ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА

УДК 519.688

К. К. Семенов

АВТОМАТИЧЕСКОЕ ДИФФЕРЕНЦИРОВАНИЕ ФУНКЦИЙ, ВЫРАЖЕННЫХ ПРОГРАММНЫМ КОДОМ

Алгебра дуальных чисел рассматривается как инструмент автоматического дифференцирования функций, выраженных программным кодом. Проанализированы специфические свойства дуальных чисел, необходимые для данного применения.

Ключевые слова: *автоматическое дифференцирование, дуальные числа.*

Введение. В практике программирования часто возникают задачи, когда необходимо помимо вычисления значений некоторой функции $f(x)$ вычислять значения ее производной $\frac{df(x)}{dx}$. В частности, в современной метрологии существует задача автоматической оценки интервала значений погрешности результатов вычислений $y = f(x)$ с помощью программ обработки, исходные данные x для которых являются результатами прямых измерений. Классическим способом учета трансформированной погрешности является оценка на основе производной $f'(x)$. Специфика задачи состоит в том, что функция $f(x)$ реализуется в виде программы вычислений и соответственно задана своим исходным кодом.

Традиционные способы оценки производной $f'(x)$ на практике сопряжены с трудностями и имеют существенные недостатки. Реализация вычислений $f'(x)$ в виде отдельной программной процедуры является избыточным решением и требует предварительного анализа функции $f(x)$. Использование метода конечных разностей требует обоснованного выбора значения приращения аргумента *x*, поскольку задача численного дифференцирования является некорректной и ее необходимо решать методом регуляризации [1].

Известен способ автоматического вычисления вместе с функцией $f(x_0)$ при некотором значении ее аргумента $x = x_0$ значения производной этой функции $f'(x_0)$ в той же точке [2]. Данный метод основан на алгебре дуальных чисел [3] и носит название „автоматическое дифференцирование“, подчеркивающее одновременность вычислений значений $f(x_0)$ и $f'(x_0)$ на основе только лишь исходного кода функции $f(x)$. К сожалению, ощущается нехватка отечественной литературы, посвященной данному методу. Настоящая статья призвана частично восполнить данный недостаток.

Дуальные числа. Рассмотрим множество чисел **z**, таких, что $\mathbf{z} = x + \varepsilon\Delta$, где $x \in R$, $\Delta \in R$, а $\varepsilon \neq 0$ — такая инфинитезимальная единица, что для нее выполняется точное равенство $\varepsilon^2 = 0$. Возможны математические операции сложения и умножения ε с вещественными числами. Числа $\mathbf{z}$ называют дуальными. Их составляющую x принято называть действительной частью $\mathbf{z}$ и обозначать $\operatorname{Re}\mathbf{z}$, а Δ — инфинитезимальной частью $\mathbf{z}$ и обозначать $\operatorname{Inf}\mathbf{z}$.

Рассмотрим разложение в ряд Тейлора гладкой функции $f(x)$ одного действительного аргумента:

$$f(x) = f(x_0) + \sum_{n=1}^{\infty} \frac{1}{n!} \frac{d^n f(x_0)}{dx^n} (x - x_0)^n .$$

Выполним его аналитическое продолжение на множество дуальных чисел: заменим в выражении разложения $f(x)$ в ряд действительный аргумент x на дуальный $\mathbf{z} = x + \varepsilon\Delta$. Тогда значение функции

$$f(x + \varepsilon\Delta) = f(x_0) + \sum_{n=1}^{\infty} \frac{1}{n!} \frac{d^n f(x_0)}{dx^n} (x - x_0 + \varepsilon\Delta)^n ,$$

согласно биному Ньютона, равно

$$f(x_0) + \sum_{n=1}^{\infty} \frac{1}{n!} \frac{d^n f(x_0)}{dx^n} \sum_{m=0}^{n} C_n^m (x - x_0)^{n-m} \varepsilon^m \Delta^m ,$$

что с учетом свойства $\varepsilon^2 = 0$ есть

$$f(x_0) + \sum_{n=1}^{\infty} \frac{1}{n!} \frac{d^n f(x_0)}{dx^n} \left((x - x_0)^n + n\varepsilon\Delta (x - x_0)^{n-1} \right) =$$

$$= \left(f(x_0) + \sum_{n=1}^{\infty} \frac{1}{n!} \frac{d^n f(x_0)}{dx^n} (x - x_0)^n \right) + \varepsilon\Delta \sum_{m=1}^{\infty} \frac{1}{(m-1)!} \frac{d^m f(x_0)}{dx^m} (x - x_0)^{m-1} = f(x) + \varepsilon\Delta \frac{df(x)}{dx} .$$

Получаем, что результат вычислений является дуальным числом, одна составляющая которого есть точное значение функции $f(x)$, а другая — точное значение производной этой же функции при том же значении аргумента x.

