Über die Beziehung des Rings der dualen Zahlen zur automatischen Differentiation

Gerhard Heindl (Ebersberg)

mail@gerhardheindl.de

Der Ring $\mathbb{D}$ der dualen Zahlen

Der Ring der dualen Zahlen lässt sich analog zum Körper der komplexen Zahlen konstruieren. Bekanntlich ist die Menge $\mathbb{R} \times \mathbb{R}$ der reellen Zahlenpaare (x, y) bezüglich der Addition

$$(A) : (x_1, y_1) + (x_2, y_2) := (x_1 + x_2, y_1 + y_2)$$

und der Multiplikation

$$(M_C) : (x_1, y_1)*(x_2, y_2) := (x_1x_2-y_1y_2, x_1y_2+y_1x_2)$$

ein Körper, nämlich der Körper $\mathbb{C}$ der komplexen Zahlen. Die Schreibweise $x + iy$ für die komplexe Zahl (x, y) leitet sich aus der Gleichung

$$(S) : (x, y) = (x, 0) + (0, 1) * (y, 0)$$

ab, wenn man $i = (0, 1)$ setzt und für jedes $x \in \mathbb{R}$, $(x, 0)$ mit x identifiziert.

Bezüglich der Addition (A) und der Multiplikation

$$(M_D) : (x_1, y_1) * (x_2, y_2) := (x_1x_2, x_1y_2 + y_1x_2)$$

ist $\mathbb{R} \times \mathbb{R}$ nur ein kommutativer Ring, der sogenannte Ring $\mathbb{D}$ der dualen Zahlen. Es sind nur die Zahlenpaare (x, y) invertierbar, für die x von 0 verschieden ist. Für $x \neq 0$ hat man

$$(x, y)^{-1} = \left(\frac{1}{x}, -\frac{y}{x^2}\right),$$

und folglich für $x_2 \neq 0$:

$$\begin{aligned}(x_1, y_1)/(x_2, y_2) &:= (x_1, y_1) * (x_2, y_2)^{-1} \\ &= \left(\frac{x_1}{x_2}, \frac{x_2y_1 - y_2x_1}{x_2^2}\right).\end{aligned}$$

Offenbar erweitern die betrachteten arithmetischen Operationen die gleichnamigen Operationen für reelle Zahlen auf duale Zahlen, wenn man, was im Folgenden stets angenommen werde, $(x, 0) \in \mathbb{D}$ mit $x \in \mathbb{R}$ identifiziert.

Duale Zahlen wurden erstmals von E. Study in [4], S. 195-197, eingeführt und für kinematische und geometrische Anwendungen benutzt. Da die Darstellung (S) auch bei Ersetzung von (M_C) durch (M_D) gilt, wird für die duale Zahl (x, y) üblicherweise $x + \varepsilon y$ oder $x + y\varepsilon$ geschrieben, wobei jetzt $(0, 1)$ mit ε bezeichnet wird (siehe z. B. [4] S. 195 oder [2] § 30*). Während bei Zugrundelegung der Multiplikation (M_C) für komplexe Zahlen $i^2 = -1$ ist, gilt bei der Zugrundelegung der Multiplikation (M_D) für duale Zahlen $\varepsilon^2 = (0, 1) * (0, 1) = 0$. Im Folgenden soll jedoch für duale Zahlen die kürzere Schreibweise (x, y) beibehalten werden.

Die Erweiterung von differenzierbaren Funktionen

Für jede differenzierbare Funktion $h : \mathbb{R} \supset D_h \to \mathbb{R}$, D_h offen, führen wir die Erweiterung

$$h_d : D_h \times \mathbb{R} \ni (x, y) \mapsto (h(x), yh'(x)) \in \mathbb{D}$$

von h zu einer Funktion von der Teilmenge $D_h \times \mathbb{R}$ von $\mathbb{D}$ in $\mathbb{D}$ ein.

Sind f, g differenzierbare Funktionen, so folgert man aus $(A), (M_D)$, der Formel für die Division von dualen Zahlen, sowie den bekannten Rechenregeln für differenzierbare Funktionen:
Ist $D := D_f \cap D_g \neq \emptyset$, dann sind $f \pm g$ und $f * g$ definiert mit $D_{f\pm g} = D_{f*g} = D$ und es gilt für alle $(x, y) \in D \times \mathbb{R}$:

$$f_d(x, y) \pm g_d(x, y) = (f \pm g)_d(x, y),$$
$$f_d(x, y) * g_d(x, y) = (f * g)_d(x, y).$$

Ist $D' := D_f \cap \{x \in D_g : g(x) \neq 0\} \neq \emptyset$, so ist f/g definiert mit $D_{f/g} = D'$ und es gilt für alle $(x, y) \in D' \times \mathbb{R}$:

$$f_d(x, y)/g_d(x, y) = (f/g)_d(x, y).$$

Neben diesen algebraischen Regeln ist für die automatische Differentiation vor allem die folgende Regel von Bedeutung:

