Заметка о дуальных числах и автоматическом дифференцировании*

Степан Захаров†

11 мая 2023 г.

1. Дуальные числа

Дуальные числа $\mathbb{D}$ расширяют действительные $\mathbb{R}$ числом ε, для которого определяется следующие правила

$$\varepsilon \neq 0, \quad \varepsilon \cdot 0 = 0, \quad \text{но} \quad \varepsilon \cdot \varepsilon = \varepsilon^2 = 0.$$

Например, в дуальных числах уравнение $x^2 = 0$ имеет корни $0, \varepsilon^2, \varepsilon^3, \ldots$

Запись дуальных чисел аналогична записи комплексных

$$a + b\varepsilon, \quad a, b \in \mathbb{R},$$

где a назовём действительной частью числа, а b дуальной.

Сложение и **вычитание** чисел определяется интуитивно

$$\begin{aligned} (a + b\varepsilon) + (c + d\varepsilon) &= (a + c) + (b + d)\varepsilon, \\ (a + b\varepsilon) - (c + d\varepsilon) &= (a - c) + (b - d)\varepsilon. \end{aligned} \tag{1}$$

При **произведении** проявляется природа числа ε

$$(a + b\varepsilon)(c + d\varepsilon) = ac + ad\varepsilon + bc\varepsilon + \cancel{bd\varepsilon^2} = ac + (ad + bc)\varepsilon. \tag{2}$$

Физикам такое поведение может напомнить пренебрежение квадратичными поправками. Например, можно считать, что $b\varepsilon$ и $d\varepsilon$ являются погрешностями значений a и c. Тогда $bd\varepsilon^2$ мало по сравнению с остальными слагаемыми и пренебрегается.

Деление чисел можно представить домножением числителя и знаменателя на сопряжённое к знаменателю[1]

$$\frac{a + b\varepsilon}{c + d\varepsilon} = \frac{(a + b\varepsilon)(c - d\varepsilon)}{(c + d\varepsilon)(c - d\varepsilon)} = \frac{ac + (-ad + bc)\varepsilon}{c^2} = \frac{a}{c} + \frac{bc - ad}{c^2}\varepsilon. \tag{3}$$

†stepanzh@gmail.com, github.com/stepanzh

[1] При этом знаменатель ненулевой, $c + d\varepsilon \neq 0$.

2. Связь с дифференциированием

Для дуальных чисел справедливо утверждение, демонстрирующее связь с дифференциированием.

Утверждение 1.

$$f(a + b\varepsilon) = f(a) + bf'(a)\varepsilon. \tag{4}$$

Доказательство. Воспользуемся представлением функции в виде ряда Тейлора

$$f(x) = \sum_{k=0}^{\infty} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k.$$

Теперь вычислим функцию от дуального числа $a + b\varepsilon$, а в качестве x_0 возьмём a

$$\begin{aligned}
f(a + b\varepsilon) &= \sum_{k=0}^{\infty} \frac{f^{(k)}(a)}{k!}(a + b\varepsilon - a)^k = \sum_{k=0}^{\infty} \frac{f^{(k)}(a)b^k\varepsilon^k}{k!} \\
&= f(a) + bf'(a)\varepsilon + \sum_{k=2}^{\infty} \frac{f^{(k)}(a)b^k\varepsilon^k}{k!} \\
&= f(a) + bf'(a)\varepsilon + \varepsilon^2 \sum_{k=2}^{\infty} \frac{f^{(k)}(a)b^k\varepsilon^{k-2}}{k!} \\
&= f(a) + bf'(a)\varepsilon.
\end{aligned}$$

□

Пример 1. Ниже показаны вычисления функций от дуального числа $3 + 2\varepsilon$. Чтобы показать определение функции и точку вычисления использована запись [тело функции](аргумент), например, $[x^2 + 3](4)$ означает вычисление $f(x) = x^2 + 3$ в точке 4, т.е. $[x^2 + 3](4) = 4^2 + 3 = 4$.

