УДК 519.688

# ПОДХОДЫ К СЛОЖНЫМ ВЫЧИСЛЕНИЯМ В РАЗЛИЧНЫХ МАТЕМАТИЧЕСКИХ ПАКЕТАХ

*Гранковский В.А., Иваница С.В., Аноприенко А.Я.*
*Донецкий национальный технический университет*

## Введение

В идеале компьютер, выполняющий вычисления с плавающей точкой, должен выдавать не только результат вычислений, но и выявлять максимально возможную ошибку расчетов. К сожалению, на сегодняшний день основные разработчики аппаратного и программного обеспечения не предоставляют такую возможность. Неспособность вычислительных машин фиксировать погрешность зачастую игнорируется как разработчиками аппаратного и программного обеспечения, так и пользователям, потому что в большинстве расчетов она пренебрежимо мала. Тем не менее, как показал С. Румп [1], в некоторых случаях она может существенно сказываться на результате. В своей статье Румп привел пример, при вычислении которого на компьютере, погрешность составляет более 10%, что уже неприемлемо. Этот пример, считающийся каноническим, представляет собой полином с входными целочисленными значениями:

$$F(x,y) = 333.75y^6 + x^2\left(11x^2y^2 - y^6 - 121y^4 - 2\right) + 5.5y^8 + \frac{x}{2y}.$$

Также приводится результат вычисления данного полинома при $x = 77617$ и $y = 33096$:

$$F = -0.827396059946821368141165095479816...$$

Современные системы компьютерной математики (СКМ) не лишены описанной проблемы. Все СКМ, на которых просчитывался пример Румпа (Wolfram Mathematica, Mathsoft MathCAD, SciLab), в стандартном режиме работы вычислили его неверно.

В данной статье приводится несколько путей корректного вычисления «сложных» для компьютера задач и определения погрешности вычислений на программном уровне.

## Фиксация вычислительных ошибок в СКМ и пути получения правильного результат на примере вычисления примера Румпа

На первом этапе, вычисление примера Румпа было произведено в выбранных для исследования СКМ без использования каких либо специальных вычислительных средств: результат был получен при выполнении условия вычисления (полином Румпа) с заданными входными значениями. На примере вычислений СКМ SciLab (рис. 1) можно убедиться в получении неверного результата.

Погрешности вычислений, проведенных таким образом во всех выбранных СКМ, представлены в таблице 1 в процентном эквиваленте.

Одним из подходов к расчету «сложных» выражений вроде примера Румпа является использование *символьных вычислений*. В данном контексте это означает, что в таком режиме СКМ не работает с числами с плавающей точкой. Вместо этого вещественные числа должны быть представлены в виде рациональных дробей.

В СКМ MathCAD данное представление нужно выполнять вручную, т. е. непосредственно с клавиатуры при вводе условия вычисления. В этом случае полином Румпа принимает следующий вид:

$$f := \frac{33375}{100} y^6 + x^2 \left(11x^2y^2 - y^6 - 121y^4 - 2\right) + \frac{55}{10} y^8 + \frac{x}{2y}.$$

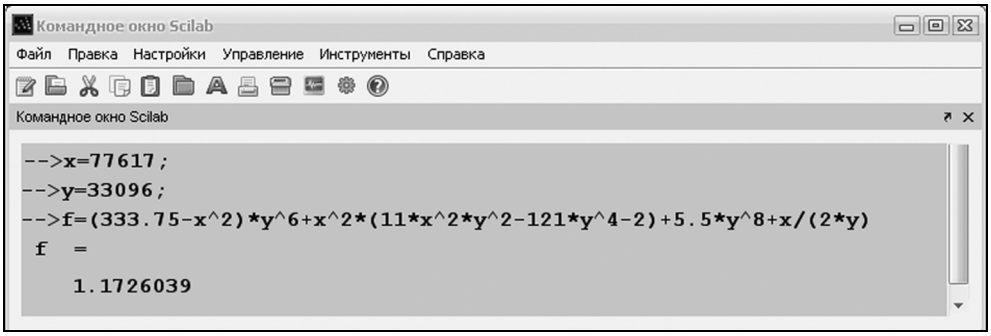


Рисунок 1 – Пример получения неверного результата в СКМ SciLab

Таблица 1 – Ошибка различных математических пакетов при вычислении примера Румпа

| СКМ | MathCAD | SciLab | Mathematica |
|---|---|---|---|
| % ошибки | $\sim 10^2$ | $\sim 10^2$ | $> 10^{23}$ |

