Last modified: March 26, 2025

講義名
福祉音響学

担当
村上 泰樹

連絡先
murakami@design.kyushu-u.ac.jp

Unit
6

1　この単元の目的

本単元の目的は、数値計算の基礎となる「誤差」について学ぶことです。数値シミュレーションは「数値実験」とも呼ばれており、実際の実験と同様の位置づけにあります。科学技術実験を行う際には誤差の扱いが重要ですが、数値計算においても同様に誤差への理解が必要です。数値計算では主に二種類の誤差が発生します。一つは計算機の有限精度による誤差であり、もう一つは計算アルゴリズムに起因する誤差です。

この単元では、計算機の有限精度に起因した誤差を適切に理解し、誤差を評価する方法を習得することを目指します。そうすることで数値計算の信頼性を高めることができます。また、数値計算や数値シミュレーションを行う際に避けられない誤差の問題に焦点を当て、それを理解・管理することで計算結果の精度と信頼性を向上させる方法を学ぶことを主眼としています。

単元は、以下の書籍を参考に進めていきます。

- 伊理正夫・藤野知建, 数値計算の基礎, 共立出版株式会社, 1985.

2　誤差

数値計算は、計算機を用いて数値的な計算を行う。計算機は有限精度で数値を表現するため、計算結果には誤差が生じる。実際、数値シミュレーションは、「数値実験」と呼ばれることもある。これは、数値計算が実験と同様の位置づけにあることを示している。実験においては、測定器の精度や測定誤差が問題となるが、数値計算においては計算機の有限精度による誤差が問題となる。

まず、最初に誤差の定義を行う。ある量 x の測定値 a に見込まれる誤差を $\Delta a(>0)$ であるとする。このとき、x の値は $a \pm \Delta a$ の範囲にあると考える。Δa の値は、x の値がこの範囲にあると考えられる範囲である。

$$x \in (a - \Delta a) \tag{1}$$

もしくは、$x = a \pm \Delta a$ のように略記を用いて記述される。ここで、2 つの量 x と y を考える。

$$x = a \pm \Delta a,\ y = b \pm \Delta b \tag{2}$$

このとき、x と y の関数として計算される量 z は、量 c と誤差 Δc を持つ。

$$z = c \pm \Delta c \tag{3}$$

量 z は、観測された 2 つの量 x と y の関数 f として計算されるとすると、量 c と誤差 Δc は次のように表される。

$$c = f(a, b),\ \Delta c = |f_x(a, b)|\,\Delta a + |f_y(a, b)|\,\Delta b \tag{4}$$

数値計算で行う計算は、四則演算がほとんどである。故に、四則演算における誤差の伝播を考えることが重要である。四則演算における誤差の伝播は、以下のように表される。まず、加減算における誤差の伝播について考える。量 z は、x と y の加減算によって計算されるとする。

$$z = x \pm y \tag{5}$$

このとき、z の誤差 Δc は、x と y の誤差 Δa と Δb によって次のように表される。

$$\Delta c = \Delta a + \Delta b \tag{6}$$

負の符号が消えているが、これは量 Δc は誤差として含まれる量の最大値であるためである。次に、乗算と除算における誤差の伝播について考える。量 z は、x と y の乗算と除算によって計算されるとする。

$$z = x \cdot y,\ z = x/y \tag{7}$$

このとき、z の誤差 Δc は、真値 a 及び b、c と x と y の誤差 Δa と Δb によって次のように表される。

$$\left|\frac{\Delta c}{c}\right| = \left|\frac{\Delta a}{a}\right| + \left|\frac{\Delta b}{b}\right| \tag{8}$$

上記の式は、分母に真値があることから分かるように相対的な値であり、相対誤差と呼ばれる。一方、加減算における誤差の伝播は、絶対誤差と呼ばれる。

3 クイズ 1

Python を用いて、以下の計算を行い、その結果を出力せよ。

1. 1 を 1000 回足し合わせる計算
2. 0.1 を 1000 回足し合わせる計算

4 数の表現

整数型は、計算機のメモリ上で 2 進数として表現される。整数型の表現には、符号付きと符号なしの 2 種類がある。符号付き整数型は、正負の値を表現するために 1 ビットを符号ビットとして用いる。符号なし整数型は、正の値のみを表現するために符号ビットを用いない。整数型の表現には、2 の補数表現が用いられる。2 の補数表現は、正の値と負の値を同じビットパターンで表現することができる。2 の補数表現は、符号ビットを用いる符号付き整数型においても、符号ビットを用いない符号なし整数型においても用いられる。現在の計算機では、64 ビットの整数型が一般的である。64 ビットの整数型は、2^{64} 個の整数を表現することができる。10 進数で表現すると、およそ 19 桁の整数を表現することができる。構造図を図 1 に示す。

一方、実数型は、計算機のメモリ上で固定少数点数もしくは浮動小数点数として表現される。固定小数点は、整数型と同様に 2 進数として表現される。固定小数点は、整数型と同様に符号付きと符号なしの 2 種類がある。固定小数点は、整数型と同様に 2 の補数表現が用いられる。固定小数点は、小数点以下の桁数を固定して表現する (図 2)。固定小数点は、小数点以下の桁数を固定して表現するため、小数点以下の桁数を表現するためにビット数を増やす必要がある。マイコンなどの組み込みシステムなどで使用されることはあるが、一般的な数値計算で使用する計算機では固定小数点を用いることは少ない。

