ИНТЕРВАЛЬНАЯ МАТЕМАТИКА И РАСПРОСТРАНЕНИЕ ОГРАНИЧЕНИЙ
МКВМ-2004 РАБОЧИЕ СОВЕЩАНИЯ С. 201–209

Интервальная математическая библиотека, основанная на разложениях в ряды Чебышева и Тейлора

А.Г. Ершов*, Т.П. Кашеварова†

Введение

Интервальный анализ, активно развиваемый в последние три десятилетия, выводит вычислительную математику на качественно новую ступень. Интервальные методы позволяют получать гарантированные интервальные оценки множеств решений в виде многомерных брусов. Кроме того, использование интервалов дает возможность корректно учитывать погрешности, вызываемые машинными округлениями.

Представление данных в виде интервалов вызывает необходимость разработки соответствующей математической библиотеки. В настоящее время существует несколько разработок интервальных математических библиотек, однако почти каждая из них, находящаяся в открытом доступе, обладает теми или иными недостатками. Так, например, для реализации библиотеки математических функций Pascal-XSC (см. [6]) используется целочисленная арифметика, которая позволяет получать точные результаты при операциях над числами с плавающей точкой, однако использование целочисленной арифметики значительно увеличивает время вычислений.

По этой причине авторами были разработаны и реализованы алгоритмы вычисления гарантированных верхних и нижних границ элементарных математических функций с помощью направленных округлений на основе разложений в ряды Чебышева и Тейлора. Эти алгоритмы позволяют обеспечить высокую точность определения верхних и нижних границ, а также производительность, не худшую по сравнению со стандартными математическими функциями библиотеки C. Разработанная интервальная математическая библиотека была реализована в рамках работ над проектом SibCalc (см. [7]).

1. Основные определения интервального анализа

Определим интервал над полем действительных чисел $\mathbb{R}$ следующим образом.

Определение. Множество

$$\boldsymbol{x} = [\underline{x}, \overline{x}] = \left\{ x \mid x \in \mathbb{R} \cup \{-\infty, +\infty\},\ \underline{x} \le x \le \overline{x} \right\}$$

*Институт систем информатики им. А.П. Ершова СО РАН, ЗАО “Ледас”, eag@sib3.ru
†Институт систем информатики им. А.П. Ершова СО РАН, toma@iis.nsk.su

будем называть интервалом на $\mathbb{R}$, а $\underline{x}$, $\overline{x}$, соответственно, его нижней и верхней границами. Кроме того, определим специальный интервал $\emptyset$, который не содержит ни одного элемента.

Если нижняя граница равна верхней, то интервал содержит только одно число, такой интервал называется вырожденным. Множество всех интервалов обозначим через $\mathbb{IR}$.

В данной работе мы будем использовать расширенную интервальную арифметику [1, 5]. Машинная реализация выполнения арифметических операций осуществлена с помощью вычислений с направленными округлениями.

Сложение и умножение в интервальной арифметике обладают ассоциативностью и коммутативностью, но не обладают дистрибутивностью. Для интервалов выполняется свойство, называемое субдистрибутивностью:

$$\boldsymbol{a} \cdot (\boldsymbol{b} + \boldsymbol{c}) \subseteq \boldsymbol{a} \cdot \boldsymbol{b} + \boldsymbol{a} \cdot \boldsymbol{c}.$$

Для операций над интервалами выполняется основное свойство интервальных вычислений – монотонность по включению, т. е. если $\boldsymbol{a}^{(k)}, \boldsymbol{b}^{(k)} \in \mathbb{IR}$ и $\boldsymbol{a}^{(k)} \subseteq \boldsymbol{b}^{(k)}$, $k = 1, 2$, тогда для любой операции справедливо

$$\boldsymbol{a}^{(1)} * \boldsymbol{a}^{(2)} \subseteq \boldsymbol{b}^{(1)} * \boldsymbol{b}^{(2)}.$$

2. Математические основы алгоритмов вычисления элементарных функций

Интервальная библиотека математических функций является базой для всех интервальных методов, поэтому от корректности ее реализации зависит гарантированность получаемых результатов.

