СООБЩЕНИЯ
ОБЪЕДИНЕННОГО
ИНСТИТУТА
ЯДЕРНЫХ
ИССЛЕДОВАНИЙ
ДУБНА

P11 - 8612

А.И.Салтыков

НОВЫЕ ПРОГРАММЫ ВЫЧИСЛЕНИЯ ЭЛЕМЕНТАРНЫХ ФУНКЦИЙ НА БЭСМ-6

1975

P11 - 8612

А.И.Салтыков

НОВЫЕ ПРОГРАММЫ ВЫЧИСЛЕНИЯ ЭЛЕМЕНТАРНЫХ ФУНКЦИЙ НА БЭСМ-6

ВВЕДЕНИЕ

В состав программного обеспечения БЭСМ-6 входят программы вычисления 8 элементарных функций: $\sqrt{x}$, $\sin x$, $\cos x$, $\operatorname{arctg} x$, $\arcsin x$, $\ln x$, e^x , $[x]$. Эти программы оформлены в виде автокодной подпрограммы - экстракода, имеющей отдельные входы для каждой из вышеупомянутых функций.

Анализ существующей программной реализации выявил ее неоптимальноость как в смысле общей организации, так и в смысле качества отдельных алгоритмов.

Неоптимальность общей организации экстракода выразилась в неоправданно больших "накладных расходах", достигающих 30% счетного времени. Эти дополнительные расходы вызваны ожиданием выполнения команд в арифметическом устройстве БЭСМ-6 (из-за наличия команд записи из сумматора в индекс-регистр).

Из отдельных алгоритмов наиболее неоптимальными оказались алгоритмы вычисления $\sin x$, $\cos x$ и $\arcsin x$. Отметим, что в существующих программах не обеспечивается точное равенство $\ln$ I=0, а $\sqrt{n^2}=n$ не для всех натуральных $n \leq 2^{20}$. Это в значительной степени обусловлено принятым на БЭСМ-6 способом округления результата (к младшему разряду результата логически добавляется "I").

Взамен существующих предлагаются новые программы, которые до предела сокращают накладные расходы. Алгоритмы этих программ работают быстрее и дают более высокую точность. Общая длина экстракода при этом сократилась на 25% (3II машинных слов против 4I2 в восьмеричном исчислении).

Изменение скорости работы и точности новых программ по сравнению со старыми видно из таблицы. Верхняя строка соотвествует старому варианту программы, нижняя – новому. Точность указана для тех значений аргумента, при которых погрешность в аргументе не сказывается на точности вычисления функции.

Ниже дается описание алгоритма вычисления каждой функции с указанием отличий от старого варианта.

I. Вычисление $\sqrt{x}$

Пусть $x = 2^P y$, где P – порядок, $\frac{1}{2} \le y < 1$ – мантисса. Тогда $\sqrt{x} = 2^{P/2}\sqrt{y}$.

Вычисление $\sqrt{y}$ производится в два этапа. Сперва вычисляется начальное приближение

$$\sqrt{y} \approx a_2 y^2 + a_1 y + a_0,$$

обеспечивающее относительную погрешность не более 10^{-3} (см. /I/, таблица I30). Затем производится уточнение по формуле Герона

$$y_{n+1} = \frac{1}{2}\left(y_n + \frac{y}{y_n}\right).$$

Для получения I2 верных знаков достаточно двух итераций уточнения.

Новый алгоритм отличается от старого коэффициентами начального приближения и блокировкой округления на этапе уточнения. Последнее позволило получить точное значение корня для всех $x = n^2$, n = I,2, ..., 2^{20}, т.е. для всех квадратов целых чисел, точно представимых в машине. Старый алгоритм давал точное значение корня лишь до n = I92.

