УДК 004.22:004.43

О.В. Мнушка

Харьковский национальный автомобильно-дорожный университет, Харьков

АНАЛИЗ ВАРИАНТОВ РЕАЛИЗАЦИИ СПЕЦИАЛЬНЫХ МАТЕМАТИЧЕСКИХ ФУНКЦИЙ В БИБЛИОТЕКАХ ЯЗЫКА ПРОГРАММИРОВАНИЯ C/C++

Проведен анализ вариантов реализации специальных математических функций в популярных прикладных и стандартных библиотеках языков C и C++. Проведено тестирование реализаций функции ошибки и дополнительной функции ошибки в стандартных и распространенных свободных прикладных библиотеках языков C и C++. Определены интервалы изменения аргумента, в которых значение абсолютной погрешности вычисления рассматриваемых функций существенно и является источником дополнительных погрешностей вычислений. Показано, что в случае применения компиляторов семейства gcc, для уменьшения дополнительной погрешности вычислений без снижения их производительности предпочтительно использовать библиотеки quadmath и boost.

Ключевые слова: *специальные функции, стандартные и прикладные библиотеки, погрешность вычислений, функция ошибок, производительность вычислений.*

Введение

Проблема и ее связь с научными и практическими задачами. Имитационное моделирование (ИМ), как метод научных исследований, получило распространение в качестве альтернативы реальным физическим экспериментам и в качестве метода исследования сложных процессов различной природы. Для разработки программ ИМ получили распространение такие языки программирования (ЯП), как Fortran, C, C++, Python и др. Использование C и C++ для задач данного класса обусловлено: 1) наличием большого числа прикладных библиотек; 2) высоким быстродействием полученного кода; 3) возможностями построения интерфейса любого типа и т.д. В процессе разработки и использования программ ИМ возникают устранимые – ошибки выбора метода, алгоритма, ошибки в программном коде, неправильный ввод данных, и неустранимые ошибки – погрешности исходной модели; ошибки и погрешности вычислений, возникающие из-за ограниченной разрядной сетки компьютера. Для устранения ошибок и уменьшения погрешностей вычислений используют специальные приемы вычислений – интервальные вычисления и вычисления с произвольной точностью, или изменяют способы отображения данных в память компьютера. В прикладных библиотеках C и C++ используются оба подхода, при этом возникает задача оценить эффективность предложенных мер как минимум по двум основным параметрам – погрешность и время вычислений.

Анализ исследований и публикаций. Проблема точности вычисления специальных математических функций лежит в двух плоскостях – представление данных в памяти компьютера (с фиксированной (ФТ) и плавающей точкой (ПТ)) и реализа-

ции алгоритмов для корректных вычислений. Формат с ФТ применяется в игровых приставках, мобильных телефонах, встраиваемых устройствах, т.е. в устройствах без математического сопроцессора (FPU), а также в СУБД и некоторых ЯП (PL/I, COBOL, Ада95 и Си (в реализации gcc [1]) для реализации финансовых операций.

В области научных вычислений, как правило, используют формат с ПТ в виде $\pm M \cdot q^{\pm p}$, где M – мантисса, q – основание системы счисления, p – порядок, при этом точность (число десятичных разрядов) оценивают как $M \cdot \log 10(q)$. В IEEE 754-2008 определены базовые форматы для двоичных и десятичных чисел с ПТ и способы их расширения [2]. Десятичные числа в формате с ПТ получили распространения в микропроцессорной и микроконтроллерной технике [3]. К недостаткам формата с ПТ относят ограниченную неравномерную размерную сетку, неассоциативность операций и т.д. [4]. В [5] предлагают «постбинарный» формат числе с ПТ, который должны обеспечить более гибкую обработку данных. Оценить предложенный подход не представляется возможным, т.к. отсутствует соответствующее аппаратное и программное обеспечение.

В основе реализации математических функций (элементарных и специальных) в прикладных библиотеках ЯП лежат работы [6, 7], современные подходы к решению подобных задач изложены в [8] и др. Учитывая большое число актуальных ЯП, ISO/IEC выпустила стандарт ISO/IEC 10967 [9], предписывающий организацию вычислений и реализации математических функций, независимых от реализации конкретного ЯП. В [10] предложены подходы к разработке и тестированию математических функций в ЯП, показано, что существует проблема отсутствия систематической методики для определения требований к математическим функциям, также отсутствуют систематические тесты, для проверки адекватности и точности реализации математических функций в ЯП.