При разработке алгоритмов вычисления математических функций стоят две главные задачи – точность и скорость. Интервальное представление позволяет получать нижнюю и верхнюю оценки искомой функции, то есть обеспечивает гарантированность вычислений, однако здесь возникает новая проблема — получение как можно более узких интервалов, в идеале, состоящих их двух подряд идущих машинопредставимых вещественных чисел. Время вычисления значений интервальных функций существенно влияет на эффективность вычислений в целом. Именно поэтому было принято решение использовать равномерные приближения Чебышева, поскольку они обеспечивают наименьшую абсолютную погрешность при фиксированном числе членов ряда [4].

Алгоритмы вычисления интервальных значений математических функций, реализованные в SibCalc, основаны на разложении функций в ряды Чебышева и Тейлора с корректным округлением результатов арифметических операций и оценкой остаточного члена. Для получения наиболее узких интервалов, гарантированно включающих математическое значение вычисляемой функции, был реализован ряд оригинальных приемов, позволяющих значительно уменьшить величину вычислительной ошибки. Ряды Тейлора используются только в окрестности нуля, так как в данном случае они обеспечивают более высокую относительную точность. На остальных участках разложения используются ряды Чебышева, поскольку они обеспечивают равномерное приближение и требуют меньшего числа членов ряда по сравнению с рядами Тейлора.

2.1. Многочлены Чебышева. Разложения по многочленам Чебышева обычно строятся в виде

$$f(x) = \sum_{n=0}^{\infty} a_n T_n(x), \quad x \in [-1, 1].$$

При этом x может быть некоторой линейной функцией: если необходимо рассматривать разложение функции на интервале $[a, b]$, то используется линейное преобразование [4]. В работе рассматриваются смещенные чебышевские многочлены $T_n^*(x)$, заданные на интервале $[0, 1]$. Для них существует рекуррентные формулы (см. [4]):

$$T_{n+1}^*(x) = 2(2x - 1)T_n^*(x) - T_{n-1}^*(x).$$

Для отрезка $[-1, 1]$, остаточный член разложения в ряд Чебышева $\omega_n(x) = \frac{1}{2^{n-1}} T_n(x)$ будет иметь наименьшее возможное значение $\sup |\omega_n(x)|$ среди всех многочленов данной степени. В этом случае оценка погрешности разложения $L_n(x)$ функции в ряд Чебышева примет вид (см. [2]):

$$\big|f(x) - L_n(x)\big| \le \frac{M_{n+1}}{2^n (n+1)!}, \quad \text{где } M_{n+1} = \sup_x \big|f^{(n+1)}(x)\big|.$$

2.2. Организация машинных вычислений. Вычисление элементарной функции на всей области определения сводится с помощью соответствующих формул приведения к вычислению ее на некотором, называемом каноническим, интервале. Для уменьшения вычислительных затрат канонический интервал делится на подинтервалы длиной 2^{-k}, на каждом из которых получаем свое разложение. Такой способ деления позволяет эффективно определять подинтервал, в который попадает аргумент функции, и провести линейное преобразование приведения к каноническому для смещенных многочленов Чебышева интервалу [0,1] без привнесения ошибки округления.

Вид частичной суммы ряда Чебышева можно преобразовать следующим образом:

$$f(x) = \sum_{i=0}^{n} a_i T_i(x) = \sum_{i=0}^{n} a_i \sum_{j=0}^{i} T_{i,j} x^j = \sum_{k=0}^{n} c_k x^k, \quad c_k = \sum_{i=k}^{n} a_i T_{i,k},$$

получив тем самым частичную сумму обычного степенного ряда. Коэффициенты разложений в степенные ряды на каждом подинтервале вычисляются заранее с необходимой точностью и хранятся в отдельных таблицах. Такой подход незначительно увеличивает объем памяти, зато существенно увеличивает скорость вычисления функции.

При вычислении элементарных функций с помощью разложения в ряд возникает три рода погрешностей:

1) погрешность, вызываемая использованием вместо бесконечного ряда его конечной частичной суммы. Эту погрешность можно оценить сверху с помощью формул оценки остаточного члена;

2) погрешность, вызываемая использованием вместо истинных значений коэффициентов полиномов их машинных приближений;

3) погрешность округления результатов арифметических операций.