Для получения правильного результата вычисления необходимо производить, используя символьный процессор MathCAD [2, с.139-143], обеспечивающий точный расчет. Полученный результат в виде обыкновенной дроби при ее вычислении дает правильный результат

$$f \to \frac{-54767}{66192} = -0.827396059946821,$$

в то время как непосредственное использование численного процессора вновь приводит к неверному результату:

$$f = 1.17260394005318.$$

В СКМ Mathematica можно осуществить ввод рациональных дробей не только ручным способом, но и воспользовавшись функцией Rationalize[], которая возвращает аргумент, преобразованный в рациональную дробь. В результате вычисления примера Румпа со всеми числами с плавающей точкой, преобразованными в обыкновенные дроби, МКС Mathematica также выдала правильный ответ в виде дроби:

$$F = -\frac{54767}{66192}.$$

Очевидно, что в данном представлении, погрешность результата равна нулю. Следует отметить, что в пакетах MathCAD и Mathematica все вычисления проводятся точно так же, как обычно, с той лишь разницей, что в вычисляемое выражение входят дроби вместо чисел с плавающей точкой.

Однако символьные вычисления не всегда способны решить вычислительную проблему. Например, если бы в выражение Румпа входила функция, возвращающая

иррациональный результат (синус, логарифм, радикал и др.), то ее преобразование к дробному виду гарантированно привело бы к погрешности, и такая нефиксированная потеря точности стала бы причиной получения неточного результата.

Ряд нестандартных вычислительных подходов на примере полинома Румпа могут предложить функции СКМ Mathematica, которая была выбрана в качестве эталонной ввиду как ее популярности, так и изначального наибольшего процента ошибки при стандартном вычислении примера Румпа. Так, в статье [3] был предложен подход к выполнению «сложных» вычислений в пакете Mathematica, суть которого сводится к тому, что в течение всего времени вычисления выражения принудительно используется фиксированную точность. Очевидно, что при повышении точности ошибка расчетов уменьшается, поэтому такое повышение имеет смысл, учитывая еще и то, что в пакете Mathematica можно установить практически сколь угодно большую точность вычисляемых значений.

Для того чтобы оставить точность в течение вычисления целого выражения неизменной, необходима фиксация внутренних параметров пакета $MaxPrecision и $MinPrecision с помощью функции Block[] на время вычисления выражения. Для удобства можно написать собственную функцию, осуществляющую вычисление таким образом:

```
SetAttributes[FixedPrecisionEvaluate,HoldFirst];
FixedPrecisionEvaluate[input_,digits_]:=
Block[{$MaxPrecision=digits,$MinPrecision=digits},input];
```

Аргумент input — это вычисляемое выражение; digits — точность, которую необходимо зафиксировать (количество десятичных значащих цифр, которые может иметь число). Этого еще недостаточно для правильного вычисления. Необходимо, чтобы все числа, входящие в выражение, имели ту же фиксированную точность digits. Для этого запишем функцию Румпа с третьим параметром — точностью, — который будет использоваться для всех чисел с плавающей точкой:

f3[x_,y_,digits_]:= SetPrecision[333.75,digits] $y^6+x^2*\left(11*x^2y^2-y^6-121*y^4-2\right)+$ SetPrecision[5.5, digits] $* y^8$ + SetPrecision[0.5, digits]$\frac{x}{y}$

Кроме того, во время вычисления необходимо устанавливать точность числам $x$ и $y$. Вычисление этой функции выполняется для разных значений точности (в диапазоне от 1 до 25):

```
Table[FixedPrecisionEvaluate[f3[SetPrecision[77617,i],SetPrecision[33096,i],i],i],{i,25}]
```

Полученный результат представлен в виде пакета данных:

```
{-1.,-1.0,-1.00,-1.000,-1.0000,-1.00000,-1.000000,-1.0000000,-1.00000000,-1.000000000,-
1.0000000000,-1.00000000000,-1.000000000000,-1.0000000000000,-1.00000000000000,-
1.000000000000000,-1.0000000000000000,-0.827396059946821368,-0.8273960599468213682,-
0.82739605994682136818,-0.827396059946821368176,-0.8273960599468213681761,-
0.82739605994682136817613,-0.827396059946821368176126,-0.8273960599468213681761258}
```

Из этих результатов видно, что, начиная с определенного значения точности (в данном случае начиная с 18 десятичных цифр), пакет выдает правильный результат, на каждом дальнейшем шаге увеличивая его точность.