浮動小数点数は、指数部と仮数部から構成される。浮動小数点数の表現には、IEEE754 規格が用いられる。図 3 が IEEE754 規格のビット割り当てを表現している。IEEE754 規格は、32 ビットと 64 ビットの 2 種類がある。32 ビットの浮動小数点数は、単精度浮動小数点数と呼ばれ、64 ビットの浮動小数点数は、倍精度浮動小数点数と呼ばれる。単精度浮動小数点数は、32 ビットのメモリを用いて表現される。単精度浮動小数点数は、符号ビット、指数部、仮数部から構成される。符号ビットは、正負の値を表現するために 1 ビットを用いる。指数部は、浮動小数点数の桁数を表現するために 8 ビットを用いる。仮数部は、浮動小数点数の精度を表現するために 23 ビットを用いる。倍精度浮動小数点数は、64 ビットのメモリを用いて表現される。倍精度浮動小数点数は、符号ビット、指数部、仮数部から構成される。符号ビットは、正負の値を表現するために 1 ビットを用いる。指数部は、浮動小数点数の桁数を表現するために 11 ビットを用いる。仮数部は、浮動小数点数の精度を表現するために 52 ビットを用いる。指数部のビット数が単精度と倍精度では、異なるため、単精度と倍精度の浮動小数点数の表現範囲が異なる。

浮動小数点数など有限の桁数で実数を表現する際、表現できない桁は丸められる。この過程で発生する理論上の値との差を丸め誤差と呼ぶ。丸め誤差により、数値計算の結果は理論上の厳密解と異なる値になることがある。

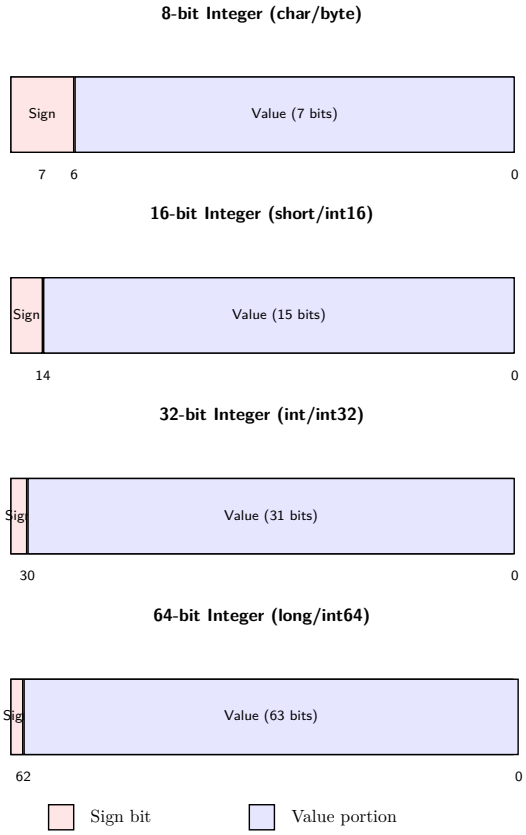

Fig. 1　Integer data type structure.

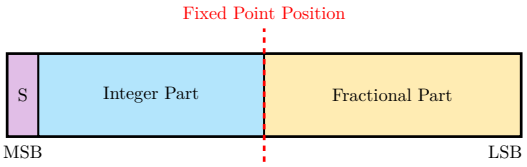

Fig. 2　Fixed–point number structure.

浮動小数点数を用いた繰り返し足し算における丸め誤差の累積が生じることは先に説明したとおりである。図 4 では、青線は 0.04 を繰り返し足していった場合の数学的に正確な結果を示し、赤い破線は丸め誤差によって理論値から徐々に逸脱していく実際の計算値を表している。0.04 のような小数は二進数形式で正確に表現できないため、各演算ごとに小さな丸め誤差が生じる。これらの誤差は相殺されるのではなく、系統的に蓄積される傾向にある。赤く塗られた領域は、足し算の回数が増えるにつれて真の値と計算値の間の差異が拡大していく様子を強調したものである。

5　クイズ 2

単精度浮動小数点と倍精度浮動小数点の変数の有効桁数を求めよ。

6　桁落ちに気をつけよう

方程式 $f(x) =$ を解くことは、数値計算的には「$f(x)$ が十分 0 に近くなるような x を求める」ということである。誤っても、$f(x)$ が 0 になる x を求めることはできない。具体的に、次の方程式について考えるとする。