Важным фактором в организации машинных вычислений частичной суммы ряда является порядок суммирования, поскольку он определяет величину погрешности третьего рода. Так как мы знаем вид элементарной функции и ее поведение на интервале, мы можем выбрать максимально эффективный порядок суммирования. Разложение в ряд элементарных функций таково, что наибольший вклад в сумму вносят члены низкого порядка, меньший – члены более высокого порядка, а наименьший – члены всех порядков выше некоторого заранее выбранного, сумму которых мы и оцениваем сверху с помощью известных формул оценки остаточного члена. Уменьшение величины членов частичной суммы с увеличением их порядка происходит достаточно быстро, как правило, соотношение между величинами двух соседних членов меньше величины интервала разбиения (в нашей реализации имеющей порядок 10^{-2}–10^{-3}). Оптимальным, таким образом, является суммирование частичной суммы, начиная с членов высокого порядка к членам более низкого порядка. В таком случае, ошибки округления, внесенные на более ранних этапах вычисления, становятся незначительными на более поздних этапах вычисления; в результате, общая погрешность вычисления третьего рода формируется почти исключительно при добавлении члена первого порядка и свободного члена. Заметим, что такому порядку вычисления соответствует известная и вычислительно эффективная (минимум операций с плавающей точкой) схема Горнера.

Для корректного осуществления вычисления верхней (нижней) границы элементарной функции с помощью разложения в ряд к частичной сумме ряда необходимо добавлять (отнимать) два дополнительных слагаемых: первое оценивает сверху погрешность первого рода (остаточный член ряда), второе оценивает сверху погрешность второго рода, возникающую из-за использования вместо истинных коэффициентов Чебышева и Тейлора их машинных приближений. Величина первого слагаемого имеет порядок главного члена в остатке ряда Чебышева, величина второго - порядок разности между истинным и точным значением свободного члена. Таким образом, отношение второго слагаемого к вычисляемой сумме имеет порядок машинной точности, а значит, учет этого слагаемого негативным образом влияет на ширину интервала между верхней и нижней оценками значения функции, во многих случаях не позволяя добиться желаемой ширины в один ulp (unit in last position — величина единицы в последнем разряде мантиссы). Для того, чтобы избежать этого эффекта, предлагается использовать следующие формулы вычисления верхней и нижней оценки (при условии $x \geq 0$ включены соответственно флаг округления вниз и вверх):

$$F_+(x) = c + (((((r[5] \times x + r[4]) \times x + r[3]) \times x + r[2]) \times x + r[1]) \times x + d_+),$$
$$F_-(x) = c + (((((s[5] \times x + s[4]) \times x + s[3]) \times x + s[2]) \times x + s[1]) \times x + d_-).$$

Здесь r и s – массивы, содержащие приближения коэффициентов ряда разложения с округлением вверх и вниз соответственно; $c + d_+ \geq a_0 \geq c + d_-$, a_0 – истинное значение свободного члена разложения, d_+ и d_- – два последовательно идущих вещественных числа, отношение d_+ к a_0 имеет порядок машинной точности.

При организованных таким образом вычислениях мы добиваемся того, что свободный член учитывается со значительно более высокой точностью, снижая тем самым погрешность второго рода, порядок которой становится равным произведению длины интервала разбиения на разность между истинным и точным значением коэффициента при x. Ценой за достигнутую точность вычислений является одна дополнительная операция сложения, расходы на выполнение которой не слишком значительны по сравнению с общим временем вычисления элементарной функции.

Рассмотренный выше прием имеет смысл, если наибольший по величине вклад в частичную сумму вносит именно свободный член — тогда представление его в виде большей и меньшей части оправдано. В противном случае, больший вклад вносит член при первой степени аргумента, и погрешность, вносимая им, превышает погрешность, вносимую неточным представлением свободного члена. На таких интервалах разбиения предлагается использовать следующие формулы:

$$F_+(x) = x + (((((r[5] \times x + r[4]) \times x + r[3]) \times x + r[2]) \times x + r[1]) \times x + r[0]),$$

$$F_-(x) = x + (((((s[5] \times x + s[4]) \times x + s[3]) \times x + s[2]) \times x + s[1]) \times x + l[0]).$$

