


ICT 演習（基本情報）

2 年生、3 学期

第 2 回：演算の誤差とその評価

第1章

1回	数値の内部表現
2回	演算の誤差とその評価 
3回	組合せ回路と順序回路
4回	オートマトンによる状態遷移記述
5回	正規表現、BNFによる記号列の生成
6回	逆ポーランド記法による数式記述

第3章

7回	磁気ディスク装置の構成と性能評価 / コンピュータの基本構成
8回	クライアントサーバシステムの事例研究
9回	システムの性能評価
10回	システムの信頼性評価

第4章

11回	E-R図によるデータモデル化
12回	3層スキーマによるデータベース設計

第7章

13回	要求分析と要求定義
-----	-----------

第9章

14回	BPR(Business Process Re-engineering) の事例
-----	--

第10章

15回	CSR(Corporate Social Responsibility) の事例
-----	--

1 回	数値の内部表現	30/11
2 回	演算の誤差とその評価	30/11
3 回	組合せ回路と順序回路	7/12
4 回	オートマトンによる状態遷移記述	7/12
5 回	正規表現、BNFによる記号列の生成	14/12
6 回	逆ポーランド記法による数式記述	14/12
7 回	磁気ディスク装置の構成と性能評価 / コンピュータの基本構成	21/12
8 回	クライアントサーバシステムの事例研究	21/12
9 回	システムの性能評価	11/1
10 回	システムの信頼性評価	11/1
11 回	E-R 図によるデータモデル化	18/1
12 回	3 層スキーマによるデータベース設計	18/1
13 回	要求分析と要求定義	25/1
14 回	BPR(Business Process Re-engineering) の事例	25/1
15 回	CSR(Corporate Xocial Responsibility) の事例	1/2

演算の誤差

- コンピュータは、有限桁の数値を扱います。
- そのため、 π のような
えい えん しょうじる
□ 永遠に続く桁を有限桁で切ることによって生じる誤差や、
かてい
□ 計算過程で生じる誤差、

□ 有限桁で表せる最大値を超えた計算を行うことによって生じる誤差
などが起こります。

演算誤差 「Calculation error」

1. 桁落ち (Digit Loss)
2. 情報落ち (Information Loss)
3. 丸め誤差 (Rounding Error)
4. 打ち切り誤差 (Truncation error)
5. オーバフロー, アンダフロー (Overflow , Underflow)

1. 桁落ち「Digit Loss」

```
var b = 0.123456;  
var a = 0.123455;  
console.log(b-a);
```

- 桁落ちは、値がほぼ等しい数値同士の減算により、有効桁数が減少することをいいます。たとえば、有効桁数3の浮動小数点数同士の演算

```
print(13.3-13.2)  
print(13000.3-13000.2)
```

「 $0.789 \times 10^5 - 0.788 \times 10^5$ 」の結果は、

$$\begin{aligned} &0.789 \times 10^5 - 0.788 \times 10^5 \\ &= 0.001 \times 10^5 \\ &= 0.1 \times 10^3 \end{aligned}$$

桁落ちとは、近い数を引き算することで有効数字が少なくなる現象のことです。

となり、有効桁数は1に減少します。

1. 桁落ち「Digit Loss」

$$\underline{1.2345} - \underline{1.2344} = \underline{0.0001}$$

有限数字 5 桁

有限数字 5 桁

有限数字 1 桁

有効数字が減少

正規化

$$1.\underline{0000} \times 10^{-4}$$

小数点以下の 4 つの 00 は意味が無い です

このような状況を「0 埋めされる」と言われています。

実際、コンピュータでは浮動小数点数を使って計算を行っているため、0 埋めが発生します。

有効桁が4桁													
0.555 × 10 ³				−	0.554 × 10 ³								
有効桁数の数字は3桁					有効桁数の数字は3桁								
= 0.001 × 10 ³													
有効桁数の数字は1桁				←	有効桁数の数字が3桁から1桁に減る								
				↓	正規化								
0.100 × 10 ¹													

2. 情報落ち 「Information Loss」

- 情報落ちとは、絶対値の大きい数値と小さい数値の加減算を行ったとき、絶対値の小さい数値が演算結果に反映されないことで生じる誤差です。これは、浮動小数点数演算において、指数の大きい値に合わせて仮数部を調整することに起因します。たとえば、有効桁数3桁の演算「 $0.111 \times 10^5 + 0.111 \times 10^{-1}$ 」の場合、

$$\begin{aligned} &0.111 \times 10^5 + 0.111 \times 10^{-1} \\ &= 0.111 \times 10^5 + \\ &0.000000111 \times 10^5 \\ &= 0.111000111 \times 10^5 \\ &= 0.111 \times 10^5 \end{aligned}$$

```
var b = 0.000000000000006789;  
var a  
=10000000000000000000001111;  
console.log(b+a);
```

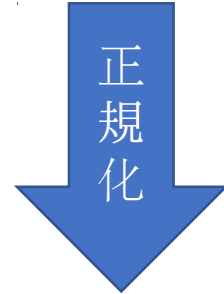

2. 情報落ち 「Information Loss」

例: 1

$$0.111 \times 10^5 + 0.111 \times 10^{-1}$$

$$\begin{array}{r} 0.111 \quad \times 10^5 \\ + 0.000000111 \times 10^5 \\ \hline \end{array}$$

