

수의 표현과 계산상의 오차

고급수치해석

2016. 3. 17

수의 표현

(2진수와 10진수)

- 10진수와 2진수의 상관관계 학습 이유?
 - : 인간에 의한 정밀한 수치적 근삿값 계산의 한계
- 10진법 (decimal system)
 - : 아라비아 숫자 0~9까지 열 개의 10진수 사용
 - 10진수: 인간의 숫자
- 2진법 (binary notation)
 - : 0과 1 두 개의 2진수 사용
 - 2진수: 컴퓨터의 숫자 혹은 언어

2진 정수와 10진 정수(1/3)

● 2진 정수 (binary integer), B

- 0과 1로 표현되는 유한수열

$$B = (b_n b_{n-1} \cdots b_2 b_1 b_0)_2$$

● 10진 정수 (decimal integer), D

● 2진 정수를 10진 정수로 변환

$$D = b_n 2^n + b_{n-1} 2^{n-1} + \cdots + b_1 2^1 + b_0$$

- $B=(10101)_2$ 를 D 로 변환

$$D = 2^4 + 2^2 + 2^0 = 21$$

- 1이 n 개인 2진 정수 $B=(111\cdots 1)_2$ 를 D 로 변환

$$D = 2^{n-1} + \cdots + 2^1 + 1 = 2^n - 1$$

2진 정수와 10진 정수(2/3)

● 10진 정수를 2진 정수로 변환

- 식 $D = b_n 2^n + b_{n-1} 2^{n-1} + \cdots + b_1 2^1 + b_0$ 을 2로 나눈다 : 몫 D_1 , 나머지 b_0

$$D_1 = b_n 2^{n-1} + b_{n-1} 2^{n-2} + \cdots + b_1 2^0$$

- D_1 을 2로 다시 나누면 새로운 몫 D_1 , 나머지 b_1 생성

- 몫이 0이 될 때까지 반복

- 나머지 0과 1를 역순으로 나열

- $D=(11)_{10}$ 를 B 로 변환

$$2 \overline{)11} = 5 = D_1 \quad b_0 = 1$$

$$2 \overline{)5} = 2 = D_2 \quad b_1 = 1$$

$$2 \overline{)2} = 1 = D_3 \quad b_2 = 0$$

$$2 \overline{)1} = 0 = D_4 \quad b_3 = 1$$

- 나머지를 역순으로 $(b_3 b_2 b_1 b_0)_2 = (1011)_2$

2진 정수와 10진 정수(3/3)

문제 1. 2진 정수와 10진 정수의 상호 변환

맷랩을 이용하여 2진 정수 111을 10진 정수로 변환하라. 또 반대로 10진 정수 7을 2진 정수로 변환하라.

Tip !

- ✓ **bin2dec**: 2진 정수를 10진 정수로 변환
 - ✓ `bin2dec('111')`
- ✓ **dec2bin**: 10진 정수를 2진 정수로 변환
 - ✓ `dec2bin(7)`

2진 소수와 10진 소수(1/6)

- **2진 소수 (binary fractions), B**
 - 0과 1로 표현되는 무한수열
- **10진 소수 (decimal fractions), D**
- **2진 소수를 10진 소수로 변환**

$$B = (b_1 b_2 b_3 \cdots b_n \cdots)_2$$

$$D = b_1 2^{-1} + b_2 2^{-2} + b_3 2^{-3} + \cdots$$

- $B = (.111)_2$ 를 D 로 변환

$$D = 2^{-1} + 2^{-2} + 2^{-3} = 0.5 + 0.25 + 0.125 = 0.875$$

2진 소수와 10진 소수(4/6)

