

Погрешности округлений в компьютерах

СРВ может работать только с целыми числами: +, -, сравнение.
 Операции /, *, и др. и работа с веществ. числами реализуется
 при спец. вычислит. методах, содержащихся в команд. сим,
 "защитках" в Co-процессор и в спец. библиотеках разных языков.

Приблизит. представление и операции с веществ. числами
 - основной источник погрешностей вычислений.

Вычислит. методы должны строиться с наименьшим
влиянием и без накопления вычисл. погрешностей.

Целое с плавающей точкой ^{с плав. точкой}
 Расселит. число представление, веществ. чисел при помощи
 целых чисел в 10-ичной системе:

$$x = 0.463 = 4 \cdot 10^{-1} + 6 \cdot 10^{-2} + 3 \cdot 10^{-3}$$

остаток
исчисления

точка \equiv кат-во
цифр

$$x = 46.3 = (4 \cdot 10^{-1} + 6 \cdot 10^{-2} + 3 \cdot 10^{-3}) \cdot 10^2$$

дробная часть - запятая

показатель

В общем виде: (для любой системы исчисления)

$$x = \pm (d_1 z^{-1} + d_2 z^{-2} + \dots + d_n z^{-n}) z^e$$

Целые числа $d_i, n, e_{\min}, e_{\max}$ определяют число
 с плавающей точкой, целые числа d_i зависят от системы
 исчисления:

$$0 \leq d_i \leq z-1.$$

$$z=2: d_i = 0, 1$$

$$z=10: d_i = 0, 1, 2, \dots, 9$$

$$z=16: d_i = 0, 1, 2, \dots, F$$

Т.к. n, e_{\min}, e_{\max} - ограничены ($\neq \infty$) \Rightarrow ограниченность
 точности и диапазона веществ. чисел. В ПК фирмы IBM
 $z=16$, а n, e могут изменяться \Rightarrow шестнадцатеричная точка

Т.о. ПК не может работать с бесконечными и беск.
 большими вещественными \Rightarrow погрешности вычислений.

C#:
 float - 4 байта - 7 знач. цифр
 double - 8 байт - 15 знач. цифр

Delphi:
 real, double - 8 байт - 15 знач. цифр
 extended - 10 байт - 19 знач. цифр

Пример погрешностей при работе с вещественными числами

Важнейшей e^x при $x < -1$. Элементы ряда даются в виде сходящегося ряда ^{Можно считать} с веществ. числами:

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$$

С. 77 Форсайт изр. Матем. анализ или. Выходящий

Рассчитаем "ПК" с $z=10$, $n=5$: еще $x=-5.5$:

$e^{-5.5}$	$n=5$	$= 1.0000$	$\rightarrow 1$
		-5.5000	$\rightarrow x$
		$+15.125$	$\rightarrow x^2/2!$
		-27.730	$\rightarrow x^3/3!$
		$+38.129$	$\rightarrow x^4/4!$
		\dots	
		-3.4902	$\rightarrow x^{11}/11!$
		$+1.5997$	$\rightarrow x^{12}/12!$
		$0.0000\dots$	$\rightarrow x^{25}/25!$
		$+0.0026363$	

Отбрасывание знаков

$\rightarrow x^{25}/25!$ - след. слагаемое не может скомпенсировать при точности $n=5$.

Точное = 0.0040867

В слагаемых 2-го порядка происходит округление 3-го знака после запятой, т.е. в порядке -3 , но результат имеет тот же -3 порядок \Rightarrow погрешность округлений соизмерима с конечным резу-том! Но самое плохое -

потеря верных знаков

Катастрофическая

В слагаемых 2-го порядка, а эти 3-го порядок этих знаков соизмерим с порядком результата. \Rightarrow Если порядок младших значащих цифр в промежуточных вычислениях соизмерим с порядком старших значащих цифр в конечном резу-те \Rightarrow вычислит. алгоритм построена некорректно или недостат. точность (n) вычислений, т.е. необходимо проводить расчеты с двойной/кварт. точностью или методом Аунда