Постановка задачи. В имитационном моделировании сложных технических систем находят применение ряд специальных функций, среди которых функции Бесселя, интегралы ошибок, гамма-функция $\Gamma(x)$ и др. В стандартной библиотеке ЯП C, соответствующего стандартам C99 и C11, в заголовочном файле math.h включены реализации функций ошибки (1) [(7.1.1), 6] и дополнительная функция ошибки (2) [(7.1.2), 6], гамма-функции $\Gamma(z)$ и ее натурального логарифма $\ln|\Gamma(z)|$. В библиотеке *quadmath* дополнительно определены функции Бесселя первого и второго рода. Широкий выбор реализаций специальных функций содержится в библиотеках *boost*::*math* и *gsl*. В черновике стандарта C++ TR1 (ISO/IEC DTR 19768:2005) набор СФ расширен и частично реализован в *boost* и *gcc*, но не входит в последний стандарт (C++11).

Рассмотрим проблемы, связанные с реализацией специальных функций в библиотеках ЯП на примере реализации функции ошибки $\mathrm{erf}(x)$ (1) и дополнительной функцию ошибки $\mathrm{erfc}(x)$ (2). В связи с тем, что рассматриваемые функции не могут быть выражены в элементарных функциях, используют разложение их в ряд, например (3) [(7.1.5), 6] и (4), (7.2.14), 6].

$$\mathrm{erf}(x)\frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}dt, \qquad (1)$$

$$\mathrm{erfc}(x) = 1 - \mathrm{erf}(x) = \frac{2}{\sqrt{\pi}}\int_x^\infty e^{-t^2}dt, \qquad (2)$$

$$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}}\sum_{n=0}^{\infty}\frac{(-1)^n x^{2n-1}}{n!(2n+1)}, \qquad (3)$$

$$\mathrm{erfc}(x) = \frac{e^{-x^2}}{x\sqrt{\pi}}\sum_{n=0}^{\infty}\frac{(2n)!}{n!(2x)^{2n}}. \qquad (4)$$

Для повышения точности вычислений функций (1) и (2) W.J. Cody была предложена рациональная аппроксимация при помощи многочленов Чебышева [11] и ее реализация на языке Fortran в библиотеке Netlib/Specfun с точностью до 18 десятичных знаков после запятой [12], которая с изменениями используется в библиотеках различных ЯП и математических пакетах, например, в Matlab.

Вычисления специальных функций (как и любые вычисления с данными в формате с ПТ) могут быть источником дополнительных погрешностей результатов моделирования, при этом, как правило, ни закон изменения этой погрешности, ни ее величина не указывается, поэтому требуется анализ доступных исходных кодов библиотек и тестирования реализаций указанных функций.

Цель статьи – исследование вариантов реализаций специальных функций в прикладных библиотеках языков программирования C и C++ и оценка дополнительной погрешности вычислений, обусловленной конкретной реализацией специальных функций в библиотеках ЯП.

Реализация специальных функций в библиотеках C и C++

В работе на основе исходного кода библиотек ЯП C и C++ проанализированы реализации специальных функций в их стандартных библиотеках (*glibc* 2.18, *libstdc*++ 4.8.2), и прикладных библиотеках – *quadmath* (C, 128-битные числа), *boost::math* (C++, *boost* 1.54) и *GNU scientific library* (C++, gsl 1.15)

Реализация в фортране [11] положила начало использованию рациональной аппроксимации многочленами Чебышева, в разных модификациях, связанных с величиной интервалов изменения аргумента.

1. $|x| \le 0,5 : \operatorname{erf}(x) \approx x \cdot R_{ij}\left(x^2\right)$.

2. $0,46875 \le |x| \le 4,0 : \operatorname{erfc}(x) \approx e^{-x^2} x \cdot R_{ij}(x)$.

3. $x > 4,0 : \operatorname{erfc}(x) \approx \frac{e^{-x^2}}{x\sqrt{\pi}} + \frac{e^{-x^2}}{x^3} R_{ij}\left(\frac{1}{x^2}\right)$,

где $R_{ij}(x) = P_i(x)/Q_j(x)$ – отношение многочленов Чебышева степени i в числители и степени j в знаменателе.