● 10진 소수를 2진 소수로 변환

- 식 $D = b_1 2^{-1} + b_2 2^{-2} + b_3 2^{-3} + \dots$ 의 다른 표현

$$D = D_1 = b_1 2^{-1} + b_2 2^{-2} + b_3 2^{-3} + \dots$$

- 2를 곱한다 : b_1 이 정수 (소거)

$$D_2 = b_2 2^{-1} + b_3 2^{-2} + b_4 2^{-3} + \dots$$

- D_2 에 2를 곱하면 정수 b_2 생성

- 소수 자리가 0이 될 때까지 반복

- 생성된 정수 부분 나열

2진 소수와 10진 소수(5/6)

- $D = \frac{1}{5}$ 를 B로 변환

$$D = D_1 = 0.2$$

$$2D_1 = 0.4 = D_2 \rightarrow b_1 = 0$$

$$2D_2 = 0.8 = D_3 \rightarrow b_2 = 0$$

$$2D_3 = 1.6 = D_4 \rightarrow b_3 = 1$$

$$2D_4 = 1.2 = D_5 = 0.2 = D_1 \rightarrow b_4 = 1$$

- 0011을 반복하는 2진 순환 소수 생성

$$(.001100110011\dots)_2 = (.0011)_2$$

2진 소수와 10진 소수(6/6)

문제 3. 2진 정수와 10진 정수의 상호 변환

2진 소수 $(.0101010101010)_2$ 을 10진 소수로 변환하라. 그리고 10진 소수 $(0.2)_{10}$ 를 2진 소수로 변환하라.

2진 부동 소수점 수와 10진 부동 소수점 수(1/6)

● 부동산 소수점 수

- 소수점의 위치를 고정하지 않고 실수를 표현
- 고정 소수점 체계보다 넓은 범위의 수를 컴퓨터에서 표시

● 10진 부동 소수점 수

- 2진 부동 소수점 수보다 인간에게 친밀
- 10진 부동 소수점 수 체계

$$x = \sigma \cdot \bar{x} \cdot 10^e$$

- σ : +1 (양, 대개 생략) 혹은 -1 (음) 의미
- e : 소수점 위치 표시 지수 (정수)
- \bar{x} : 범위 $1 \leq \bar{x} < 10$
 - ✓ 가수(mantissa), 유효숫자 자릿수, 부동 소수의 정밀도 표시

2진 부동 소수점 수와 10진 부동 소수점 수(2/6)

- 분수 $\frac{40}{3}$ 표시: $\frac{40}{3} = +1 \cdot (1.33333\cdots)_{10} \cdot 10^1$
- 정수 1011 표시: $(1.011)_{10} \cdot 10^3$

● 2진 부동 소수점 수

- 컴퓨터 기본 연산 체계
 - 2진 부동 소수점 수 체계
- $$x = \sigma \cdot \bar{x} \cdot 2^e$$
- 가수의 범위: $\bar{x} = (1)_2$
 - $(0.1)_{10}$ 표시: 2진 부동 소수점 수 변환

$$(0.1)_{10} = (.0001100110011\cdots)_2$$

$$(1.10011001100\cdots)_2 \cdot 2^{-4}$$

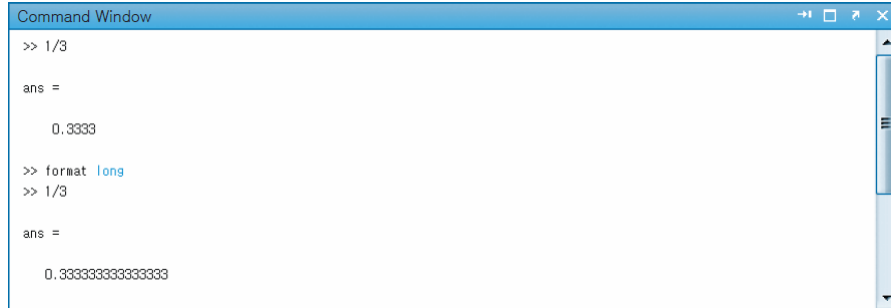
2진 부동 소수점 수와 10진 부동 소수점 수(3/6)