Таким образом, возможно использовать дуальные числа для автоматического вычисления значения производной $f'(x)$ **точно** вплоть до ошибки округления. Для того чтобы воспользоваться этим приемом, на практике необходимо создать новый тип данных. Поскольку дуальные числа $\mathbf{z} = x + \varepsilon\Delta$ суть пары вещественных чисел $\begin{pmatrix} x \\ \Delta \end{pmatrix}$, то можно на языке C++ ввести их следующим образом.

тип данных, описывающий дуальное число.

<pre>struct dual {
 double real;   // действительная часть.
 double inf;   // инфинитезимальная часть.
 }</pre>

Для того чтобы реализовать автоматическое дифференцирование функции $f(x)$, представленной своим исходным кодом, необходимо перейти в программе от вычислений с действительными данными (double) к вычислениям с дуальными числами (dual).

Вычисления, производимые в любой программе математической обработки, являют собой вычисления значений сложных функций, например, для вычисления значения $y = f\left(g_1\left(g_2\left(...g_n(x)...\right), x\right), x\right)$ необходимо начать с вычисления значения $y_n = g_n(x)$, продолжить вычислениями $y_{n-1} = g_{n-1}(y_n, x)$, $y_{n-2} = g_{n-2}(y_{n-1}, x)$ и т.д. Как известно, производная сложной функции $y = f\left(g_1\left(g_2\left(...g_n\left(x\right)...\right), x\right), x\right)$ по аргументу x вычисляется по цепному правилу, т.е.

$$\frac{dg_i\left(y_{i+1}, x\right)}{dx} = \frac{\partial g_i\left(y_{i+1}, x\right)}{\partial y_{i+1}} \cdot \frac{dg_{i+1}\left(y_{i+2}, x\right)}{dx} + \frac{\partial g_i\left(y_{i+1}, x\right)}{\partial x}$$

для $i = 1, 2, \ldots, n$. Следовательно, можно вычислять значение производной $\frac{df\left(x\right)}{dx}$ в последовательности вычисления значения функции $y = f\left(g_1\left(g_2\left(...g_n\left(x\right)...\right), x\right), x\right)$, т.е. начиная с $\frac{dg_n\left(x\right)}{dx}$, продолжая вычислением

$$\frac{dg_{n-1}\left(y_n, x\right)}{dx} = \frac{\partial g_{n-1}\left(y_n, x\right)}{\partial y_n} \cdot \frac{dg_n\left(x\right)}{dx} + \frac{\partial g_{n-1}\left(y_n, x\right)}{\partial x}$$

и т.д. вплоть до

$$\frac{df\left(x_1\right)}{dx_1} = \frac{\partial g_1\left(y_2, x\right)}{\partial y_2} \cdot \frac{dg_2\left(x\right)}{dx} + \frac{\partial g_1\left(y_2, x\right)}{\partial x}.$$

Таким образом, можно выполнить автоматическое дифференцирование одновременно с **вычислениями основной функции**.

На практике математические преобразования, реализуемые компьютерными программами, сводятся к тому, что все функции g_i, вообще говоря, являют собой последовательность элементарных арифметических действий из набора $\{+, -, \times, /\}$ и элементарных функций — математических примитивов из стандартных библиотек. Можно констатировать, что приведенное рассуждение о вычислении производной сложной функции справедливо и для функции, реализованной программой. Таким образом, для работы с дуальными числами необходимо задать основные арифметические действия с ними и указать, как производить вычисления элементарных математических функций при дуальном аргументе.

С математической точки зрения эта задача равносильна определению алгебры дуальных чисел $\Lambda = \left\langle R^2, +, \times \right\rangle$ как соответствующей структуры над множеством пар действительных чисел R^2 с операциями сложения и умножения.

Программно-ориентированное задание алгебры дуальных чисел. Рассмотрим множество пар действительных чисел R^2 и зададим для двух произвольных его элементов $\mathbf{z}_1 = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix}$ и $\mathbf{z}_2 = \begin{pmatrix} x_2 \\ y_2 \end{pmatrix}$ арифметические операции следующим образом.

