2020.03.18

О вычислительной погрешности и ее контроле.

О чем речь

Известно, что внутреннее представление вещественных чисел в ЭВМ предполагает наличие погрешности представления, поскольку точно представляется только определеное подмножество чисел (см., например, стандарт IEEE-754), другие числа округляются до ближайшего представимого.

Эта погрешность представления имеет относительный характер, т.е. ее величина непосредственно связана с величиной самого числа. Мы здесь не будем погружаться по тонкости способов представления вещественных чисел по действующим стандартам, а просто будем считать, что каждое число несет в себе относительную погрешность, и эта погрешность как-то эволюционирует в процессе математических вычислений.

Напомним базовые определения. Пусть есть некоторое число $\bar{a}$ и его приближение a. Тогда величина $\varepsilon = a - \bar{a}$ называется абсолютной погрешностью, а величина $\delta = \dfrac{|a-\bar{a}|}{|\bar{a}|}$ — относительной погрешностью. Иногда в определении абсолютной погрешности также ставят знак модуля, но это не меняет сути дальнейших рассуждений.

Нас интересует как будет трансформироваться погрешность исходных данных в погрешность результата при выполнении математических вычислений. Так как мы не можем предсказать величину и знак абсолютной погрешности какждого конкретного числа, то рассуждения придется вести в терминах оценок наихудшей ситуации.

Трансформация абсолютной погрешности.

Пусть мы имеем два числа, представленных с некоторой точностью: $a = \bar{a} + \varepsilon_a$ и $b = \bar{b} + \varepsilon_b$. Элементарные выкладки дают

$a + b = (\bar{a} + \bar{b}) + (\varepsilon_a + \varepsilon_b) = (\bar{a} + \bar{b}) + \varepsilon_{a+b}$;
$a - b = (\bar{a} - \bar{b}) + (\varepsilon_a - \varepsilon_b) = (\bar{a} - \bar{b}) + \varepsilon_{a-b}$;
$ab = (\bar{a}\bar{b}) + (\bar{a}\varepsilon_b + \bar{b}\varepsilon_a) + \varepsilon_b\varepsilon_a = (\bar{a}\bar{b}) + \varepsilon_{ab}$;

Т.е. при сложении и вычитании абсолютные погрешности суммируется, при умножении погрешность также может быть вычислена по очевидной явной формуле, а при делении она может быть оценена из следующих соображений

$$\frac{a}{b} = \frac{\bar{a}+\varepsilon_a}{\bar{b}+\varepsilon_b} = \frac{\bar{a}}{\bar{b}} \cdot \frac{1+\varepsilon_a/\bar{a}}{1+\varepsilon_b/\bar{b}} \approx \frac{\bar{a}}{\bar{b}}(1+\varepsilon_a/\bar{a})(1-\varepsilon_b/\bar{b}) = \frac{\bar{a}}{\bar{b}} + \Big(\frac{\varepsilon_a}{\bar{b}} - \frac{\varepsilon_b\bar{a}}{\bar{b}^2} - \frac{\varepsilon_a\varepsilon_b}{\bar{b}^2}\Big).$$

В действительности, мы не можем точно знать какой знак окажется у абсолютной погрешности конкретного числа в конкретный момент вычислений. Поэтому для оценки ожидаемой величины этой погрешности мы должны предусмотреть “самый плохой случай” и поставить во всех выражениях для погрешности знак $+$ и все значения взять по модулю.

Трансформация относительной погрешности.

