after n subsequent operations. If $E_n \approx CnE_0$, where C is a constant independent of n, then the growth of error is said to be **linear**. If $E_n \approx C^n E_0$, for some $C > 1$,then the growth of error is called **exponential**.

An algorithm that exhibits linear growth of error is stable, whereas an algorithm exhibiting exponential error growth is unstable and therefore should be avoided.

Example 3: The recursive equation

$$p_n = \frac{10}{3} p_{n-1} - p_{n-2}, \text{ for } n = 2, 3, \cdots,$$

has the solution

$$p_n = c_1 \left(\frac{1}{3}\right)^n + c_2 3^n.$$

If $p_0 = 1$ and $p_1 = \frac{1}{3}$, we have $c_1 = 1$ and $c_2 = 0$ so $p_n = \left(\frac{1}{3}\right)^n$.

Suppose that 5-digit rounding arithmetic is used, then $\hat{p}_0 = 1.0000$ and $\hat{p}_1 = 0.33333$, so we have $\hat{c}_1 = 1.0000$ and $\hat{c}_2 = -0.12500 \times 10^{-5}$, therefore

$$\hat{p}_n = 1.0000 \left(\frac{1}{3}\right)^n - 0.12500 \times 10^{-5} 3^n,$$

and the roundoff error

$$p_n - \hat{p}_n = 0.12500 \times 10^{-5} 3^n,$$

grows exponentially with n, see Table 1.5.

Table 1.5

n	Computed $\hat{p}_n$	Correct p_n	Relative Error
0	0.10000×10^{1}	0.10000×10^{1}	
1	0.33333×10^{0}	0.33333×10^{0}	
2	0.11110×10^{0}	0.11111×10^{0}	9×10^{-5}
3	0.37000×10^{-1}	0.37037×10^{-1}	1×10^{-3}
4	0.12230×10^{-1}	0.12346×10^{-1}	9×10^{-3}
5	0.37660×10^{-2}	0.41152×10^{-2}	8×10^{-2}
6	0.32300×10^{-3}	0.13717×10^{-2}	8×10^{-1}
7	-0.26893×10^{-2}	0.45725×10^{-3}	7×10^{0}
8	-0.92872×10^{-2}	0.15242×10^{-3}	6×10^{1}

Consider the recursive equation

$$p_n = 2p_{n-1} - p_{n-2}, \text{ for } n = 2, 3, \cdots,$$

it has the solution

$$p_n = c_1 + c_2 n.$$

If $p_0 = 1$ and $p_1 = \frac{1}{3}$, we have $c_1 = 1$ and $c_2 = -\frac{2}{3}$ so $p_n = 1 - \frac{2}{3}n$.

5-digit rounding arithmetic in this case results in $\hat{p}_0 = 1.0000$ and $\hat{p}_1 = 0.33333$, and $\hat{c}_1 = 1.0000$ and $\hat{c}_2 = -0.66667$, thus

$$\hat{p}_n = 1.0000 - 0.66667n,$$

and the roundoff error

$$p_n - \hat{p}_n = \left(0.66667 - \frac{2}{3}\right) n,$$

grows linearly with n, see Table 1.6.

Table 1.6

n	Computed $\hat{p}_n$	Correct p_n	Relative Error
0	0.10000×10^1	0.10000×10^1	
1	0.33333×10^0	0.33333×10^0	
2	-0.33330×10^0	-0.33333×10^0	9×10^{-5}
3	-0.10000×10^1	-0.10000×10^1	0
4	-0.16667×10^1	-0.16667×10^1	0
5	-0.23334×10^1	-0.23333×10^1	4×10^{-5}
6	-0.30000×10^1	-0.30000×10^1	0
7	-0.36667×10^1	-0.36667×10^1	0
8	-0.43334×10^1	-0.43333×10^1	2×10^{-5}

Another example: Compute the values of

$$I_n = \int_0^1 \frac{x^n}{x+5} dx, n = 0, 1, \cdots, 8.$$

Note that I_n satisfies $I_n + 5I_{n-1} = \frac{1}{n}$, two algorithms can be designed (4-digits are used):

Algorithm 1. $I_0 \to I_1 \to I_2 \to \cdots \to I_8$ (unstable)

Algorithm 2. $I_8 \to I_7 \to I_6 \to \cdots \to I_0$ (stable)

n	Algorithm 1	Algorithm 2	accurate value
0	0.1823	0.1823	0.182322
1	0.08850	0.08839	0.0883922
2	0.05750	0.05804	0.0580389
3	0.04583	0.04314	0.0431387
4	0.02085	0.03431	0.0343063
5	0.09575	0.02847	0.0284684
6	-0.3121	0.02433	0.0243249
7	1.703	0.02123	0.0212326
8	-8.392	0.01884	0.0188369

The effects of roundoff error can be reduced by using high-order-digit arithmetic such as the double- or multiple-precision option available on most computers. Disadvantages in using double-precision arithmetic are that it takes more computation time and the growth of roundoff error is not eliminated but is only postponed until subsequent computations are performed.

Rate of convergence:

Definition 1.18 Suppose $\{\beta_n\}_{n=1}^{\infty}$ is a sequence known to converge to zero, and $\{\alpha_n\}_{n=1}^{\infty}$ converges to a number α. If a positive constant K exists with

$$|\alpha_n - \alpha| \leq K|\beta_n|,$$

for large n, then we say that $\{\alpha_n\}_{n=1}^{\infty}$ converges to α with the **rate of convergence** $O(\beta_n)$, writing $\alpha_n = \alpha + O(\beta_n)$.

We often use

$$\beta_n = \frac{1}{n^p}$$

for some number $p > 0$. We are generally interested in the largest value of p with $\alpha_n = \alpha + O(1/n^p)$.

Example 4: Suppose that for $n \geq 1$,

$$\alpha_n = \frac{n+1}{n^2}, \quad \text{and} \quad \hat{\alpha}_n = \frac{n+3}{n^3}.$$

we can see that

$$|\alpha_n - 0| = \frac{n+1}{n^2} \leq \frac{n+n}{n^2} = 2\frac{1}{n}$$

and

$$|\hat{\alpha}_n - 0| = \frac{n+3}{n^3} \leq \frac{n+3n}{n^3} \leq 4\frac{1}{n^2}$$

so

$$\alpha_n = 0 + O(\frac{1}{n}), \quad \text{and} \quad \hat{\alpha}_n = 0 + O(\frac{1}{n^2}).$$

Definition 1.19 Suppose that

$$\lim_{h\to 0} G(h) = 0$$

and

$$\lim_{h\to 0} F(h) = L.$$

If a positive constant K exists with

$$|F(h) - L| \leq K|G(h)|,$$

for sufficient small h, then we write

$$F(h) = L + O(G(h)).$$

We often use $G(h) = h^p$ for $p > 0$. We are interested in the largest value of p for which $F(h) = L + O(h^p)$.

Example 5:

$$\cos h = 1 - \frac{1}{2}h^2 + \frac{1}{24}h^4 \cos \xi(h)$$

since

$$|(\cos h + \frac{1}{2}h^2) - 1| = |\frac{1}{24}h^4 \cos \xi(h)| \leq \frac{1}{24}h^4,$$

so

$$\cos h + \frac{1}{2}h^2 = 1 + O(h^4)$$

1.4 Numerical Software

Computer Software:

C, Fortran, Maple, Mathematica, Matlab, Pascal

programs:

special purpose

general purpose