● 매트랩을 이용한 부동 소수점 표시 예

- 매트랩에서 분수 $\frac{1}{3}$ 을 표시: `format long` 지정
- $(3.333333333333333)_{10} \cdot 10^{-1}$
- 더 많은 순환 소수 3을 표시
- 실제값에 더욱 근접하여 오차 발생 줄임
- ✓ 수치해석을 배우는 가장 큰 이유: 오차를 줄여 실제값에 접근한 근삿값 추정

2진 부동 소수점 수와 10진 부동 소수점 수(4/6)

[매트랩을 이용한 부동 소수점 표시 예]



```

Command Window
>> 1/3

ans =

    0.3333

>> format long
>> 1/3

ans =

    0.3333333333333333
  
```

2진 부동 소수점 수와 10진 부동 소수점 수(5/6)

- 컴퓨터에 저장되는 2진 부동 소수점 수 x

$$FL(x) = \sigma \cdot (1b_1b_2b_3 \cdots b_{23})_2 \cdot 2^e$$

- $FL(x)$: x 를 컴퓨터 숫자로 변형시켜 처리한 부동 소수점의 근삿값
- 컴퓨터 저장 크기
 - σ : 1 비트
 - 가수: 23 비트 -> (2진수 표시) 1 비트 + (부동 소수점 자리) 23 비트
 - 지수 표시: 8 비트 -> (음과 양 표시) 1 비트 + (나머지 정수) 8 비트
- 지수의 범위: $-(1111111)_2 \leq e \leq +(1111111)_2$
- 지수의 실제 범위: $-126 \leq e \leq +127$ (지수 0인 2^0 포함)

2진 부동 소수점 수와 10진 부동 소수점 수(6/6)

● 기계 입실론(machine epsilon), ε

- 컴퓨터로 계산할 때 정밀도 표시 인자
- 1보다 큰 수 표현하는 소수점 이하의 자릿수
- 식 $FL(x) = \sigma \cdot (1.b_1b_2b_3 \cdots b_{23})_2 \cdot 2^e$ 의 형태로 1을 가장 간단한 2진 부동 소수점 수로 표현

$$1 = \underbrace{(1.00 \cdots 0)}_{23\text{비트}}_2 \cdot 2^0$$

부동 소수점 수의 끝수처리와 잘라버리기(1/10)

● 10진 연산의 끝수처리와 잘라버리기

: 10진 유효숫자가 n 자리인 부동 소수점 수

$$x = \sigma \cdot (d_1.d_2d_3 \cdots d_n)_{10} \cdot 10^e$$

- d_1 : 0이 아닌 1~9 사이의 10진 정수
 - $d_2 \sim d_n$: 0을 포함한 0~9 사이의 10진 소수
 - 식 $x = \sigma \cdot (d_1.d_2d_3 \cdots d_n)_{10} \cdot 10^e$ 의 일반적인 수의 형태
- $$x = \sigma \cdot (d_1.d_2d_3 \cdots d_nd_{n+1} \cdots)_{10} \cdot 10^e$$

부동 소수점 수의 끝수처리와 잘라버리기(2/10)

● 잘라버리기(chopping)

- 컴퓨터 허용자리 n
- $n+1$ 의 자리인 d_{n+1} 부터 오른쪽 자리 잡은 모든 부동 소수 자리 버림

● 잘라버리기 이후 컴퓨터 저장 형식

$$FL(x) = \sigma \cdot (d_1.d_2d_3 \cdots d_n)_{10} \cdot 10^e$$

● 올림 끝수처리(rounding up)

- 컴퓨터 허용자리 n
- $n+1$ 의 값이 5~9 사이: d_n 값에 1을 더함

부동 소수점 수의 끝수처리와 잘라버리기(3/10)

● 내림 끝수처리(rounding down)

- 컴퓨터 허용자리 n
- $n+1$ 의 값이 0~4 사이: d_n 값은 그대로 둔다
- d_{n+1} 부터 오른쪽에 있는 모든 부동 소수점 수 생략