Реализация в glibc (1993 г., Sun Microsystems, 1997 г., N. Shimizu): $\operatorname{erf}(x) = 1 - \operatorname{erfc}(x), |x| > 1.0$, $\operatorname{erfc}(x) = 1 - \operatorname{erf}(x), |x| < 1/4$. Особые точки: $\operatorname{erf}(0) = 0$, $\operatorname{erf}(\infty) = 1$, $\operatorname{erf}(-\infty) = -1$, $\operatorname{erfc}(0) = 1$, $\operatorname{erfc}(\infty) = 0$, $\operatorname{erf}(-\infty) = 0$. Используется разбиение на следующие интервалы изменения аргумента:

1. $|x| \le 7/8$, $\operatorname{erf}(x)$ из (2).

2. $|x| \in [7/8, \ 1]$, $\operatorname{erf}(s + c) = \operatorname{sign}(x) \cdot (c + \operatorname{Rij}(s))$, $s = |x| - 1$, $c = 0,84506291151$.

3. $|x| \in [1/4, \ 5/4]$, $\operatorname{erfc}(s + c) = \operatorname{erfc}(c) + s \cdot R_{ij}(s)$, $c = 1/4, 3/8, \ldots, 9/8$, $s \in [0 \ \ 1/8]$.

4. $|x| \in [5/4, \ 107] : \operatorname{erfc}(x) = x^{-1} e^{\left(-x^2 - 0.5625 + R_{ij}(x^{-2})\right)}$.

5. $|x| \in [107, \ \infty) : \operatorname{erf}(x) = \operatorname{sign}(x) \cdot (1 - \varepsilon)$, $\operatorname{erfc}(x) = \begin{cases} \varepsilon^2, & x > 0 \\ 2 - \varepsilon, & x < 0 \end{cases}$, где $\varepsilon = 10^{-300}$ для данных типа double, $\varepsilon = 10^{-4931}$ для данных long double.

Реализация в quadmath (1993 г., Sun Microsystems, 2001 г. S. L. Moshier) аналогична glibc, но с большим числом интервалов разбиения функций (1):

1. $|x| \le 7/8$, $\operatorname{erf}(x)$ из (2).

2. Для $|x| \in [0;1,125] : \operatorname{erfc}(x + c) = \operatorname{erfc}(c) + xR_{ij}(x)$, где $c = 0,125n$, $n = 2,3,...,9$.

3. $|1/x| < 1 : \operatorname{erfc}(1/x) = \frac{1}{x} e^{\left(-x^{-2} - 0.5625 + R_{ij}(x^{-2})\right)}$,

где функция $\operatorname{erfc}(x)$ разбивается на интервалы размером $1/x + 1/8$, а $x \in [1; 128]$.

Реализация gsl (2003 г. J.Theiler, 2006 г. G. Jumgman) основана на (7.1.5) и (7.2.14) [6]:

1. $|t| < 1$ $\operatorname{erfc}((t+1)/2)$.

2. $|x| \le 1 : t = 2x - 1$.

3. $|x| \in [1; 5] : \operatorname{erfc}(x) = e^{x^2}$, $x = 2t + 3$, $|t| < 1$.

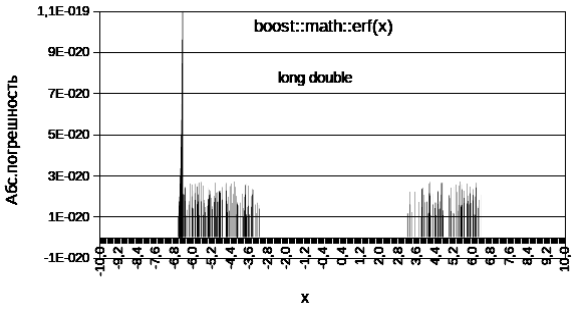
4. $|x| \in [5; 10] : \operatorname{erfc}(x) = xe^{x^2}$, $x = (5t + 15)/2$, $|t| < 1$.

5. $|x| > 8$ используется разложение в ряд №5725 [13].