Таблица

Функция	$\sqrt{x}$	sinx	cosx	arctgx	arcsinx	lnx	e^x	[x]
Счетное время в мкс	90	I20	I20	I25	I70	80	90	40
	60	55	55	90	80	60	65	25
Макс. отн. погрешность	I,5·IO^{-I2}	7·IO^{-I2}	7·IO^{-I2}	I5·IO^{-I2}	I5·IO^{-I2}	4·IO^{-I2}	3·IO^{-I2}	0
	I,5·IO^{-I2}	4·IO^{-I2}	4·IO^{-I2}	7·IO^{-I2}	5·IO^{-I2}	3·IO^{-I2}	5·IO^{-I2}	0

2. Вычисление $\sin x$ и $\cos x$

Сначала аргумент приводится к отрезку $[0,\frac{\pi}{2}]$, после чего производится вычисление по формуле

$$\sin\frac{\pi}{2}z = zP(z^2), \qquad 0 \le z \le 1 .$$

Коэффициенты полинома взяты из /I/ (таблица 3343), степень полинома равна 6. Чтобы не получилось значений $|\sin x|>1$, вычисление полинома производится с блокировкой округления.

В старом варианте программы вместо хорошо зарекомендовавшего себя полиномиального приближения почему-то использовано дробно-рациональное приближение. Это привело к значительному снижению скорости работы программы и не дало преимущества в точности. Вероятно, автор программы не нашел иного способа предотвратить получение значений $|\sin x|>1$.

Кроме того, в процессе приведения аргумента к стандартному отрезку в старом варианте была допущена неосторожность. В результате при малых значениях аргумента терялись его младшие разряды, что приводило к резкому ухудшению точности. Для $x\approx 10^{-n}$ у результата терялось n десятичных знаков.

Вычисление $\cos x$ в обоих вариантах программы производится по формуле

$$\cos x = \sin\left(\frac{\pi}{2}-x\right).$$

При этом снижается точность для значений x, близких к $\frac{\pi}{2}$, за счет потери старших разрядов у аргумента. Это снижение точности неизбежно и не может быть устранено путем непосредственного вычисления $\cos x$.

3. Вычисление $\operatorname{arctg} x$

Аргумент приводится к отрезку $[0,1]$ с помощью преобразования

$$\operatorname{arctg} x = \frac{\pi}{2} - \operatorname{arctg}\frac{1}{x}, \qquad |x|>1.$$

Далее используется дробно-рациональное приближение

$$\operatorname{arctg} x \approx x\frac{P(x^2)}{Q(x^2)} .$$

Коэффициенты полиномов взяты из /I/ (таблица 5096), степени обоих полиномов равны 4.

В прежнем варианте программы аргумент приводился к отрезку малой длины, после чего использовалось полиномиальное приближение.

Неэффективность полиномиального приближения здесь обусловлена тем, что в процессе приведения аргумента к стандартному отрезку происходит ожидание выполнения команд в арифметическом устройстве БЭСМ-6 (из-за наличия команды условной передачи управления). Даже более эффективный алгоритм (см. /2/, стр. 150), использующий на отрезке $[0,\ 2-\sqrt{3}]$ полиномиальное приближение 6-й степени (/I/, таблица 4943), оказался хуже предлагаемого.

4. Вычисление $\arcsin x$

Аргумент приводится к отрезку $[0,1/2]$ с помощью преобразования

$$\arcsin x = \frac{\pi}{2} - 2\arcsin\sqrt{\frac{1-x}{2}}, \qquad \frac{1}{2}\le x\le 1.$$

Далее используется дробно-рациональное приближение

$$\arcsin x \approx x\frac{P(x^2)}{Q(x^2)} .$$

Степени полиномов равны 3 и 4 (см. /I/, таблица 4696).

В старом варианте программы вычисление производилось по формулам

$$\arcsin x = \operatorname{arctg}\frac{x}{\sqrt{1-x^2}}, \qquad 0\le x\le\frac{\sqrt{2}}{2},$$
$$\arcsin x = \frac{\pi}{2} - \operatorname{arctg}\frac{\sqrt{1-x^2}}{x}, \qquad \frac{\sqrt{2}}{2} < x \le 1,$$

что привело к значительному снижению скорости счета.