- Константа

$$\begin{aligned}
f(x) &= 5, \\
f(3 + 2\varepsilon) &= [5](3) + 2 \cdot [0](3)\varepsilon = 5.
\end{aligned}$$

- Полином

$$\begin{aligned}
f(x) &= 4x^3 + x, \\
f(3 + 2\varepsilon) &= [4x^3 + x](3) + 2 \cdot [12x^2 + 1](3) \cdot \varepsilon \\
&= 111 + 2 \cdot 109 \cdot \varepsilon = 111 + 218\varepsilon.
\end{aligned}$$

- Логарифм

$$\begin{aligned}
f(x) &= \log x, \\
f(3 + 2\varepsilon) &= [\log x](3) + 2 \cdot [1/x](3) \cdot \varepsilon = \log 3 + \frac{2}{3}\varepsilon.
\end{aligned} \tag{5}$$

Рассмотрим теперь **сложение** функций

$$
\begin{aligned}
f(a+b\varepsilon)+g(a+b\varepsilon) &= (f(a)+bf'(a)\varepsilon)+(g(a)+bg'(a)\varepsilon)\\
&= (f(a)+g(a))+b(f'(a)+g'(a))\varepsilon.
\end{aligned}
$$

В действительной части мы получили сумму функций, а в дуальной присутствует сумма их производных. Положим $b = 1$ (см. (1))

$$
f(a+\varepsilon)+g(a+\varepsilon) = (f(a)+g(a))+(f'(a)+g'(a))\varepsilon.
$$

Теперь в дуальной части находится сумма производных функций, она же является производной суммы $(f+g)' = f'+g'$. Таким образом, сумма функций от дуального числа $a+\varepsilon$ является дуальным числом, действительная часть которого является суммой функций, а дуальная — производной суммы функций. Иначе говоря, в дуальных числах соблюдается правило дифференцирования суммы функций. Аналогичное верно для **вычитания**.

Проверим теперь **произведение** функций (см. (2))

$$
\begin{aligned}
f(a+\varepsilon)\times g(a+\varepsilon) &= (f(a)+f'(a)\varepsilon)\times(g(a)+g'(a)\varepsilon)\\
&= f(a)g(a)+(f'(a)g(a)+f(a)g'(a))\varepsilon.
\end{aligned}
$$

И снова есть соответствие. В действительной части получаем произведение функций, а в дуальной — производную произведения функций $(fg)' = f'g + fg'$ в точке a.

Наконец, проверим **деление** функций (см. (3))

$$
\begin{aligned}
\frac{f(a+\varepsilon)}{g(a+\varepsilon)} &= \frac{f(a)+f'(a)\varepsilon}{g(a)+g'(a)\varepsilon}\\
&= \frac{f(a)}{g(a)}+\frac{f'(a)g(a)-f(a)g'(a)}{g^2(a)}\varepsilon.
\end{aligned}
$$

И снова наблюдается соответствие. Действительная часть является делением функций, а дуальная — производной деления функций $(f/g)' = (f'g-fg')/g^2$ в точке a.

Последнее, что осталось проверить, это вычисление **сложной функции**

$$
g(f(a+\varepsilon)) = g(f(a)+f'(a)\varepsilon) = g(f(a))+g'(f(a))f'(a)\varepsilon.
$$

Как видно, наблюдается соответствие с правилом дифференцирования сложной функции $(g(f(x)))' = g'(f(x))f'(x)$.

