Background and Implementation

Nick Attkiss

October 20, 2022

Background

Our automatic differentiation package relies on the elegant arithmetic of dual numbers to compute derivatives of complicated real-valued functions in terms of the derivatives of their simpler pieces.

A *dual number* is a symbol of the form $z = a + \epsilon b$, where $a, b \in \mathbb{R}$; a is called the *real part* of z and b is called the *dual part* of z. We denote the set of all dual numbers by $\mathbb{D}$. Dual numbers can be added and multiplied according the following formulas:

$$
\begin{aligned}
(a_1 + \epsilon b_1) + (a_2 + \epsilon b_2) &= (a_1 + a_2) + \epsilon(b_1 + b_2), \\
(a_1 + \epsilon b_1)(a_2 + \epsilon b_2) &= (a_1 a_2) + \epsilon(a_1 b_2 + a_2 b_1).
\end{aligned}
$$

The beauty of using dual numbers to compute derivatives of real-valued functions lies in the following construction: Let $f : \mathbb{R}^n \to \mathbb{R}^m$ be a differentiable function. From f we can define a function $f_* = F : \mathbb{D}^n \to \mathbb{D}^m$ whose ith component $F_i : \mathbb{D}^n \to \mathbb{D}$ is given by

$$F_i(x_1 + \epsilon p_1, \ldots, x_n + \epsilon p_n) = f_i(x) + \epsilon D_p f(x),$$

where $x = (x_1, \ldots, x_n)$, $p = (p_1, \ldots, p_n)$, and $D_p f_i(x)$ is the directional derivative of f_i in the direction of p, evaluated at x. We call f_* the *augmentation* of f. The sum, product, and chain rules of calculus combine with the operations of addition and multiplication of dual numbers in an extremely nice way. Specifically, let $f, g : \mathbb{R}^n \to \mathbb{R}^m$ and let $h : \mathbb{R}^m \to \mathbb{R}^k$. Then we have

$$
\begin{aligned}
(f + g)_* &= f_* + g_* \\
(fg)_* &= f_* g_* \\
(h \circ f)_* &= h_* \circ f_*.
\end{aligned}
$$

Implementation and Use

Our package `AutomaticDifferentiation` contains a class `DualNumber`, instances of which can be added and multiplied using standard Python syntax. Instances of `DualNumber` have two attributes, `real` and `dual`, which are the real and dual parts respectively.

`AutomaticDifferentiation` also contains augmentations of standard functions such as sin, cos, exp, and log. Suppose we wish to compute the first partial derivative of the function $f : \mathbb{R}^2 \to \mathbb{R}$ given by

$$f(x_0, x_1) = \sin(3x_0 + 2x_1)$$

at the point $x = (4, 5)$. To do so, we would use the following code:

```
import AutomaticDifferentiation as ad

x = [4,5]
p = [1,0]

dual_output = ad.sin(DualNumber(3,0) * DualNumber(x[0],p[0])
              + DualNumber(2,0) * DualNumber(x[1],p[1]))

derivative = dual_output.dual
```