К сожалению, у данного метода тот же недостаток, что и у метода символьных вычислений: заранее не известна погрешность результата. В общем случае совершенно не ясно, до каких пор стоит увеличивать точность, чтобы получить правильный результат с необходимой точностью.

Зависимость времени вычисления от задаваемой точности в пакете Wolfram Mathematica 7 в виде точек, а также ее линейная аппроксимация приведены на рис. 2. Из рисунка видно, что исследуемая зависимость линейна.

Еще одним способом вычислений «сложных» выражений является использование *интервальной арифметики* [4]. В этом случае для расчетов вместо чисел используются так называемые интервалы — диапазоны числовых значений, определенных левой (нижней) и правой (верхней) границами.

В СКМ Mathematica 7 реализована поддержка интервальной арифметики. Для того, чтобы ей пользоваться, достаточно при расчете использовать не числа, а интервалы. При этом ответ также выдается в виде интервала, в котором должно находиться искомое правильное число. Например, выражение Sin[Interval[{0, 0.1}]] равно Interval[{0, 0.0998334}] при вычислении в данном математическом пакете.

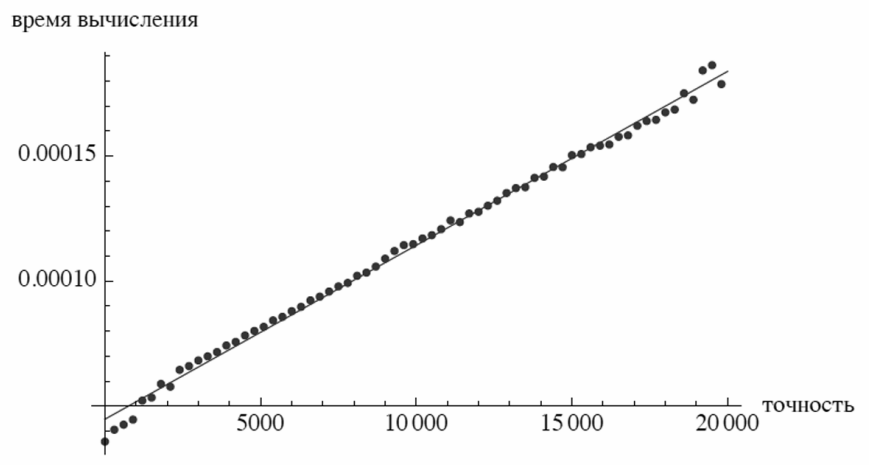


Рисунок 2 — Зависимость времени вычисления функции Румпа от задаваемой фиксированной точности.

Вычисление примера Румпа с использованием интервальной арифметики осуществляется при помощи интервалов, имеющих ширину, обеспечивающую необходимую точность (такие числа представляются в виде набора Interval[SetPrecision[a, digits]]). Чтобы вычислить значение выражения Румпа с использованием интервальной арифметики с точностью 20 цифр можно воспользоваться командой:

$$f[\mathrm{Interval}[\mathrm{SetPrecision}[77617, 20]], \mathrm{Interval}[\mathrm{SetPrecision}[33096, 20]]]$$

Где $f$ — та же функция Румпа, которая ранее была использована при обычных расчетах. С помощью интервальной арифметики можно получить интервал, в котором лежит искомое число, с произвольно задаваемой точностью и определить максимальную возможную погрешность вычислений (определяется из ширины интервала). Так, при заданной точности в 60 цифр, был получен достаточно узкий интервал, в котором находится правильное значение примера Румпа:

```
f[Interval[SetPrecision[a, 60]], Interval[SetPrecision[b, 60]]]

Interval[{-0.827396059946821368142, -0.827396059946821368140}]
```

Зависимость ширины интервала от задаваемой точности вычислений представлена на рис. 3, где по оси ординат отложена ширина интервала, а по оси абсцисс — задаваемая точность в количестве десятичных цифр. Ось ординат имеет логарифмический масштаб.