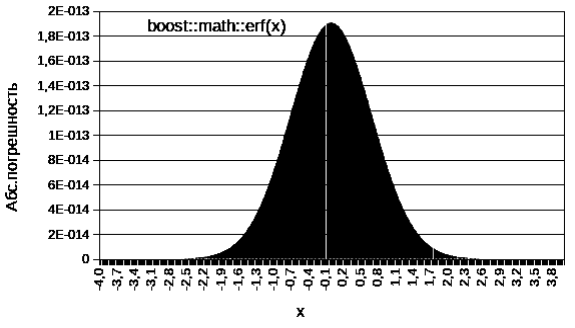
Реализация в boost::math (2006 г., J. Maddock) основана на разложении в ряд (2), позволяет производить вычисления с произвольной и фиксированной – double, long double, quadruple – точности. Для каждого типа данных существует отдельная шаблонная реализация и собственная схема разбиения функций (1) на интервалы, на которых происходит аппроксимация полиномами Чебышева высокой степени.

Тестирование реализаций функций. Для оценки погрешности вычислений функции ошибок были составлены программы на Си и C++. В соответствии с методикой [10] в качестве эталонной функции для тестирования были выбраны реализации функций (1) из библиотеки вычислений с произвольной точностью GNU *mpfr* (www.mpfr.org), которая используется в *Maple*, *Matlab*, *Sage* и др. Параметры компьютера – процессор Intel Core2 Duo T5750 (2 ГГц, 2 Мб кэш, FSB 667 МГц), ОЗУ 4 Гб, ОС Linux (amd64, kernel 3.13).

На рис. 1 приведены результаты расчета абсолютной погрешности вычислений функции ошибки (1) для аргументов *long double* (рис. 1, а) и *double* (рис. 1, б).



а



б

Рис. 1. Абсолютная погрешность вычисления функции ошибки: а – функция *boost::math erf(x)* для аргумента *long double*; б – для аргумента типа *double*

Абсолютная погрешность вычислений (1) составляет:

1) для данных типа *double* не более $2 \cdot 10^{-13}$ для $x \in [-3; 3]$;

2) для данных типа *__float128 (binary128)* не более $1,7\cdot10^{-13}$ для $x\in[-3;3]$;

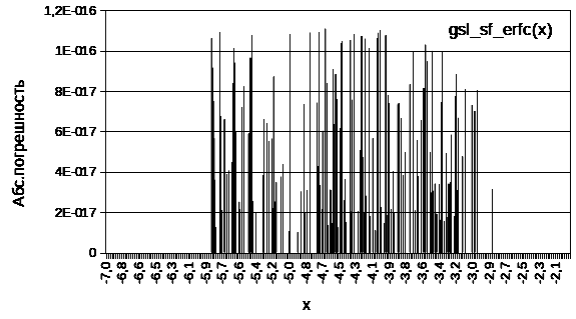
3) для данных типа *long double* – не более 10^{-19} и локализована в двух областях $x\in[-7;-3]$ и $x\in[3;7]$.

Абсолютная погрешность вычисления функции $\mathrm{erfc}(x)$:

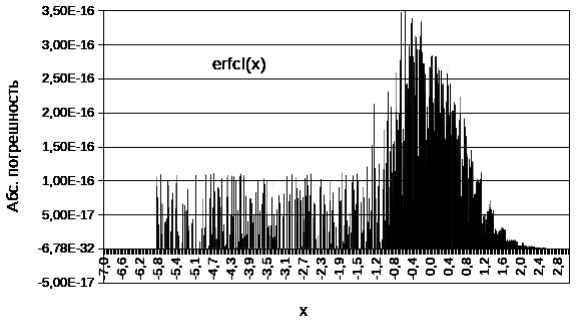
1) для данных типа double (рис. 2, а) не превышает $1,2\cdot10^{-16}$ для $x\in[-6;-2,5]$;

2) для данных типа *__float128* не более 10^{-34} для $x\in[-9;-2,5]$;

3) для данных типа *long double* (аналог. рис. 2, а) – $(1\ldots3)\cdot10^{-16}$ для $x\in[-6;2,5]$ (C99 *glibc* erfcl(x), рис. 2, б) и не более $2,5\cdot10^{-16}$ для $x\in[-3;3]$.