3. Автоматическое дифференцирование

Тесная связь дуальных чисел с дифференцированием позволяет вычислять производные **автоматически**. Имеется ввиду следующее. Программисту требуется задать только функцию f, а код для вычисления производной f' писать

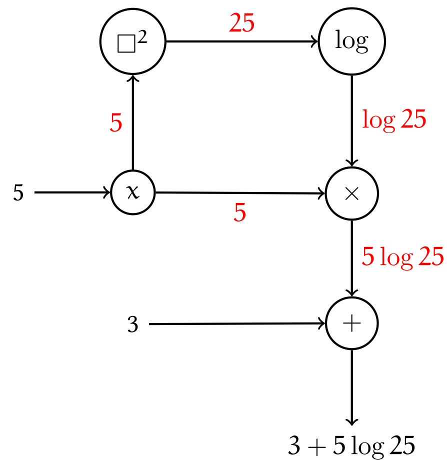

Рис. 1: Граф вычисления функции $f(x) = x \log x^2 + 3$ в точке $a = 5$. Промежуточные результаты отмечены цветом.

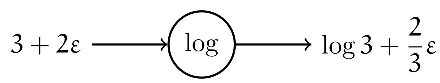

Рис. 2: Представление вычисления $\log x$ в $a = 3 + 2\varepsilon$. Сравните с примером (5).

не нужно: достаточно передать алгоритму функцию и точку вычисления, а алгоритм вернёт значение функции и её производную в заданной точке.

Мы рассмотрим только дифференцирование вперёд[2]. Этот вид автоматического дифференцирования использует правило дифференцирования сложной функции, углубляясь до тех пор, пока не останется «простая» функция. Проще всего рассмотреть пример.

Пусть нам нужно вычислить значение и производную функции $f(x) = x \log x^2 + 3$ в точке $a = 5$. Сначала построим граф вычисления функции (Рисунок 1). В вершинах этого графа расположены операции, входящее ребро показывает аргумент операции, а исходящее — результат операции. Похожий граф является промежуточным представлением исходного кода в компиляторе.

Как нам известно, для вычисления производной достаточно подсчитать $f(a+\varepsilon)$. Теперь для каждой операции аргумент и результат являются дуальными числами. Знакомый пример вычисления одной операции показан на Рисунке 2.

Теперь возьмём граф вычисления $f(x)$ (Рисунок 1), но на вход подадим $5+\varepsilon$. Воспользуемся изложенными выше правилами для дуальных чисел и получим граф, изображённый на Рисунке 3. В результате вычисления мы получили

$$f(5 + \varepsilon) = 3 + 5\log 25 + (2 + \log 25)\varepsilon.$$

[2] Forward mode (accumulation) automatic differentiation.

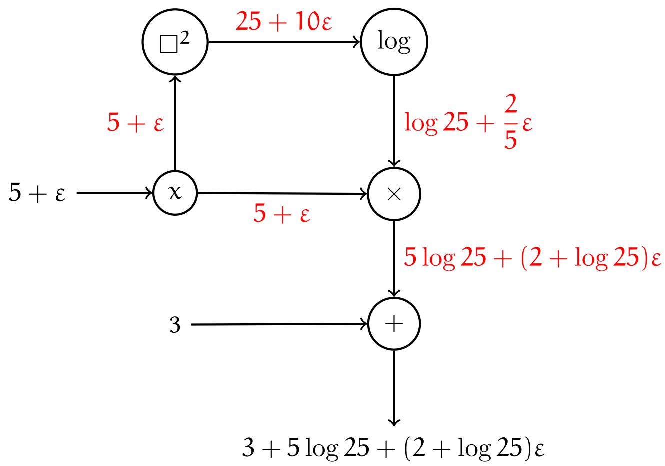

Рис. 3: Граф вычисления функции $f(x) = x\log x^2 + 3$ и её производной в точке $a = 5$. Промежуточные вычисления показаны цветом.

Как и ожидалось, действительная часть результата $3 + 5\log 25$ со значением функции $f(x) = x\log x^2 + 3$ в точке $a = 5$. В свою очередь, дуальная часть $2 + \log 25$ совпадает с производной f' в той же точке

$$f'(x) = \log x^2 + x\frac{2x}{x^2} = 2 + \log x^2, \quad f'(5) = 2 + \log 25.$$

Алгоритм автоматического дифференцирования вперёд теперь должен представляться несложным. Граф вычислений за нас строит компилятор, от нас же требуется:

- Определить тип данных для представления дуальных чисел;
- Определить приведение числовых типов данных к дуальному числу. Например, число 5 в дуальном представлении выглядит как $5 + 0\cdot\varepsilon$;
- Перегрузить необходимые операторы для дуальных чисел, например, для сложения и произведения;
- Перегрузить необходимые функции в соответствии с (4). Важно, что при этом нам нужны производные только от «простых» функций.