$$f(x) = x^3 - 15.70875x^2 + 61.69729x - 0.04844725 \tag{9}$$

この方程式の零点は以下の付近にある。

$$x \simeq 0.0007853982,\ 7.844763,\ 7.863201 \tag{10}$$

問題は、それぞれの零点の値を求めるとき、$f(x)$ の値をどこまで小さくすることで解くことができたのかということである。

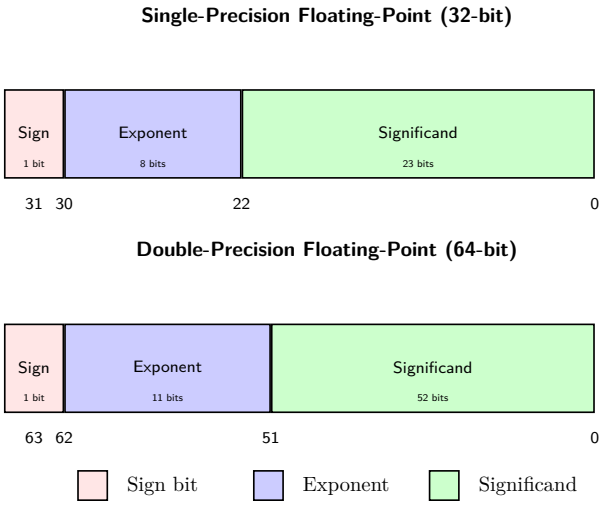

Fig. 3　IEEE754 floating–point structure.

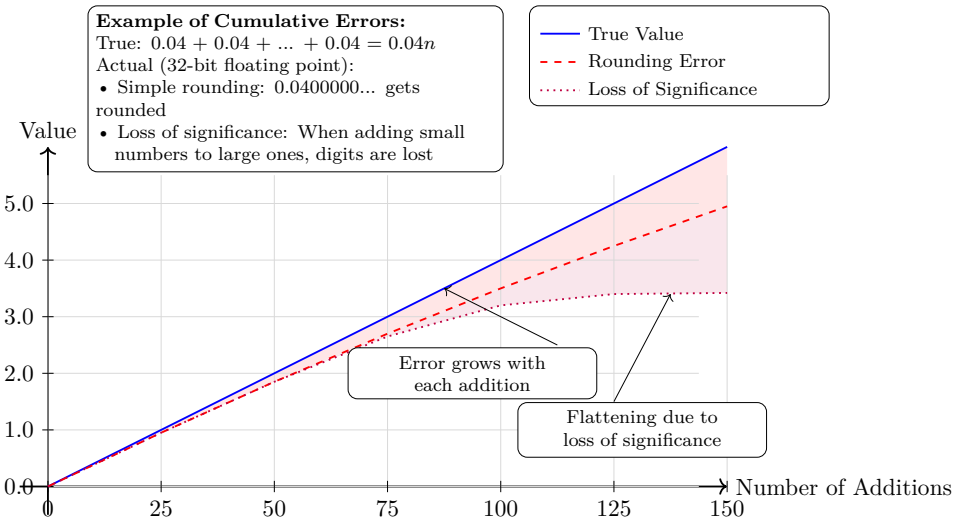

Fig. 4　Accumulation of Rounding Errors in Repeated Addition.

あまり厳しい条件を設定しても、計算中に発生する丸め誤差があるため条件を満足させることはできない。ここでは、丸め誤差の知識を利用して、大まかな見当をつけることが重要である。

$x \simeq 0.00078$ の付近について検討する。それぞれの項の値を計算すると、次のようになる。

$$x^3 \simeq 4.8 \times 10^{-10},\ -15.70875 \times x^2 \simeq -9.7 \times 10^{-6},\ 61.69729 \times x \simeq 0.048 \tag{11}$$

ここで、計算を単精度浮動小数点数の条件で行うと、10 進 7 桁程度の有効数字で計算（四捨五入）される。このとき、各項で 10^{-16} 及び 10^{-11}、10^{-8} 程度の誤差が生じる。従って、$|f(x)|$ が 10^{-8} のオーダーになったならば、それ以上細かく x の値を調整しても、それに伴う $f(x)$ の値の変化には意味がない。

7　クイズ 3

式 9 について、$x \simeq 7.8\cdots$ の付近について検討せよ。

8　Unit 6 のまとめ

数値計算は実際の実験と同様に「数値実験」と呼ばれており、計算結果の信頼性には誤差の理解が不可欠です。このユニットで学んだ主な内容は以下の通りです：

1. 誤差の基本概念：
   - 絶対誤差と相対誤差の区別
   - 四則演算における誤差の伝播法則（加減算では絶対誤差、乗除算では相対誤差が重要）
2. 計算機における数値表現：
   - 整数型（符号付き・符号なし）と 2 の補数表現
   - 実数型（固定小数点・浮動小数点）の構造
   - IEEE754 規格による単精度・倍精度浮動小数点数の表現
3. 丸め誤差と桁落ち：
   - 浮動小数点数の有限桁数による丸め誤差の発生
   - 繰り返し計算による誤差の蓄積現象
   - 単精度と倍精度での有効桁数の違い
4. 数値計算における実践的考慮：
   - 方程式の解法における誤差の影響評価
   - 求める精度に応じた適切な計算方法の選択

実際の数値計算では、理論上の厳密解と計算結果の差を最小限に抑えるため、これらの誤差の性質を理解し、適切に対処することが重要です。このユニットで学んだ知識は、信頼性の高い数値シミュレーションを実施するための基礎となります。