AutoDiff

Eine Einführung in algorithmisches Ableiten

Michael Brand

19/10/2022

Inhaltsverzeichnis

Vorwort

Einleitung

Danksagung

1 Ableitungen und ihre Anwendungen

1.1 Ableitungen von Funktionen

Wir kennen Ableitungen von Funktionen $f : \mathbb{R} \to \mathbb{R}$ aus dem Mathematikunterricht. Sie geben uns darüber Auskunft, wie gross die Steigung der Tangente in einem bestimmten Punkt des Funktionsgraphen ist. Die Tangente stellt dabei die beste lineare Annäherung an den Funktionsgraph dar. Ableitungen beschreiben auch die lokale Änderungsrate der Funktion. Ableitungen erlauben es uns ausserdem, die Extrema und Wendepunkte einer Funktion zu bestimmen.

Die folgende Tabelle fasst die bekannten Ableitungen der Grundfunktionen zusammen.

Tabelle 1.1: Ableitungen der Grundfunktionen

$f(x)$	$f'(x)$	$f(x)$	$f'(x)$
x^n	$n \cdot x^n \quad (n \in \mathbb{R})$	$\sqrt{x}$	$\frac{1}{2\cdot\sqrt{x}}$
e^x	e^x	a^x	$a^x \cdot \ln(a) \quad (a > 0, a \neq 1)$
$\ln(x)$	$\frac{1}{x}$	$\log_a(x)$	$\frac{1}{x\cdot\ln(a)} \quad (a > 0, a \neq 1)$
$\sin(x)$	$\cos(x)$	$\arcsin(x)$	$\frac{1}{\sqrt{1-x^2}}$
$\cos(x)$	$-\sin(x)$	$\arccos(x)$	$-\frac{1}{\sqrt{1-x^2}}$
$\tan(x)$	$\frac{1}{\cos^2(x)} = 1 + \tan^2(x)$	$\arctan(x)$	$\frac{1}{x^2+1}$
$\sinh(x)$	$\cosh(x)$	$\operatorname{arsinh}(x)$	$\frac{1}{\sqrt{x^2+1}}$
$\cosh(x)$	$\sinh(x)$	arcosh(x)	$\frac{1}{\sqrt{x^2-1}}$

$f(x)$	$f'(x)$	$f(x)$	$f'(x)$
$\tanh(x)$	$\frac{1}{\cosh^2(x)} = 1 - \tanh^2(x)$	artanh(x)	$-\frac{1}{x^2-1}$

Neue Funktionen erhält man, indem man die Grundfunktionen aus Tabelle **??** addiert, subtrahiert, multipliziert, dividiert und komponiert, d.h. Verkettungen der Form $(f \circ g)(x) = f(g(x))$ bildet. Um solche Funktionen abzuleiten, brauchen wir die Regeln aus Tabelle **??**. Mit diesen Regeln sind wir dann schon in der Lage, alle differenzierbaren Funktionen abzuleiten.

Tabelle 1.2: Ableitungsregeln

Summenregel	$\frac{d}{dx}(f(x) \pm g(x)) = f'(x) \pm g'(x)$
Produktregel	$\frac{d}{dx}(f(x) \cdot g(x)) = f'(x) \cdot g(x) + f(x) \cdot g'(x)$
Spezialfall: Faktorregel	$\frac{d}{dx}(a \cdot f(x)) = a \cdot f'(x)$
Quotientenregel	$\frac{d}{dx}\frac{f(x)}{g(x)} = \frac{f'(x) \cdot g(x) - f(x) \cdot g'(x)}{g(x)^2}$
Kettenregel	$\frac{d}{dx}f(g(x)) = f'(g(x)) \cdot g'(x)$

An dieser Stelle sei noch angemerkt, dass sich der Begriff der Ableitung sinngemäss auf Funktionen $f : \mathbb{R}^n \to \mathbb{R}^m$ verallgemeinern lässt. Eine kurze Beschreibung der Grundidee findet sich in Slater (2022). Weitergehende Informationen findet man z.B. in Arens u. a. (2022) oder in jedem Lehrbuch zur Analysis 2.

1.2 Programme als Funktionen

Programme, die numerische Werte einlesen und numerische Werte ausgeben, können als mathematische Funktionen betrachtet werden. Wir beschränken uns zunächst auf Programme, die nur ein Argument erhalten und nur einen Rückgabewert liefern.

Beispiel 1.1 (Eine Funktion als Programm).

```
def f(x):
    y = (2 + x) * (x - 3)
    return y

x0 = 2
print( f(x0) )
```

Diese Python-Funktion entspricht der Funktion $f : \mathbb{R} \to \mathbb{R}, x \mapsto y = (2 + x)(x - 3)$ im Sinne der Mathematik. Natürlich kann der Funktionskörper viel komplizierter aufgebaut sein und z.B. Schleifen und Bedingungen enthalten.

Um zu verstehen, wie der Computer einen Ausdruck wie `y = (2 + x) * (x - 3)` auswertet, ist es hilfreich, ihn als Baum (im Sinne der Graphentheorie) darzustellen. Ausdrucksbäume sind ein Spezialfall von so genannten *computational graphs* und werden z.B. in Hromkovic u. a. (2021) erklärt.

Wir wollen nun unsere Python-Funktion so umschreiben, dass diese Struktur auch im Funktionskörper sichtbar wird. Dazu führen wir drei Hilfsvariablen `v0, v1, v2` ein.

```
def f(x):
    v0 = x
    v1 = 2 + v0
    v2 = v0 - 3
    y = v1 * v2
    return y
```

! Konvention

Eine Funktion berechnet aus einem Argument `x` einen Rückgabewert `y` über eine Reihe von Hilfsvariablen `v`, die mit aufsteigenden Indizes versehen sind. Dabei setzen wir am Anfang immer `v0 = x`.

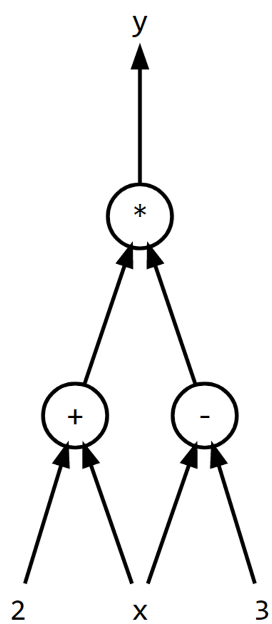

Abbildung 1.1: Ausdrucksbaum zum Ausdruck `y = (2 + x) * (x - 3)`.