Сложение будем выполнять по правилу

$$\mathbf{z}_1 + \mathbf{z}_2 = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} + \begin{pmatrix} x_2 \\ y_2 \end{pmatrix} = \begin{pmatrix} x_1 + x_2 \\ y_1 + y_2 \end{pmatrix}. \quad (1)$$

Операция обладает свойствами коммутативности и ассоциативности. Действительно, выполнены соответствующие соотношения $\mathbf{z}_1 + \mathbf{z}_2 = \mathbf{z}_2 + \mathbf{z}_1$ и $\left(\mathbf{z}_1 + \mathbf{z}_2\right) + \mathbf{z}_3 = \mathbf{z}_1 + \left(\mathbf{z}_2 + \mathbf{z}_3\right)$.

Существует нейтральный элемент операции сложения $\varnothing=\begin{pmatrix}0\\0\end{pmatrix}$, справедливо равенство $\mathbf{z}+\varnothing=\begin{pmatrix}x\\y\end{pmatrix}+\begin{pmatrix}0\\0\end{pmatrix}=\begin{pmatrix}x\\y\end{pmatrix}=\mathbf{z}$.

Вычитание — обратная сложению операция. При выводе правил для компонентов ее результата требуется решить уравнение $\begin{pmatrix}x_1\\y_1\end{pmatrix}=\begin{pmatrix}x_2\\y_2\end{pmatrix}+\begin{pmatrix}x_3\\y_3\end{pmatrix}$ относительно $\begin{pmatrix}x_3\\y_3\end{pmatrix}$. Получаем систему уравнений $\begin{cases}x_1=x_2+x_3\\y_1=y_2+y_3\end{cases}$, ее решением является $\begin{cases}x_3=x_1-x_2\\y_3=y_1-y_2\end{cases}$. Следовательно,

$$\mathbf{z}_1-\mathbf{z}_2=\begin{pmatrix}x_1\\y_1\end{pmatrix}-\begin{pmatrix}x_2\\y_2\end{pmatrix}=\begin{pmatrix}x_1-x_2\\y_1-y_2\end{pmatrix}. \qquad (2)$$

Умножение: введем операцию умножения как

$$\mathbf{z}_1\times\mathbf{z}_2=\begin{pmatrix}x_1\\y_1\end{pmatrix}\times\begin{pmatrix}x_2\\y_2\end{pmatrix}=\begin{pmatrix}x_1\cdot x_2\\y_1\cdot x_2+y_2\cdot x_1\end{pmatrix}. \qquad (3)$$

Операция умножения также коммутативна и ассоциативна, поскольку $\mathbf{z}_1\times\mathbf{z}_2=\mathbf{z}_2\times\mathbf{z}_1$ и $(\mathbf{z}_1\times\mathbf{z}_2)\times\mathbf{z}_3=\mathbf{z}_1\times(\mathbf{z}_2\times\mathbf{z}_3)$. Для нее существует нейтральный элемент $1=\begin{pmatrix}1\\0\end{pmatrix}$. Действительно, $\mathbf{z}\times\mathbf{1}=\begin{pmatrix}x\\y\end{pmatrix}\times\begin{pmatrix}1\\0\end{pmatrix}=\begin{pmatrix}x\cdot 1\\y\cdot 1+0\cdot x\end{pmatrix}=\begin{pmatrix}x\\y\end{pmatrix}=\mathbf{z}$.

Деление — обратная умножению операция. При выводе правил для компонентов ее результата требуется решить уравнение $\begin{pmatrix}x_1\\y_1\end{pmatrix}=\begin{pmatrix}x_2\\y_2\end{pmatrix}\times\begin{pmatrix}x_3\\y_3\end{pmatrix}$ относительно $\begin{pmatrix}x_3\\y_3\end{pmatrix}$. Получаем систему уравнений

$$\begin{cases}x_1=x_2x_3\\y_1=y_2x_3+y_3x_2\end{cases}\Rightarrow\begin{cases}x_3=\dfrac{x_1}{x_2}\\y_3x_2=y_1-y_2\dfrac{x_1}{x_2}\end{cases}\Rightarrow\begin{cases}x_3=\dfrac{x_1}{x_2}\\y_3=\dfrac{y_1x_2-y_2x_1}{x_2^2}\end{cases}.$$

В качестве вывода получаем, что при $x_2\neq 0$

$$\frac{\mathbf{z}_1}{\mathbf{z}_2}=\begin{pmatrix}x_1\\y_1\end{pmatrix}\Big/\begin{pmatrix}x_2\\y_2\end{pmatrix}=\begin{pmatrix}\dfrac{x_1}{x_2}\\\dfrac{y_1x_2-y_2x_1}{x_2^2}\end{pmatrix}. \qquad (4)$$

Имеет место также **дистрибутивный закон**: $\mathbf{z}_1\times(\mathbf{z}_2+\mathbf{z}_3)=\mathbf{z}_1\times\mathbf{z}_2+\mathbf{z}_1\times\mathbf{z}_3$.