Также была оценена зависимость времени вычисления результата с использованием интервальной арифметики от точности вычислений в математическом

пакете Wolfram Mathematica 7. Эта зависимость приведена на рис. 4 в виде точек. На том же рисунке приведена аппроксимация результатов функцией $y = ax^{1.5}$.

**Заключение**

Приведенный и исследованный в данной работе пример Румпа наглядно демонстрирует, что вычисления над числами в формате с плавающей точкой неточны по самой своей природе, и современные СКМ, не фиксируя правильности получения промежуточных вычислений, в конечном счете, выдают результат, полностью состоящий из «шума».

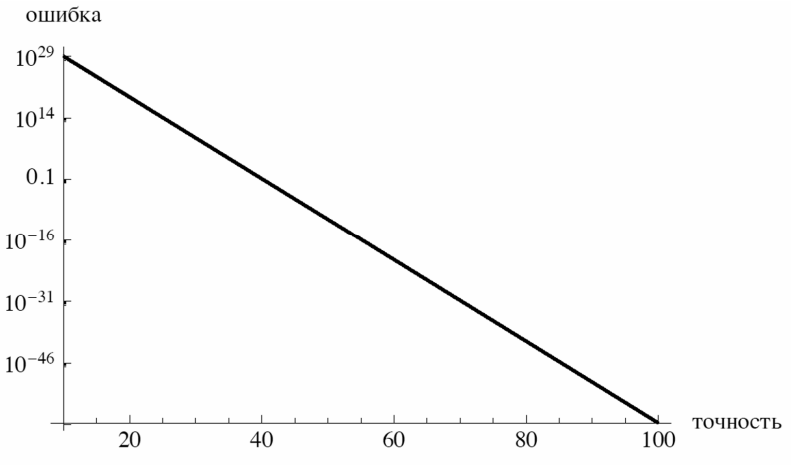


Рисунок 3 – Зависимость ошибки при вычислении примера Румпа от точности с использованием интервальной арифметики

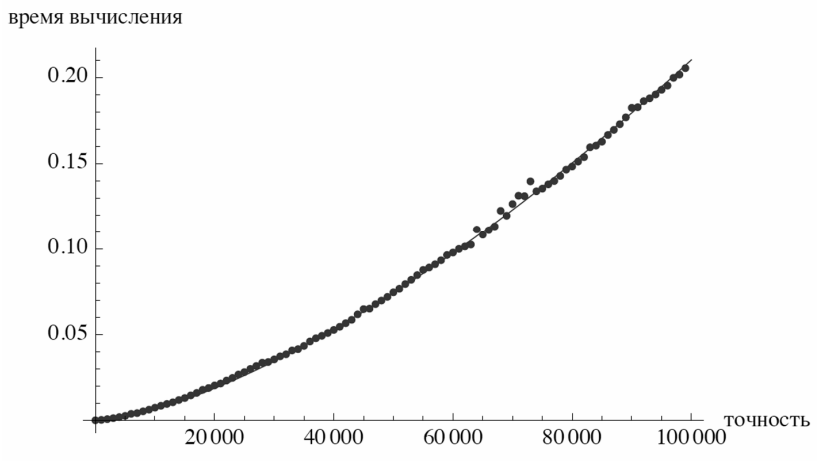


Рисунок 4 – Зависимость времени вычисления функции Румпа от точности с использованием интервальной арифметики.

Полученные результаты говорят о том, что правильно подобранный вычислительный метод приводит к получению точного результата и сведению погрешности к минимуму. Однако анализ решения поставленной задачи и выбор нужного метода остается за человеком, и для получения точного (заранее неизвестного)

результата часто приходится выполнять ряд дополнительных (уточняющих) вычислений. На этом фоне можно выделить интервальную арифметику как наиболее оптимальный метод вычислений, поскольку вычисленное при этом значение гарантированно заключено между границами результирующего интервала.

**Литература**

[1] S.M. Rump, Algorithm for verified inclusions – theory and practice, in Reliability in Computing, R. E. Moore ed., Academic press, San Diego, 1988.

[2] Кирьянов Д.В. Самоучитель MathCAD 2001 / Д.В. Кирьянов – СПб.: БХВ-Петербург, 2002. – 544с.

[3] M. Sofroniou, G. Spaletta — Precise Numerical Computation.

[4] Алефельд Г., Херцбергер Ю. Введение и интервальные вычисления. / Г. Алефельд и др., – М.:Мир, 1987. – 360 с.