● 끝수처리 이후 컴퓨터 저장 형식

$$FL(x) = \begin{cases} \sigma \cdot (d_1.d_2d_3 \cdots d_n)_{10} \cdot 10^e & (d_{n+1} < 5) \\ \sigma \cdot \left[(d_1.d_2d_3 \cdots d_n)_{10} + \underbrace{(0.00 \cdots 1)}_n \right]_{10} \cdot 10^e & (d_{n+1} \geq 5) \end{cases}$$

- $(0.00 \cdots 1)_{10}$ 에 대한 식: 10^{-n+1}

부동 소수점 수의 끝수처리와 잘라버리기(4/10)

● 2진 연산의 끝수처리와 잘라버리기

➤ 식 $x = \sigma \cdot \bar{x} \cdot 2^e$ 이용 다시 표현

$$x = \sigma \cdot (1.b_2b_3 \cdots b_nb_{n+1} \cdots)_2 \cdot 2^e$$

● 식 $x = \sigma \cdot (1.b_2b_3 \cdots b_nb_{n+1} \cdots)_2 \cdot 2^e$ 의 잘라버리기

➤ 컴퓨터 허용자리 n

➤ $n+1$ 의 자리인 b_{n+1} 부터 오른쪽 자리 잡은 모든 부동 소수 자리 버림

● 잘라버리기 이후 컴퓨터 저장 형식

$$FL(x) = \sigma \cdot (1.b_2b_3 \cdots b_n)_2 \cdot 2^e$$

부동 소수점 수의 끝수처리와 잘라버리기(5/10)

● 식 $x = \sigma \cdot (1.b_2b_3 \cdots b_nb_{n+1} \cdots)_2 \cdot 2^e$ 의 올림 끝수처리

➤ 컴퓨터 허용자리 n

➤ $n+1$ 의 값이 1: b_n 값에 1을 더한다

● 식 $x = \sigma \cdot (1.b_2b_3 \cdots b_nb_{n+1} \cdots)_2 \cdot 2^e$ 의 내림 끝수처리

➤ 컴퓨터 허용자리 n

➤ $n+1$ 의 값이 0: b_n 값은 그대로 둔다

➤ b_{n+1} 부터 오른쪽에 있는 모든 부동 소수점 수 생략

● 끝수처리 이후 컴퓨터 저장 형식

$$FL(x) = \begin{cases} \sigma \cdot (1.b_2b_3 \cdots b_n)_2 \cdot 2^e & (b_{n+1} = 0) \\ \sigma \cdot \left[(1.b_2b_3 \cdots b_n)_2 + \underbrace{(0.00 \cdots 1)}_n \right]_2 \cdot 2^e & (b_{n+1} = 1) \end{cases}$$

➤ $(0.00 \cdots 1)_2$ 에 대한 식: 2^{-n+1}

부동 소수점 수의 끝수처리와 잘라버리기(6/10)

문제 4. 부동 소수점 수 변환하기

10진 부동 소수점 $(89.625)_{10}$ 을 손으로 계산하여 2진 부동 소수점 수로 변환하고, 매트랩을 이용해서 확인하라. 그리고 10진 부동 소수점 수 $(0.7)_{10}$ 은 매트랩만을 이용해서 구하라.

Tip !

✓ 10진 정수 89를 2진 정수로 변환

부동 소수점 수의 끝수처리와 잘라버리기(7/10)

문제 5. 끝수처리

문제 4에서 구한 값을 각각 $x = (1011001.101)_2$ 와 $y = (.10110011001100\cdots)_2$ 라고 하자. 5비트 정밀도를 허용하는 컴퓨터에 맞게 끝수처리를 이용하여 x 와 y 를 근삿값으로 나타내라.

Tip !

✓ 식 $x = \sigma \cdot (1.b_2b_3 \cdots b_nb_{n+1} \cdots)_2 \cdot 2^e$ 형태 표시

부동 소수점 수의 끝수처리와 잘라버리기(8/10)

문제 6. 잘라버리기

2비트 2진 컴퓨터를 이용하여 다음을 계산한다고 가정한다.

$$4 + 2 + 1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{8}$$

잘라버리기를 이용하여 왼쪽에서 오른쪽으로 더하라.