Для таким образом введенной алгебры $\Lambda=\left\langle R^2,+,\times\right\rangle$ дуальных чисел справедливо следующее свойство:

$$\begin{pmatrix}0\\y_1\end{pmatrix}\times\begin{pmatrix}0\\y_2\end{pmatrix}=\begin{pmatrix}0\\0\end{pmatrix},$$

т.е. для квадрата чисто инфинитезимального числа $\mathbf{z}$ всегда имеет место свойство $\mathbf{z}^2 = 0$. Таким образом, из свойств арифметических операций (1)—(4) для инфинитезимальной единицы ε вытекает ее свойство $\varepsilon^2 = 0$.

В силу того что справедливы соотношения

$$\operatorname{Re}(\mathbf{z}_1 \pm \mathbf{z}_2) = \operatorname{Re}(\mathbf{z}_1) \pm \operatorname{Re}(\mathbf{z}_2),$$
$$\operatorname{Re}(\mathbf{z}_1 \times \mathbf{z}_2) = \operatorname{Re}(\mathbf{z}_1)\operatorname{Re}(\mathbf{z}_2),$$
$$\operatorname{Re}(\mathbf{z}_1 / \mathbf{z}_2) = \operatorname{Re}(\mathbf{z}_1) / \operatorname{Re}(\mathbf{z}_2),$$

алгебра Λ описывает такое расширение множества действительных чисел на двумерную плоскость, которое включает в себя поле действительных чисел как составную часть. Действия с инфинитезимальными составляющими дуальных чисел не влияют на значения их действительных составляющих.

Результат вычисления элементарной функции $g(\mathbf{z})$ от дуального аргумента $\mathbf{z} = \begin{pmatrix} x \\ y \end{pmatrix}$ равен

$$g(\mathbf{z}) = \begin{pmatrix} g(x) \\ yg'(x) \end{pmatrix},$$

что следует из рассмотренного выше аналитического продолжение ряда Тейлора для функции $g(x)$. Указанное свойство задает правило вычисления элементарных функций от дуального аргумента, которое необходимо использовать для перегрузки математических примитивов.

процедура для вычислений значения тригонометрического синуса	процедура для вычислений значения гиперболического синуса

`dual sin (dual x) {`
`dual z;`
`z.real = sin(x.real);`
`z.inf = x.inf * cos(x.real);`
`return (z); }`
	`dual sinh (dual x) {`
	`dual z;`
	`z.real = sinh(x.real);`
	`z.inf = x.inf * cosh(x.real);`
	`return (z); }`

Рассмотрим пример. Пусть требуется вычислить значение функции $f(x) = \sin(x)\exp(-x^2) + x$ в точке x_0. В соответствии с представленными выше правилами произведем последовательно вычисления, заменив действительный аргумент $x = x_0$ дуальным $\mathbf{z}_0 = \begin{pmatrix} x_0 \\ 1 \end{pmatrix}$.

$$f(\mathbf{z}_0) = \sin\left[\begin{pmatrix} x_0 \\ 1 \end{pmatrix}\right] \times \exp\left[-\begin{pmatrix} x_0 \\ 1 \end{pmatrix} \times \begin{pmatrix} x_0 \\ 1 \end{pmatrix}\right] + \begin{pmatrix} x_0 \\ 1 \end{pmatrix} = \begin{pmatrix} \sin(x_0) \\ 1 \cdot \cos(x_0) \end{pmatrix} \times \exp\left[\begin{pmatrix} -x_0^2 \\ -2x_0 \end{pmatrix}\right] + \begin{pmatrix} x_0 \\ 1 \end{pmatrix} =$$

$$= \begin{pmatrix} \sin(x_0) \\ \cos(x_0) \end{pmatrix} \times \begin{pmatrix} \exp(-x_0^2) \\ -2x_0 \exp(-x_0^2) \end{pmatrix} + \begin{pmatrix} x_0 \\ 1 \end{pmatrix} = \begin{pmatrix} \sin(x_0)\exp(-x_0^2) \\ -2x_0 \sin(x_0)\exp(-x_0^2) + \cos(x_0)\exp(-x_0^2) \end{pmatrix} + \begin{pmatrix} x_0 \\ 1 \end{pmatrix} =$$

$$= \begin{pmatrix} \sin(x_0)\exp(-x_0^2) + x_0 \\ -2x_0 \sin(x_0)\exp(-x_0^2) + \cos(x_0)\exp(-x_0^2) + 1 \end{pmatrix}.$$