а



б

Рис. 2. Абсолютная погрешность вычисления функции, тип данных *long double*: а – библиотека *gsl*; б – библиотека *glibc (C99)*

Анализ результатов показывает, что погрешность вычисления функций (1) зависит от библиотеки и типа данных. Погрешность вычисления функции erf(x) приблизительно на три порядка меньше, чем погрешность вычисления функции erf(x). Следует отметить, что:

1) результаты, полученные для данных типа *long double* в целом не согласуются для различных библиотек;

2) не выполняется строгое равенство $\mathrm{erfc}(x)=1-\mathrm{erf}(x)$, погрешность $\sim\left(10^{-16}\ldots10^{-15}\right)$ для $x<0$.

Данные по среднему времени вычисления тестируемых функций (табл. 1) получены при вычислении значений исследуемых функций 10^6 раз для 30 испытаний каждой функции. Результаты измерения времени вычислений справедливы только для приведенной выше конфигурации тестовой системы и могут отличаться для других конфигураций. Для вычислений выбраны данные максимально возможной стандартной точности (по IEEE 754-2008) для данной библиотеки. Для библиотеки *boost* не использовались библиотеки вычислений с произвольной точностью, т.к. они основаны на *mpfr*(*gmp*) и *quadmath*.

Таблица 1

Время вычислений функции ошибки

Библиотека	Мантисса, разр.	Среднее время, с.
qaudmath	113	2,55
glibc	64	0,09
boost	64	0,14
gsl	53	0,22
mpfr	23	14,67
mpfr	53	36,21
mpfr	64	45,16
mpfr	113	93,86

Наивысшее быстродействие показала реализация функции ошибки в стандартной библиотеке C/C++. Быстродействие реализации в *quadmath* в 10-20 раз медленнее, а реализация в *mpfr* – на 2-3 и более порядка медленней в зависимости от точности, (рис. 3).

С точки зрения времени вычислений, предпочтительно использовать реализации *boost* и стандартной библиотеки, для повышения точности при приемлемом времени вычислений – *qaudmath* (или аналог из *boost*). Реализация *mpfr* не обеспечивает достаточного быстродействия, поэтому может применяться только для тестирования или специальных исследований, связанных с точностью вычислений.

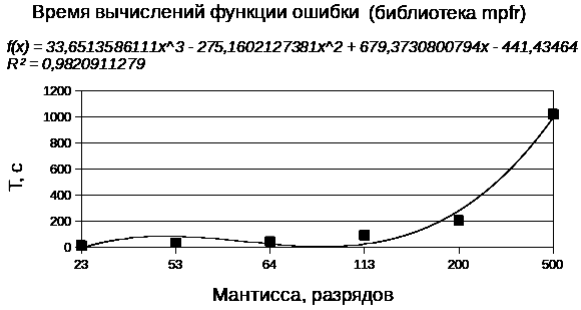


Рис. 3. Время вычислений функции *erfc* (реализация *mpfr*)

Выводы и перспективы дальнейших исследований

В работе проанализированы способы реализации специальных математических функций в прикладных библиотеках C/C++. Большинство актуаль-

ных библиотек ЯП C/C++ разработаны десять-двадцать лет назад и по большинству из них нет сведений об использованных методиках тестирования и погрешностях вычислений, производимых с их помощью.

Рассмотренные реализации функций используют дробно-рациональную аппроксимацию функций (1) полиномами Чебышева высоких порядков. Использование расширенного формата чисел с ПТ (80 бит) с целью уменьшения погрешности вычислений нежелательно ввиду его отсутствия в [2], привязки к определенной архитектуре и отсутствия поддержки в ряде компиляторов, что делает программу непортабельной. Предпочтительным является использование библиотек *boost* или *quadmath*, которые обеспечивают высокоточные вычисления без снижения (или с приемлемым снижением) производительности вычислений.

Полученные результаты могут быть использованы для выбора прикладных библиотек языков C/C++ при разработке программного обеспечения имитационного моделирования с уменьшенной дополнительной погрешностью вычислений, а также при анализе существующих прикладных библиотек языков программирования с целью приведения их к требованиям стандарта ISO/IEC 10967, снижения погрешностей вычислений и устранения возможных ошибок.