Tip !

✓ 식 $x = \sigma \cdot \bar{x} \cdot 2^e$ 조건에 맞추어서 푼다.

부동 소수점 수의 끝수처리와 잘라버리기(9/10)

문제 7. 잘라버리기

문제 6에 주어진 계산을 잘라버리기를 이용하여 오른쪽에서 왼쪽으로 더하라.

Tip !

- ✓ 수학에서 왼쪽에서 오른쪽으로 더할 때와 오른쪽에서 왼쪽으로 더할 때의 결과는 항상 같다.
- ✓ 2비트 2진 컴퓨터라는 조건 아래에서는 다른 결과를 얻는다.

부동 소수점 수의 끝수처리와 잘라버리기(10/10)

문제 8. 부동 소수점 수 구하기

끝수처리 기능을 가진 컴퓨터의 정밀도를 유효숫자는 24($n=24$), 지수의 최소 정수값은 $-126(m=-126)$, 최대 정수값은 $127(M=127)$ 이라고 가정하자. 이 컴퓨터에 저장되는 0이 아닌 부동 소수점 수는 다음과 같은 정규형으로 나타낼 수 있다.

$$F: \{ "0", \pm (1.b_1 \dots b_{23})_2 \times 2^e \} \quad (e \in [-126, +127])$$

- (a) 가장 큰 부동 소수점 수를 구하라. $(1.1 \dots 1)_{23} \cdot 2^{127}$
- (b) 가장 작은 양의 부동 소수점 수를 구하라. $(1.1 \dots 1)_{23} \cdot 2^{-126}$
- (c) 주어진 정규형 조건을 이용하여 나타낼 수 있는 조합 가능한 모든 부동 소수점 수의 개수를 찾아라. $2^{23} (254)(2) + 1$

계산상의 오차

● 오차의 정의

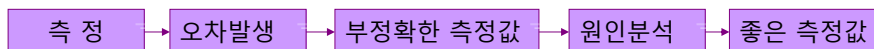
- 실제값 x 와 근삿값 \hat{x} 의 차이, $x - \hat{x}$
- 오차 발생 없다: $x - \hat{x} = 0$

● 오차 발생 원인

- 수학적 모델링 계산 과정의 오류: 오차 줄이기 어렵다.
- 컴퓨터 사용에 의한 프로그래밍 오류: 오류 발견과 수정이 어렵다.
- 실험의 측정에서 발생하는 작은 오차의 부주의한 제거

● 오차의 영향력의 최소화

- 전체 계산의 정확도를 향상



절대 오차와 상대 오차(1/4)

● 절대 오차

: 실제값과 근삿값 사이의 절대값

$$\text{절대 오차} = |x - \hat{x}|$$

- 잘라버리기 적용 오차: 항상 양의 값
- 올림 끝수처리 적용 오차: 항상 음의 값
- 내림 끝수처리 적용 오차: 항상 양의 값

● 상대 오차

: 계산된 오차를 실제값으로 나눈 뒤 절댓값을 적용

$$\text{상대 오차} = \left| \frac{x - \hat{x}}{x} \right|$$

- 비교 대상에 의해서 크거나 혹은 작게 느껴지는 오차
- 절대 오차가 작으면 상대 오차도 작다.

절대 오차와 상대 오차(2/4)

문제 9. 절대 오차와 상대 오차 구하기

유효숫자 여섯 자리($n=6$), 지수의 최소 정수값 -40 ($m=-40$), 최대 정수값 40 ($M=40$)과 끝수처리 기능이 있는 컴퓨터가 있다.

- (a) 주어진 조건을 고려하여 부동 소수점 수 $2.99792458 \times 10^{10}$ (1초당 센티미터로 표시한 빛의 속도)와 $1.67492716 \times 10^{-24}$ (그램 단위로 표시한 중성자의 무게)의 근삿값을 구하라.
- (b) 절대 오차와 상대 오차를 구하라.