Действительно, $f'(x) = -2x\exp(-x^2)\sin(x) + \cos(x)\exp(-x^2) + 1$, что в точности совпадает с результатом вычислений при помощи дуальных чисел.

Если, например, $x_0 = 0$, то получим $\mathbf{z}_0 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ и

$$\sin z_0 = \sin\left[\begin{pmatrix} 0 \\ 1 \end{pmatrix}\right] = \begin{pmatrix} \sin 0 \\ 1 \cdot \cos 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix},$$

$$\exp\left(-z_0^2\right) = \exp\left[-\begin{pmatrix} 0 \\ 1 \end{pmatrix} \times \begin{pmatrix} 0 \\ 1 \end{pmatrix}\right] = \exp\left[\begin{pmatrix} 0 \\ 0 \end{pmatrix}\right] = \begin{pmatrix} \exp 0 \\ 0 \cdot \exp 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

Следовательно,

$$f\left(\mathbf{z}_0\right) = \sin z_0 \times \exp\left(-z_0^2\right) + z_0 = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \times \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 2 \end{pmatrix},$$

откуда получаем, что $f(0) = 0$ и $f'(0) = 2$.

Рассмотренный метод давно и с успехом применяется в самых различных приложениях. Представленное выше описание соответствует так называемому прямому ходу (*forward mode*) автоматического дифференцирования, когда значение производной вычисляется параллельно вычислению значения функции при движении от начала алгоритма к его концу. Метод легко обобщается на случай одновременного вычисления всех частных производных одновременно с вычислением значения функции.

Из всего сказанного следует, что метод автоматического дифференцирования характеризуется следующими свойствами:

— автоматическое дифференцирование функций, выраженных программным кодом, носит аналитический характер и позволяет получить точное (вплоть до ошибок округления) значение производной;

— использование подхода не влияет на сходимость задействованных в выполняемой программе итерационных методов и не искажает конечного результата вычислений;

— автоматическое дифференцирование позволяет выполнить программу вычислений лишь единожды в отличие от приближенных методик численного дифференцирования на основе конечных разностей;

— использование автоматического дифференцирования позволяет не изменять порядок вычислений в программах.

Заключение. Метод автоматического дифференцирования функций, выраженных программным кодом, обладает рядом полезных для метрологической практики свойств, что обусловливает возможность его применения в метрологических целях [4, 5]. В частности, с помощью метода можно автоматически получать оценки характеристик трансформированной погрешности результатов вычислений, выполняемых в измерительных системах с цифровой обработкой сигналов [6].

СПИСОК ЛИТЕРАТУРЫ

1. *Тихонов А. Н., Арсенин В. Я.* Методы решения некорректных задач. М.: Наука, Главная редакция физико-математической литературы, 1979.

2. *Corliss G., Faure C., Griewank A., Hascoлt L., Naumann U.* Automatic Differentiation Bibliography // Automatic Differentiation of Algorithms: From Simulation to Optimization. Springer, 2002. P. 383—425.

3. *Яглом И. М.* Комплексные числа и их применение в геометрии. М.: Государственное изд-во физико-математической литературы, 1963.

4. *Hall B. D.* Calculating measurements uncertainty using automatic differentiation // Measurement Science and Technology. 2002. Vol. 13, N 4. P. 421—427.

5. *Luca Mari.* A computational system for uncertainty propagation of measurement results // Measurement. 2009. Vol. 42, is. 6. P. 844—855.

6. *Семенов К. К., Солопченко Г. Н.* Теоретические предпосылки реализации метрологического автосопровождения программ обработки результатов измерений // Измерительная техника. 2010. № 6. С. 9—14.

Сведения об авторе

Константин Константинович Семенов — аспирант; Санкт-Петербургский государственный политехнический университет, кафедра измерительных информационных технологий; E-mail: semenov.k.k@gmail.com

Рекомендована кафедрой измерительных информационных технологий

Поступила в редакцию 23.06.11 г.