Заметим, что относительную погрешность δ числа a можно интерпретировать в виде равенства $a = \bar{a}(1+\delta)$. Исходя их этого представления, для арифметических операци мы получаем

$$a \pm b = \bar{a}(1+\delta_a) \pm \bar{b}(1+\delta_b) = (\bar{a} \pm \bar{b})\Big(1 + \frac{\bar{a}\varepsilon_a \pm \bar{b}\varepsilon_b}{\bar{a} \pm \bar{b}}\Big);$$
$$ab = \bar{a}(1+\delta_a)\bar{b}(1+\delta_b) = ab(1+\delta_a+\delta_b+\delta_a\delta_b);$$
$$\frac{a}{b} = \frac{\bar{a}(1+\delta_a)}{\bar{b}(1+\delta_b)} \approx \frac{a}{b}\Big(1+\delta_a)(1-\delta_b) = \frac{a}{b}(1+\delta_a-\delta_b-\delta_a\delta_b\Big)$$

Так как нас интересует именно оценка относительной погрешности, то мы так же должны поставить модули и знак $+$ в выражениях для относительной погрешности. Обратите внимание, что для сложения и вычитания мы должны получить верхнюю границу для погрешности и поэтому взять максимально возможное значение числителя и минимально возможное значение

знаменателя. Таким образом, для этих операций оценка относительной погрешности приобретает вид $\delta_{a\pm b} = \dfrac{|\bar{a}||\varepsilon_a| + |\bar{b}||\varepsilon_b|}{|\bar{a} \pm \bar{b}|}$. Отсюда и сдедует хорошо известный вывод, что при вычитании больших и близких чисел относительная погрешность результата может возрасти катастрофически (числитель относительно большой, а знаменатель близок к нулю). Для умножения и деления относительная погрешность суммируется (с точностью до членов 2 порядка).

Непосредственный контроль вычислительной погрешности.

Формулы для погрешности, приведенные выше, позволяют контролировать погрешность в ходе вычислений, если параллельно с выполнением операций к конкретными числами также выполнять соответствующие операции с их оценками их погрешностей. Фактически мы можем сопоставить каждому числу пару, включающую текущее значение этого числа и текущую погрешность этого числа. В языке C есть естественное средство для работы с подобными парами чисел — это структурный тип данных. Таким образом, мы можем реализвать набор функций для четырех арифметических операций, эти функции будут получать на вход две структуры, представляющие собой аргументы операции (число и погрешность) и возвращать в качестве ответа соответствующий результат. Однако наиболее естественно такие построения выполняются в языке C++.

Язык C++ позволяет переопределить существующие операции для работы с другими, новыми типами данных. В нашем случае таким новым типом будет структура (число, погрешность). Здесь мы не будем углубляться в особенности конструкций языка C++, а просто проведем пример кода, который будет релизовывать описанные выше идеи и может быть понят исходя из этих идей.

Примеры кода далее будут использовать соглашения C++ и компилироваться они должны также компилятором C++ (например, g++).

Основным содержанием кода будет переопределение арифметических операций, операций присваивания и других. Определенную проблему представляет начальное задание погрешности числа. Мы будем считать, что для обычных переменных типа double эта погрешность соответствует относительной погрешности 10^{-15}.

Следует заметить, что при переопределении операций приходится рассматривать много разных сочетаний типов аргументов. (целый, вещественный и наш новый тип NumberA). Однако по сути речь идет только о четырех основных операциях +-*/. Остальные функции введены исключительо для согласовани типов аргументов и ссылаются на одну и ту же реализацию операций.

Пример кода приведен в отдельном файле number.zip.

В качестве тестовой задачи там приводится процедура вычисления определенного инетграла методом прямоугольников. Один вариант процедуры записан в терминах типа double, другой — в терминах типа NumberR, сохраняющего абсолютную погрешность числа. По форме эти две процедуры абсолютно идентичны, что как раз и является целью введения такого нового типа, поддерживающего все требуемые арифметические операции.

Упражнения.

1. Реализуйте структуру NumberR для числа с относительной погрешностью.

2. Реализуйте метод Гаусса с матрицами, элементы который есть подобные числа Number. Тогда проверка на нуль лиагонального элемента матрицы будет сводить к проверке погрешности этого элемента. Нетрудно понять, что если относительная погрешность числа оказадась больше единицы или абсолютная погрешность больше модуля этого числа, то точное значение такого числа вполне может быть нулем.

Если написана программа работы с матрицей в терминах типа double, то при правильной реалиации Number тот же самой код должен правильно работать и с типом Number, т.е. новый код получается из старого просто заменой doule на Number.