Здесь мы разбиваем на два слагаемых уже член первой степени, используя в качестве большего слагаемого x и в качестве меньшего слагаемого $r[1] \times x$ (или $s[1] \times x$). Для этого есть три причины. Первая состоит в том, что разлагаемые функции (sin(), asin(), tan(), atan(), ln(1+x)) в областях, где значения функции существенно меньше единицы, достаточно хорошо приближаются функцией x – а значит, величина x существенно превышает остальные члены суммы. Вторая заключается в том, что представление члена первой степени в виде суммы большей и меньшей части $c \times x$ и $d \times x$ ведет к тому, что возникает погрешность третьего рода при вычислении произведения c и x, чего можно избежать, используя в качестве большей части x. Третья причина состоит в том, что, отказываясь от лишнего умножения на коэффициент, мы экономим время вычислений.

Существенным недостатком разложения в ряд Чебышева является тот факт, что частичная сумма ряда Чебышева обеспечивает наилучшее приближение на интервале в смысле абсолютного отклонения, но не в смысле относительного отклонения (см. [3]), которое наиболее важно для расчета элементарных функций. Поэтому на интервалах, где относительная погрешность существенно превышает абсолютную погрешность, используется разложение в ряд Тейлора. Однако почти для всех элементарных функций (sin(), asin(), tan(), atan(), ln(1 + x)) на области их разложения в ряд такого рода интервал разбиения только один — это интервал $[0, 2^{-k}]$ (для ln(1+x) еще есть интервал $[-2^{-k}, 0]$). Заметим, что вычисление этих функций в нуле с помощью разложения в ряд Чебышева привело бы к плачевным результатам — максимальное отклонение приближения Чебышева от истинной функции достигается как раз на концах интервала приближения, и в малой окрестности нуля относительная погрешность была бы очень большой. Кроме того, есть еще один аргумент в пользу использования на этом интервале разложения в ряд Тейлора – все эти функции разлагаются в ряд Тейлора в окрестности нуля только по нечетным степеням, что существенно экономит время вычислений.

2.3. Использование одного флага округления. Реализация флагов округления на многих компьютерных платформах такова, что время, затрачиваемое на изменение флагов округления, сравнимо со временем, затрачиваемым на вычисление элементарной функции как частичной суммы ряда Чебышева или Тейлора. В связи с этим в описываемой авторами реализации флаг округления выставляется один раз, до проведения вычислений, а вычисления верхней и нижней границ функций реализованы так, что они делаются при одном и том же флаге округления. Например, если выставлен флаг округления "округлять вверх", и $x \geq 0$, то вычисление верхней границы элементарной функции происходит по обычной формуле Горнера:

$$F(x) = (((((r[5] \times x + r[4]) \times x + r[3]) \times x + r[2]) \times x + r[1]) \times x + r[0]),$$

где $r[i]$ – коэффициенты полинома Чебышева, посчитанные для определения верхней границы, а вычисление нижней границы происходит по следующей модифицированной формуле Горнера:

$$F(x) = -((((((-s[5]) \times x - s[4]) \times x - s[3]) \times x - s[2]) \times x - s[1]) \times x - s[0]),$$

где $s[i]$ – коэффициенты полинома Чебышева, посчитанные для определения нижней границы.

Использование функций вычисления верхних и нижних границ с одним и тем же флагом округления позволяет эффективно осуществлять большие объемы вычислений. Кроме того, во избежание влияния вызова функций математической библиотеки на дальнейшие вычисления, все функции математической библиотеки имеют дубликаты, которые дополнительно меняют флаг округления на значение "округлять вверх" перед началом вычисления элементарной функции, и устанавливают исходный флаг округления после завершения вычисления.

3. Примеры интервальных алгоритмов вычисления элементарных функций

В качестве примеров приведем алгоритмы вычисления некоторых элементарных функций.

3.1. Вычисление функции синус. Разложение синуса в ряд проводится на интервале $[0, \pi/2]$ (см. [3]). Этот интервал разбивается на подинтервалы длиной 2^{-k}, на каждом подинтервале $x \in \left[\frac{i}{2^k}, \frac{i+1}{2^k}\right]$ с помощью преобразования $t = 2^k \times \left(x - \frac{i}{2^k}\right)$ задача о разложении синуса сводится к задаче о разложении функции $\sin\left(2^{-k} \times t + \frac{i}{2^k}\right)$ на каноническом интервале $t \in [0, 1]$. Для данной длины подинтервала достаточно использовать разложение в ряд Чебышева до пятой степени включительно. Заметим, что разложение в ряд Тейлора требует для достижения такой же точности при том же числе членов разложения использования в 4–8 раз большего числа подинтервалов, что ведет к 4–8-кратному увеличению объема используемой памяти.