5. <u>Вычисление $\ln x$</u>

Пусть $x = 2^P$, P – порядок, $\frac{1}{2} \leq y < 1$ – мантисса. Тогда $\ln x = P\ln 2 + \ln y$. Для вычисления $\ln y$ используется быстросходящееся полиномиальное приближение

$$\ln y = -\frac{\ln 2}{2} + zP(z^2), \qquad z = \frac{y - \frac{1}{\sqrt{2}}}{y + \frac{1}{\sqrt{2}}}.$$

Степень полинома равна 5 (/I/, таблица 2664). Для получения точного равенства $\ln 1 = 0$ константа $\frac{\ln 2}{2}$ уменьшена на единицу младшего разряда и вычисление полинома производится с блокировкой округления.

В старой программе использован тот же алгоритм, но при других коэффициентах полинома. Точного равенства $\ln 1 = 0$ не обеспечивалось.

По сравнению с часто используемыми алгоритмами, требующими разбиения отрезка [I/2, I] на части (см. /3/, стр. 25 и /4/, стр.27), примененный здесь алгоритм дает выигрыш в скорости и требует меньшего числа команд и констант. То обстоятельство, что в окрестности $x = 1$ точность резко падает, не является недостатком алгоритма, так как вблизи $x = 1$ $\ln x \approx x - 1$ и достижение высокой точности все равно невозможно из-за увеличения относительной погрешности величины $x - 1$.

6. <u>Вычисление e^x</u>

С помощью преобразования $e^x = 2^{\frac{x}{\ln 2}} = 2^{\left[\frac{x}{\ln 2}\right]} \cdot \left(2^{\frac{1}{2}\left\{\frac{x}{\ln 2}\right\}}\right)^2$

задача сводится к вычислению 2^z, $0 \leq z < \frac{1}{2}$. Для вычисления 2^z используется полиномиальное приближение 7-й степени

$$2^z \approx P(z)$$

(см. /I/, таблица I024).

В старом варианте программы задача сводилась к вычислению e^z, где $-\frac{\ln 2}{2} \leq z \leq \frac{\ln 2}{2}$, с использованием полиномиального приближения 9-й степени.

Отметим, что здесь не требуется (как в случае $\sin x$) осторожности в процессе приведения аргумента к стандартному отрезку, так как при малых x $e^x \approx 1 + x$ и младшие разряды аргумента все равно будут потеряны.

Отметим также, что относительная погрешность вычисления e^x пропорциональна модулю аргумента. Для $|x| \approx 40$ она может достигать $5 \cdot 10^{-11}$ из-за погрешности аргумента.

7. <u>Вычисление $[x]$</u>

Алгоритм вычисления $[x]$ достаточно прост. Сначала устанавливается режим блокировки нормализации и округления, после чего производится сложение x с "целым" нулем, затем устанавливается режим с блокировкой только округления и производится нормализация результата. Отметим, что из-за малости времени счета здесь начинают сказываться накладные расходы.

<u>Заключение</u>

Как видно из сказанного, программы вычисления элементарных функций не всегда составляются достаточно оптимальными. Причиной этому является, на наш взгляд, отсутствие в отечественной литературе хороших таблиц полиномиальных и дробно-рациональных приближений, подобных таблицам /I/. Одной из причин является также отсутствие единых требований к алгоритмам вычисления этих функций.

Ввиду сказанного нам представляется необходимой разработка тестовых программ для проверки качества алгоритмов, реализующих вычисление элементарных функций. Эти программы должны быть составлены на одном из общепринятых языков программирования высокого уровня (например, на ФОРТРАНе) с тем, чтобы их можно было использовать на различных ЭВМ.

Автор благодарит Л.Г.Каминского, И.Н.Силина и А.В.Гусева за помощь и полезные консультации.

ЛИТЕРАТУРА

I. J.F.Hart a.o. Computer Approximations. New York, John Wiley and Sons, 1968.

2. Л.А.Люстерник, О.А.Червоненкис, А.Р.Янпольский. Математический анализ. Вычисление элементарных функций. М.,Физматгиз, 1963.

3. Библиотека стандартных программ. Ред. М.Р.Шура-Бура. М., ЦБТИ, 1961.

4. Интерпретирующая система и элементарные функции. М., ВЦ АН СССР, 1965.

Рукопись поступила в издательский отдел
17 февраля 1975 года.