절대 오차와 상대 오차(3/4)

문제 10. **round** 함수로 절대 오차와 상대 오차 구하기

맷랩을 이용하여 문제 9에서 제시한 $2.99792458 \times 10^{10}$ (1초당 센티미터로 표시한 빛의 속도)에 대한 절대 오차와 상대 오차를 구하라.

Tip !

- ✓ **round** 함수: 올림 혹은 내림 실행, 제한적 사용
- ✓ 지수승을 제외한 부동 소수점 수에 대한 끝수처리 먼저 실행
- ✓ 지수승을 다시 곱해서 표현
- ✓ **abs** 함수 : 절대값 구하기

절대 오차와 상대 오차(4/4)

문제 11. 부동 소수점의 근삿값 구하기

(a) 문제 3에서 10진 부동 소수점 수 $x = (89.625)_{10}$ 과 $y = (.7)_{10}$ 에 대한 2진 부동 소수점 수를 각각 $x = (1011001.101)_2$ 와 $y = (.101100110...)_{2}$ 로 변환하였다. 5비트 정밀도와 끝수처리 기능이 있는 컴퓨터를 이용하여 변환된 x 와 y 의 2진 부동 소수점의 근삿값을 구하라.

(b) 10진 정수 x 와 y 의 절대 오차와 상대 오차를 유효숫자가 두 자리인 부동 소수점 수로 나타내라.

유효숫자 오차의 손실(1/7)

유효숫자 (significant figure)

: 0이 아닌 숫자

- 22는 유효숫자가 두 자리이고 22.3은 유효숫자가 세 자리
- 오차가 작게 혹은 크게 발생

함수 $f(x) = x[\sqrt{(x+1)} - \sqrt{x}]$ 를 구하는 연산 과정

- 계산기는 여섯 자리의 10진수까지 사용 가능
- x 의 값을 1부터 10배씩 증가시켜 100000까지 입력
- $x = 1$ 일 때 연산 과정

1. x 에 1을 더하고 제곱근을 구한다.
2. x 에 대한 제곱근을 구한다.
3. 과정 ①에서 구한 제곱근에서 과정 ②에서 구한 제곱근을 뺀다.
4. 과정 ③의 계산 결과에 x 의 값을 곱한다.

유효숫자 오차의 손실(2/7)

- 모든 과정이 끝난(④) 이후에 끝수처리 적용

$$f(1) = 1 \cdot [\sqrt{(2)} - \sqrt{1}] = [1.4142135623 - 1] = 0.4142135623 = 0.414214$$

- 과정마다 끝수처리 적용

$$f(1) = 1 \cdot [\sqrt{(2)} - \sqrt{1}] = [1.41421 - 1] = 0.414210$$

- [표 1] 오차 손실 보기: 유효숫자를 증가시킨 결과

x	근삿값 $f(x)$	실제값 $f(x)$
1	0.414210	0.414214
10	1.54340	1.54347
100	4.99000	4.98756
1000	15.8000	15.8074
10000	50.0000	49.9988
100000	100.000	158.113

- 입력값 증가: 근삿값과 실제값 사이의 엄청난 오차 발생
- 유효숫자 증가: 오차의 손실을 최소화

유효숫자 오차의 손실(3/7)

문제 12. 유효숫자의 기초 개념

다음에 주어진 10진수의 유효숫자를 구하라.

0.046, 7.90, 8200

0이 아닌 수들의 뒤에 있는 부동 소수점 0은 유효숫자로 처리

0.046 : 4와 6 두자리

7.90 : 7, 8, 0 세자리

8200 : 네자리, 두자리, 세자리 모두

유효숫자 오차의 손실(4/7)

정밀도

- 유효숫자의 허용 범위가 낮아서 발생하는 오차를 줄이기 위해 계산기에 설정
- 정밀도 조정은 쉽지 않다.