Пусть, к примеру, $k = 7$ и $i = 30$, т. е. мы разлагаем $\sin(x)$ на промежутке $x \in \left[\frac{30}{128}, \frac{31}{128}\right]$. Мы вычисляем верхние и нижние оценки коэффициентов Чебышева и составляем коэффициенты разложения в степенной ряд для вычислений верхней и нижней границы синуса, в результате чего получаем формулы для приведенного аргумента $t = 128 \times \left(x - \frac{30}{128}\right)$:

$$\begin{aligned}
\sin_+(x) = {} & (((((8.097870612551672648 \cdot 10^{-3} \times t + \\
& 9.676530503240125813 \cdot 10^{-3}) \times t - 1.621099466468840788 \cdot 10^{-1}) \times t - \\
& 1.161175593052558991 \cdot 10^{-1}) \times t + 9.726596782449125067 \cdot 10^{-1}) \times t + \\
& 1.952311138366777654 \cdot 10^{-17}) + 0.232235118611511443, \\
\sin_-(x) = {} & -((((((-8.097870612551670913 \cdot 10^{-3} \times t - \\
& 9.676530503240124079 \cdot 10^{-3}) \times t + 1.621099466468841066 \cdot 10^{-1}) \times t + \\
& 1.161175593052559130 \cdot 10^{-1}) \times t - 9.726596782449123957 \cdot 10^{-1}) \times t - \\
& 1.942458581633222228 \cdot 10^{-17}) - 0.232235118611511443), \\
\sin(x) = {} & [\sin_-(x), \sin_+(x)].
\end{aligned}$$

Вычисления значений синуса для значений аргумента, выходящих за пределы интервала $\left[0, \frac{\pi}{2}\right]$, требуют применения формул приведения тригонометрического аргумента. Основной трудностью в них является корректное вычитание полных периодов тригонометрической функции для значений аргумента, достаточно удаленных от нуля. Обычно выбирается некоторый достаточно большой интервал, на котором значение тригонометрических функций желательно вычислять с высокой точностью (в разработанной библиотеке – $[-10^9, 10^9]$), а за пределами этого интервала высокая точность вычислений не требуется. Используя специальное представление числа $2\pi = \sum_{i=0}^{m} p_i$, где p_i удовлетворяют неравенству $8 \times 2^{-l \times i} \geq p_i \geq 4 \times 2^{-l \times i}$ и имеют мантиссу длины l, мы можем от тригонометрического аргумента вычесть величину $2\pi n = \sum_{i=0}^{m} p_i n$ так, чтобы вычисление произведений $p_i n$ происходило без ошибки округления. Для этого необходимо, чтобы сумма l и $\lceil \ln n \rceil$ — соответственно разрядностей мантисс p_i и n – не превышала разрядности мантиссы машинного представления. Заметим также, что разрядность мантиссы машинного представления, удовлетворяющему стандарту IEEE-754, равна 52, а для указанного выше интервала $[-10^9, 10^9]$ величина $\lceil \ln n \rceil$ ограничена сверху 28 — таким образом, l можно выбрать равным 24. Из практических соображений длина представления 2π ограничена величиной $m = 3$.

Вычисление функции синус для интервального значения аргумента состоит в том, чтобы найти глобальные минимум и максимум на заданном интервале. Понятно, что если ширина интервального значения аргумента x больше, чем 2π, то $\sin(x) = [-1.0, 1.0]$. Если ширина интервала-аргумента меньше, чем 2π, то определяются значения аргумента $x_{\min}$ и $x_{\max}$, в которых достигаются соответственно минимальное и максимальное значения функции синус на дан-

ном интервале. Это можно сделать используя свойство кусочной монотонности функции. Результатом вычисления синуса для невырожденного интервального аргумента будет интервал, нижняя граница которого равна нижней границе значения синуса в точке $x_{\min}$, а верхняя – верхней границе значения синуса в точке $x_{\max}$.