Список литературы

1. ISO/IEC TR 18037:2008. Programming languages. C. Extensions to support embedded processors [Text]. – ISO, 2008. – 97 p.

2. IEEE Standard for Floating-Point Arithmetic [Text]. – New York, 2008. – 70 p.

3. Vincent R. Decimal Floating Point Format Based on Commonly Used Precision For Embedded System Applications [Text] / R. Vincent, S.L. Anju // Proc. of Int. Conf. on Microel., Comm. and Renew. Energy. – 2013. – P. 1-4.

4. Goldberg D. What Every Computer Scientist Should Know About Floating-Point Arithmetic [Text] / D. Goldberg // ACM Comp. Surv. – 1991. – Vol. 23. – No 1. – P. 5-48.

5. Аноприенко А.Я. Особенности представления вещественных чисел в постбинарных форматах [Текст] / А.Я. Аноприенко, С.В. Иваница // Математичні машини і системи. – 2012. – №3. – С. 49-60.

6. Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables [Text] / Ed. by M. Abramovitz and I.A. Stegun. – 1964. – 1047 p.

7. Cody W.J. Software Manual For the Elementary Functions [Text] / W.J. Cody, W. Waite. – New Jersey, 1980. – 279 p.

8. Gil A. Numerical methods for special functions [Text] / A. Gil, J. Segura, N.M. Temme. – SIAM, 2007. – 417 p.

9. ISO/IEC 10967-1. Information technology. Language independent arithmetic. Part 1: Integer and floating point arithmetic [Text]. – Geneva, 2012. – 144 p.

10. Кулямин В.В. Формальные подходы к тестированию математических функций [Текст] / В.В. Кулямин // Труды ИСП РАН. – 2006. – Т. 10. – С. 69-114.

11. Cody W.J. Rational Chebyshev approximations for the error function [Text] / W.J. Cody // Math. Comp. – 1969. – No 23. – P. 631-637.

12. SUBROUTINE CALERF(ARG, RESULT, JINT) [Электронный ресурс]. – Режим доступа к ресурсу: www.netlib.org/specfun/erf. – 04.2014.

13. Computer Approximations [Text] / [J.F. Hart, E.W. Cheney, C.L. Lawson and others]. – 1968. – 354 p.

Поступила в редколлегию 19.04.2014

Рецензент: д-р техн. наук, проф. О.Я. Никонов, Харьковский национальный автомобильно-дорожный университет, Харьков.

АНАЛІЗ ВАРІАНТІВ РЕАЛІЗАЦІЇ СПЕЦІАЛЬНИХ МАТЕМАТИЧНИХ ФУНКЦІЙ В БІБЛІОТЕКАХ МОВИ ПРОГРАМУВАННЯ C/C++

О.В. Мнушка

Проведено аналіз варіантів реалізації спеціальних математичних функцій у популярних прикладних і стандартних бібліотеках мов C/C++. Проведено тестування реалізацій функції похибки та додаткової функції похибки в стандартних та поширених вільних прикладних бібліотеках мов C та C++. Визначені інтервали змінення аргументу, в яких значення абсолютної похибки обчислення функцій, що розглядаються, є суттєвими та служать джерелом додаткових похибок обчислень. Показано, що у разі використання компіляторів C/C++ сімейства gcc, для зменшення додаткової похибки обчислень без зниження їх продуктивності бажано використовувати бібліотеки quadmath та boost.

Ключові слова: *спеціальні функції, стандартні та прикладні бібліотеки, похибка обчислень, функція помилок, продуктивність обчислень.*

ANALYSIS OF THE SPECIAL MATHEMATICAL FUNCTIONS IMPLEMENTATIONS IN THE LIBRARIES OF THE C/C++ PROGRAMMING LANGUAGES

O.V. Mnushka

The paper deals with the analysis of the implementation of special mathematical functions in popular applied and standard libraries of C and C++. Implementations of the error function and complementary error function in the C and C++ standard and common free applied libraries were tested. Intervals of the argument, in which the value of the absolute calculation errors of the considered functions are significant and cause the additional calculations errors, have been calculated. To conclude, for gcc compilers, to reduce additional computational errors without reducing (or with acceptable reducing) their performance is preferable to use quadmath and boost libraries.

Keywords: *special functions, standard and applied libraries, computational error, the error function, performance computing.*