함수 $f(x) = \frac{1 - \cos x}{x^2}$ ($x \neq 0$)를 구하는 연산 과정

- 계산기는 유효숫자 열 자리만 표시
- x 의 호도값 (radian value)을 0.1부터 10배씩 감소시켜 0.00001까지 입력
- $x = 0.01$ 일 때 연산 과정

- ① $\cos(0.01)$ 의 값을 구한다.
- ② 1에서 과정 ①의 결과를 뺀다.
- ③ (0.01) 의 제곱값을 구한다.
- ④ 과정 ②를 과정 ③으로 나눈다.

유효숫자 오차의 손실(5/7)

- 모든 과정이 끝난(④) 이후에 끝수처리 적용

$$f(0.01) = \frac{1 - \cos(0.01)}{(0.01)^2} = \frac{1 - 0.9999500004166652}{0.0001} = 0.4999958333$$

- 과정마다 끝수처리 적용

$$f(0.01) = \frac{1 - \cos(0.01)}{(0.01)^2} = \frac{1 - 0.9999500004}{0.0001} = 0.4999960000$$

- [표 2] 오차 손실 보기: 입력값을 감소시킨 결과

x	근삿값 f(x)	실제값 f(x)
0.1	0.4995834700	0.4995834722
0.01	0.4999960000	0.4999958333
0.001	0.5000000000	0.4999999583
0.0001	0.5000000000	0.4999999996
0.00001	0.0	0.5000000000

- 오차를 줄이기가 쉽지 않다.

유효숫자 오차의 손실(6/7)

- 유효숫자 변환 없이 $f(x) = \frac{1 - \cos x}{x^2}$ 오차 줄이기

- 삼각함수의 유사성을 이용한다.

$$\cos(2\theta) = 2\cos^2(\theta) - 1 = 1 - 2\sin^2(\theta)$$

- $x = 2\theta$ 로 치환

$$\begin{aligned} f(x) &= \frac{1 - \cos x}{x^2} = \frac{2\sin^2(x/2)}{x^2} \\ &= \frac{1}{2} \left[\frac{\sin(x/2)}{x/2} \right]^2 \end{aligned}$$

- $x = 0.01$ 을 수정된 식에 다시 입력

$$\begin{aligned} f(0.01) &= \frac{1}{2} \left[\frac{\sin(0.01/2)}{0.01/2} \right]^2 = \frac{1}{2} \left[\frac{0.004999979167}{0.005} \right]^2 = \frac{1}{2} [0.9999916668] \\ &= 0.4999958334 \end{aligned}$$

유효숫자 오차의 손실(7/7)

문제 13. 유리화를 이용한 유효숫자의 처리

방정식 $ax^2 + bx + c = 0$ 의 근을 구하기 위해 보편적으로 사용하는 공식은 다음과 같다.

$$x_{\pm} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \quad (a)$$

식 (a)의 분자와 분모에 $-b \mp \sqrt{b^2 - 4ac}$ 를 곱하면 다음과 같은 공식을 얻는다.

$$x_{\pm} = \frac{-2c}{b \pm \sqrt{b^2 - 4ac}} \quad (b)$$

3자리 유효숫자로 끝수처리 연산을 실행하는 컴퓨터로 식 (a)와 (b)를 각각 사용하여 이차방정식 $x^2 - 15x + 1 = 0$ 의 근을 계산하라.

언더플로우 오류와 오버플로우 오류

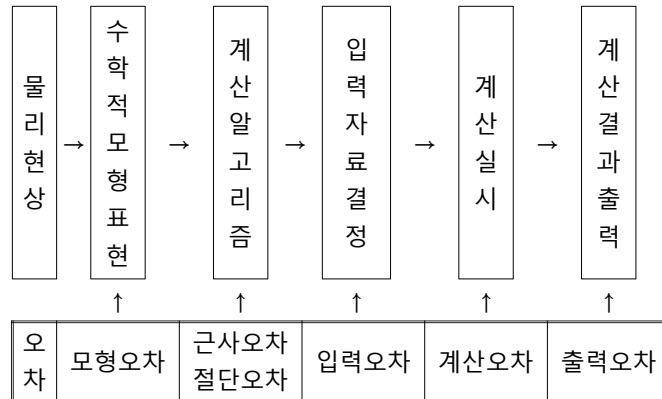
● 언더플로우 오류(underflow error)