3.2. Вычисление логарифма. Разложение логарифма в ряд проводится на интервале $[2^{-1/2}, 2^{1/2}]$. Вне этого интервала мы используем формулу $x = 2^m y$ для выбора подходящего значения y из интервала разложения, вычисляем $\ln(y)$, а затем пользуемся соотношением $\ln(x) = \ln(y) + m\ln(2)$. Такое значение y всегда можно найти, поскольку отношение верхней и нижней границы интервала $[2^{-1/2}, 2^{1/2}]$ равно 2. Выбор этого интервала обладает также тем преимуществом, что он содержит значение 1.0, на котором логарифм обращается в нуль, что позволяет вычислять логарифм в окрестности 1.0 с высокой относительной точностью.

В силу того, что логарифм является монотонно возрастающей функцией, его вычисление для невырожденного интервального аргумента сводится к вычислению нижней границы на левом конце и верхней границы на правом конце интервала-аргумента.

3.3. Вычисление экспоненты. Разложение экспоненты в ряд проводится на интервале $[0, \ln 2]$. Вне этого интервала мы используем формулу $x = y + m \times \ln 2$ для выбора подходящего значения y из интервала разложения, вычисляем e^y, а затем пользуемся соотношением $e^x = 2^m e^y$. Заметим, что вычисления по последней формуле проводятся без привнесения ошибки округления.

3.4. Вычисление квадратного корня. Поскольку на современных компьютерных платформах затраты на вычисление квадратного корня малы (они примерно равны затратам на вычисление операции деления), авторами было принято решение использовать для вычисления верхних и нижних границ этой функции специальную технику. Для вычисления нижней границы мы используем значение t, возвращаемое стандартной математической библиотекой C, и проверяем, является ли оно нижней границей с помощью вычисления его квадрата $t \times t$ с округлением вверх. Если результат не превосходит аргумента, то t является нижней границей; иначе мы выполняем сдвиг: берем вместо t соседнее снизу машиннопредставимое число и повторяем описанную выше операцию. Вычисление верхней границы проводится аналогичным способом с помощью сдвига вверх. Как показали эксперименты, для вычисления границ ни разу не пришлось выполнять более одного сдвига.

Заключение

Математическая библиотека реализована как модуль, независимый от платформы (с единственным ограничением соответствия представления вещественных чисел стандарту IEEE-754), с интерфейсом на языке C, состоящий из набора функций на языке C++. Он содержит следующие функции: синус, косинус, тангенс, котангенс, арксинус, арккосинус, арктангенс, квадратный корень,

натуральный логарифм, экспонента, функция возведения в вещественную степень. Данная библиотека встроена в интервальный решатель нелинейных систем уравнений SibCalc. Для обеспечения точности результата в библиотеке используются оригинальные разработки авторов. Гарантированный интервальный результат вычисления функции для вырожденного интервального значения аргумента в 80–90% случаях состоит из двух идущих подряд машинно-представимых чисел, то есть является неулучшаемой интервальной оценкой. При этом обеспечивается высокая производительность вычисления элементарных функций, не уступающая производительности стандартной математической библиотеки C.

Список литературы

[1] Алефельд Г., Херцбергер Ю. Введение в интервальные вычисления. – М.: Мир, 1987. – 260 с.

[2] Березин И.С. Жидков Н.П. Методы вычислений. – М.: Физматгиз, 1959. – Т. 1. – 464 с.

[3] Люк Ю. Специальные математические функции и их аппроксимации. – М., 1980. – 608 с.

[4] Ремез У.Я. Основы численных методов чебышевского приближения. – Киев: Наукова думка, 1969.

[5] Hansen E. Global Optimization Using Interval Analysis. – New York: Marcel, Dekker, 1992.

[6] Клатте Р., Кулиш У., Неага М., Рац Д., Ульрих У. PASCAL-XSC. Язык численного программирования. – М.: ДМК, 2000.

[7] Kleymenov A., Petunin D., Semenov A., Vazhev I. A Model of Cooperative Solvers for Computational Problems // Proc. of the 4th Intern. Conf. PPAM 2001, Poland, September: Lect. Notes Comp. Sci. – Vol. 2328. – P. 797–802.