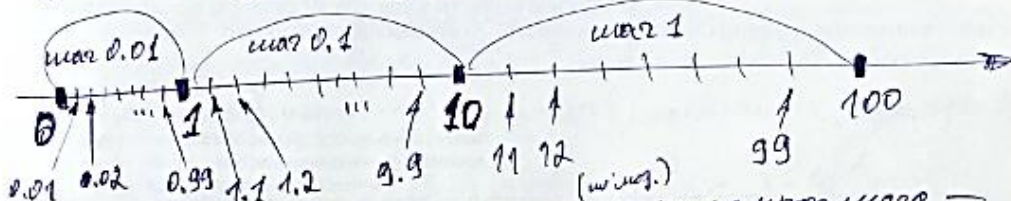
$$e^{-5.5} = \frac{1}{e^{5.5}} = \frac{1}{1+5.5+15.125+\dots} = 0.0040865 - \text{Начальная цифра}$$

В крайнем случае целесообразно контролировать так порядок чисел и анализировать их с порядком и точностью результата.

Дискретность чисел с плавающей запятой (точкой)

предел $n = 2, e_{min} = -2, e_{max} = 0$

диапазон $.\overline{99} \div \overline{99} = 0.01 \div 99.$



Вещнозначные числа меньше шести десятичного шага

Машинный ноль

{ т.е. всё, что ≤ 0.01 (по модулю) }

{ В числе шаг ∞ - это всё, что больше 99 }

Машинная бесконечность

Если в результате резултат $<$ шаг $\infty \Rightarrow$ исчерпание

Если $\dots >$ шаг $\infty \Rightarrow$ переполнение
погрешность!

В C# имеется специальное средство checked / unchecked, связанное с генерированием исключений, возникающих при переполнении в арифметических вычислениях:

```
try
{
    checked { ... }
}
catch (OverflowException) { Console.WriteLine("Переполнение\n\n"); }
```

т.о. погрешность возникает с глав. точкой складываться и т.д.

- 1 - исчерпание
 - 2 - переполнение
 - 3 - округление
 - 4 - катастрофич. потеря верных зн.
- } характерно для * /
} характерно для + -

- это всё и т.д. нецелого, дискретного представления вещнозначных чисел.

Нормальная форма:

мантисса $\in [0, 1)$, т.е.

$0 \dots E_{min}$ здесь для

float $\epsilon = 3.4E-38$

Нормализованная

мантисса $\in [1, 10)$, т.е.

$N \neq 0 \dots E_{min}$ здесь $\epsilon = 1.4E-45$

т.к. целая часть в двоичн. виде всегда всегда "1", E_{min} не может и поэтому можно не учитывать

Погрешности вычислительных алгоритмов

Внешне проявляются в виде абсурдного результата при формально правильной программе вычислений.

Причины:

1. Высокая чувствительность

Высокая чувствительность

Прим. 2. $ax^2 + bx + c = 0$, \exists Теорема, что

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

Возмем для этих формул a, b, c не заданные значения результата.

Пусть $n = 10$,

$$n = 7, \quad e_{\min} = -50, \quad e_{\max} = 50$$

Случай 1. $a = c$ - порядок 1, b - порядок 5

$$\Rightarrow b^2 \sim 10^5, \quad 4ac \sim 2 \Rightarrow b^2 \ominus 4ac$$

$$\Rightarrow x_1 = 0 \text{ - неверно!}$$

Катастрофическая потеря верных знаков у b

\Rightarrow либо возмущать дискриминант с повышенной точностью, либо менять алгоритм (сначала возмущать x_2 , затем x_1 через x_2).

Случай 2. $a, b, c \sim 30 \Rightarrow$ нормальные функции b^2

\Rightarrow повышенной точности или менять алгоритм

$$\sim 60 > e_{\max}$$

2. Алгоритмы с накоплением погрешностей

Такое накопление происходит из-за некорректно построенных рекуррентных соотношений, например,

$$E_{i+1} = i * E_i, \quad i = 1, 2, 3, \dots, N \text{ или } i \geq 1$$

Здесь может присутствовать ошибка округления только в расчёте E_1 , далее эта ошибка увеличивается экспоненциально на i . Чем больше N - тем абсурднее результат.

\Rightarrow либо увеличивать точность расчёта E_1 , либо менять алгоритм, например, если известно E_N , то можно найти E_1 :

$$E_i = \frac{E_{i+1}}{i}, \quad i = N-1, N-2, \dots, 1$$

При этом погрешность возмущения E_N не так и велика, а уменьшается движением на i .

3. Некорректность математической постановки задачи