- 부동 소수점 수가 너무 작아 컴퓨터에서 허용하는 범위가 못 되는 오류
- [문제9]에서 지수의 최소 정수값 $-20(m=-20)$ 으로 수정
- 중성자 경우: 근삿값의 지수승은 10^{-20} 보다 작아 언더플로우 오류 발생

● 오버플로우 오류(overflow error)

- 부동 소수점 수가 너무 커서 컴퓨터에서 허용하는 지수의 상한값을 넘겨서 생기는 오류
- [문제9]에서 지수의 최대 정수값 $5(M=9)$ 으로 수정
- 빛의 속도 경우: 근삿값의 지수승은 10^5 보다 커 오버플로우 오류 발생

● 수치해석의 과정 중 발생하는 각 단계 별 오차



● 모형오차

물리 현상을 수학적 모형으로 표현하는 경우, 예를 들면 엄밀하게는 비선형이어도 선형으로 다루는 등, 보다 단순하게 모형을 구성하기 위해서 생기는 오차를 **모형오차**(**방정식 오차**라고도 함)라고 함

● 근사오차

수학적 모형을 계산 가능하게 하기 위해서 근사식으로 계산하는 경우에 발생하는 오차를 **근사 오차**라고 함.

예) 정 적분의 계산을 누적 계산으로 근사하는 때나, 미분방정식을 차분방정식으로 변환하는 경우 근사 계산이 되고, 그 때문에 근사오차가 발생함

- 절단오차

수학적으로 엄밀하게 주어지는 함수의 값을, 유한의 사칙 연산의 반복 계산식으로 근사하는 경우,의 오차가 생긴다. 이것을, **절단 오차** (truncation error)라 함.

예) 테일러급수를 이용하여 삼각함수를 계산할 때, 무한급수의 계산을 유한 항까지의 계산으로 중단하기 위해서, 절단오차가 발생.

- 입력오차

관측한 입력자료의 경우는, 관측기의 정밀도에 따르는 오차나 읽어서 취하는 오차 등의 **관측 오차**가 생긴다. 또, 유한의 수로 입력 자료를 표현하기 위해서 근사 오차가 생긴다. 이것들은, 계산시의 **입력 오차**가 된다.

- 계산오차

계산 과정에서, **반올림, 항 소거, 오차의 전달** 등에 따라 발생하는 오차를 계산오차라 함.

① 반올림의 오차 : 계산기에 따라 수치 표현은 유한의 수로 표현되기 때문에, 사칙 연산때마다 그 결과는 유한의 수로 표현된다. 유한의 수로 반올림하는 경우에 생기는 오차를 **반올림 오차**(round-off error)라고 함. 반올림하는 데는, 계산 처리가 간단한 **절사**를 일반적으로 이용하지만, **사사오입**을 이용하는 계산기도 있음.

② 항 소거의 오차 : 근접하는 두 개의 수의 뺄셈인 경우에 유효수를 잃어버리는 현상을 **항 소거**(cancellation)라 하며, 이 때 생기는 오차를 **항 소거 오차**라 함.

③ 오차의 전달에 따르는 오차 : 주어졌던 변수에 포함되는 오차가, 계산 과정으로 계속해서 전해지고, 최후의 결과에 이르는 모습을 **오차의 전달**(propagation of errors)이라 함

- 출력 오차

계산 결과를 출력하는 단계에서는, 2진법이나 16진법으로 표현되는 계산기내의 수치를 10진법으로 변환하는 경우의 변환 오차나, 출력하는 수가 실제의 수보다 적어서 생기는 오차 등의 **출력 오차**가 발생함.