$$0.111000111 \times 10^5$$



$$\underline{0.111 \times 10^5}$$

有効桁数 3 桁

すっごく大きい数と
すっごく小さい数を
加減算すると、
有効数字によって小
さい数の意味がなく
なるということ

… となり、期待する 0.111000111 とは 0.000000111 の誤差が生じます。

2. 情報落ち 「Information Loss」

例: 2

1234 + 0.01 を計算すると、答えは 1234.01 だが、これを浮動小数点演算すると。。。。

$$\begin{array}{r} 1234 \\ + \quad \quad \quad 0.01 \\ \hline \end{array}$$

1 2 3 4 . 01

… となり、期待する 1234.01 とは 0.01 の誤差が生じます。

正規化

1234

ここで、0.01 が 0 になってしまうので、答えが 1234 になる

2. 情報落ち 「Information Loss」

例: 3

しかし、2つの数値を加算する場合は指数を揃えて計算するた

0.123 => 0.123 浮動小数点演算すると
0.00123 => 0.001

$$\begin{array}{r} 0.123 \\ + 0.001 \\ \hline 0.124 \end{array}$$

… となり、期待する 0.12423 とは 0.00023 の誤差が生じます。

3 . 丸め誤差 「Rounding error」

- コンピュータ内で扱える数値は有限桁の数値です。
- そのため、有限桁に入り切らない部分に対し四捨五入、切上げ、切捨てのいずれかを行うことによって誤差が生じます。
- これを丸め誤差と呼びます。
- たとえば、10 進数 1234.567 を小数第 1 位で四捨五入すると 1234.6 となり、0.033 の丸め誤差が生じます。

```
var a = 0.1 + 0.2; // 0.30000000000000004
```

```
console.log(2*(1/3)-(2/3));  
Console.log(0.1+0.2)
```

4 . 打ち切り誤差 Truncating error

- ある値を演算で求めるとき，その結果が循環小数になる場合や時間の経過とともに徐々に真の値に近づくような場合に，時間の制限または一定の桁数で演算を打ち切ることで生じる誤差を打ち切り誤差と呼びます。

円周率やネイピア数のような無限に続く計算を途中でやめることで生じてしまう誤差のこと

4 . 打ち切り誤差 Truncating error

- コンピュータ側が計算処理を完了まで待たずに途中で打ち切ることによって起こる誤差が打ち切り誤差です。

例 1 :

Pi

$\pi = 3.14159265358979 \dots\dots\dots$

こういった値は、あらかじめ定められた規則（「X桁で終了」）で途中で打ち切られる。

4 . 打ち切り誤差 Truncating error

例 2 :

「 $1 \div 3 = 0.333333333333333333333333\dots$ 」のような無限に続く計算を行う場合、コンピュータは途中で計算を打ち切ります。

その結果、正しい結果との間に誤差が発生します。この時に発生する誤差が「打ち切り誤差」です。

演算誤差

演算したときに、有効桁数の関係で発生する誤差

丸め誤差

ある桁数で切り上げ、切り捨て、
四捨五入したときに発生する誤差

打ち切り誤差

計算や処理を、途中で打ち
切ったときに発生する誤差

情報落ち

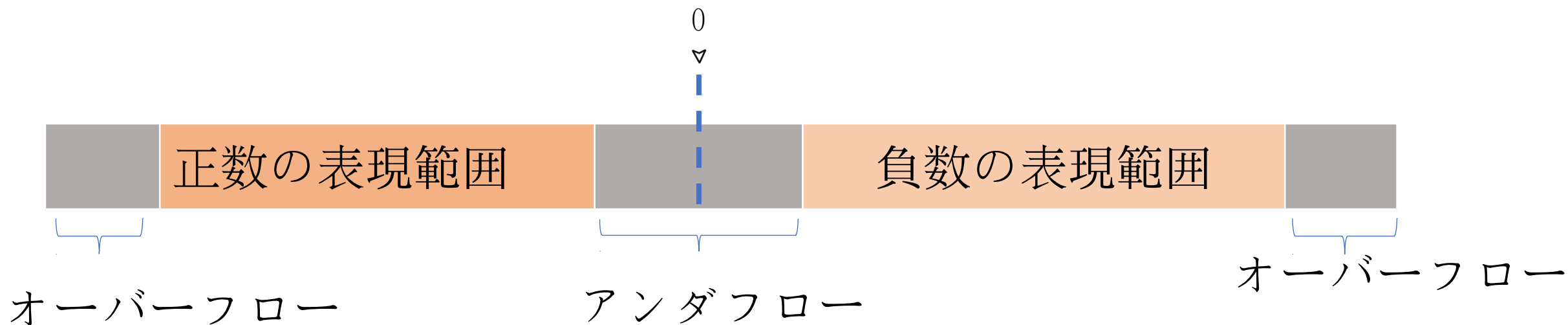
絶対値が違い数字同上を加
減算したときに発生する誤差

桁落ち

絶対値に近い数字同上を加
減算したときに発生する誤差

5. オーバフロー, アンダフロー

- 演算結果が表現可能な数値の範囲を超えた場合に生じる誤差で、最大値を超えた場合はオーバフロー（桁あふれ）、最小値を超えた場合はアンダフローとなります。



5. オーバフロー, アンダフロー

8ビット符号付き整数同士の演算によるオーバフローの例を示します。

```

                                11100000
00100000 ( 32 )      ( - 32 )
+ 01100001 ( 97 ) + 10011111 ( -
-----97 )
---
10000001 ( -
127 ? )

01111111 ( 127 ? )
```

7	6	5	4	3	2	1	0
1	1	1	1	1	1	1	1

Decimal : $2^0 + 2^1 + 2^2 + 2^3 + 2^4 + 2^5 + 2^6 + 2^7 = 255$

5. オーバフロー, アンダフロー

オーバフロー

1001 (+9)₁₀

1000 (+8)₁₀

10001 (+17)₁₀



コンピュータは **+1** だと思う

アンダフロー

1010 (-6)₁₀

1011 (-5)₁₀

10101 (-11)₁₀



コンピュータは **+5** だと思う