Так, мы можем «внедрить» автоматическое дифференцирование в существующую программу без переписывания исходного кода, задающего функцию f.

В Листинге 1 показан показан пример **реализации** автоматического дифференцирования на языке программирования Julia [1]. Стандартным пакетом для автоматического дифференцирования вперёд на текущий момент является ForwardDiff.jl [2]. Тот же пример с использованием пакета показан в Листинге 2.

```julia
struct Dual{T}
    a::T
    b::T
end

# Приведение типов.
Base.convert(::Type{<:Dual}, x::Real) = Dual(x, zero(x))
Base.promote_rule(::Type{Dual{T}}, ::Type{S}) where {T,S} = Dual{promote_type(T, S)}

Base.:+(x::Dual, y::Real) = +(promote(x, y)...)
Base.:+(x::Real, y::Dual) = +(promote(x, y)...)

# Перегрузка +, * и ^.
Base.:+(x::Dual, y::Dual) = Dual(x.a + y.a, x.b + y.b)
Base.:*(x::Dual, y::Dual) = Dual(x.a * y.a, x.b * y.a + x.a * y.b)
Base.:^(x::Dual, n::Int) = Dual(x.a^n, x.b * n * x.a^(n-1))

# Перегрузка логарифма.
Base.log(x::Dual) = Dual(log(x.a), x.b / x.a)

f(x) = 3 + x * log(x^2)
df(x) = 2 + log(x^2)

@show f(Dual(5, 1))  # Dual{Float64}(19.094379124341003, 5.218875824868201)
@show f(5)           # 19.094379124341003
@show df(5)          # 5.218875824868201
```

Лист. 1: Пример реализации автоматического дифференцирования на Julia.

```julia
using ForwardDiff

f(x) = 3 + x * log(x^2)

@show ForwardDiff.derivative(f, 5)  # 5.218875824868201
```

Лист. 2: Вычисление производной с помощью ForwardDiff.jl.

4. Заключение

В этой короткой заметке мы познакомились с дуальными числами и одним из их практических применений — автоматическим дифференцированием.

Мы рассмотрели только дифференцирование функции одного аргумента, однако, существуют расширения: нахождение частных производных, производных высших порядков, градиента, матриц Якоби и матриц Гессе.

Простота «внедрения» автоматического дифференцирования (по крайней мере, в Julia) в существующий код делают этот вычислительный способ особенно привлекательным для расчётов. При этом разработчики программных библиотек могут оставлять автоматическое дифференцирование в качестве способа нахождения производных по умолчанию. Пользователь, в свою очередь, может создавать лишь функции, перегружая библиотечный код по своему усмотрению.

5. Обратная связь

Исходный код примеров доступен в репозитории. Исправления и дополнения принимаются по почте или через механизм Issues в репозитории.

Список литературы

[1] Jeff Bezanson и др. «Julia: A Fresh Approach to Numerical Computing». В: *SIAM Review* 59.1 (янв. 2017), с. 65—98. ISSN: 0036-1445, 1095-7200. DOI: `10.1137/141000671`. URL: `https://epubs.siam.org/doi/10.1137/141000671` (дата обр. 04.08.2022).

[2] Jarrett Revels, Miles Lubin и Theodore Papamarkou. *Forward-Mode Automatic Differentiation in Julia*. 26 июля 2016. DOI: `https://doi.org/10.48550/arXiv.1607.07892`. arXiv: `1607.07892[cs]`. URL: `http://arxiv.org/abs/1607.07892` (дата обр. 13.04.2023).