Die Funktionen h und g seien differenzierbar, und $g(D_g)$ sei eine Teilmenge von D_h. Dann gilt für alle $(x, y) \in D_g \times \mathbb{R}$:

$$h_d(g_d(x, y)) = (h \circ g)_d(x, y).$$

(Beweis: Für $(x, y) \in D_g \times \mathbb{R}$ ist $g_d(x, y) = (\xi, \eta)$ mit $\xi = g(x)$ und $\eta = yg'(x)$. Daher ist

$$h_d(g_d(x, y)) = h_d(\xi, \eta) = (h(\xi), \eta h'(\xi)) =$$

$$(h(g(x)), yg'(x)h'(g(x))) = (h \circ g)_d(x, y),$$

wobei die letzte Gleichung aus der Kettenregel folgt.)

Bemerkung zur Verknüpfung von differenzierbaren Funktionen mit konstanten Funktionen: Ist $c \in \mathbb{R}$, und f differenzierbar, so bezeichnet man die Funktion $D_f \ni x \mapsto f(x) \pm c$ üblicherweise mit $f \pm c$. Dabei wird also mit c auch die Funktion bezeichnet, die auf D_f den konstanten Wert c annimmt. Entsprechend sind Verknüpfungen wie $c \pm f, c * f, f * c, c/g$ und im Fall $c \neq 0$, g/c zu betrachten. Da immer klar ist, in welchem Kontext c auftritt, sind Verwechslungen nicht zu befürchten.

Automatische Differentiation

Eine, mit Hilfe der in einer Programmiersprache zur Verfügung stehenden algebraischen Operatoren $+, -, *, /$ und differenzierbaren Standardfunktionen zusammengesetzte Funktion q in einem in der Programmiersprache realisierbaren $x \in D_g$ automatisch zu differenzieren, bedeutet $q(x)$ und $q'(x)$ (bis auf Rundungsfehler) dadurch zu ermitteln, indem man

$$q_d(x, 1) = (q(x), q'(x))$$

berechnet. Es wird damit vermieden, dass man einen Funktionsausdruck für q' berechnen, und anschließend die Werte $q(x)$ und $q'(x)$ getrennt auswerten muss, um z. B. (im Fall $q'(x) \neq 0$) einen Newtonschritt

$$x - \frac{q(x)}{q'(x)}$$

durchzuführen. Selbst wenn man die Bestimmung eines Funktionsausdrucks für q' einem Computer-Algebra-Programm überlässt, so ist der Aufwand für eine anschließende getrennte Berechnung von $q(x)$ und $q'(x)$ wegen der dabei auftretenden Mehrfachberechnungen von Zwischenresultaten erheblich höher als der für die simultane Berechnung von $(q(x), q'(x))$ als $q_d(x, 1)$.

Ein einfaches, aber repräsentatives Beispiel:

Nehmen wir an, wir wollen für die für alle $x \in \mathbb{R}$ durch

$$q(x) := \frac{(\cos(x) - 1) * (\sin(x) + 3)}{\operatorname{asinh}(\cos(x) * \sin(x)) + 2}$$

definierte differenzierbare Funktion q sowohl $q(5)$ als auch $q'(5)$ bestimmen.

Man setzt dann also $w := (5, 1)$ und berechnet

$$\frac{(\cos_d(w) - 1) * (\sin_d(w) + 3)}{\operatorname{asinh}_d(\cos_d(w) * \sin_d(w)) + 2},$$

wobei die arithmetischen Operatoren die in $\mathbb{D}$ eingeführten sind. Aus den bereitgestellten Rechenregeln für erweiterte differenzierbare Funktionen folgt, dass das Ergebnis

$$q_d(w) = q_d(5, 1) = (q(5), q'(5))$$

ist. Entscheidend ist dabei, dass die arithmetischen Operatoren und für die Standardfunktionen $\cos, \sin, \operatorname{asinh}$ ihre Erweiterungen $\cos_d, \sin_d, \operatorname{asinh}_d$ programmierbar sind. Für alle $(x, y) \in \mathbb{R} \times \mathbb{R}$:

$$\begin{aligned}\cos_d(x, y) &= (\cos(x), -y\sin(x)),\ \sin_d(x, y) \\ &= (\sin(x), y\cos(x)), \\ \operatorname{asinh}_d(x, y) &= (\operatorname{asinh}(x), \frac{y}{\sqrt{1 + x^2}}).\end{aligned}$$

Das ist auch für die üblichen anderen differenzierbaren Standardfunktionen der Fall. Die Auswertung von $q_d(w)$ erfolgte mit einer Matlab-Funktion (für die nur die tatsächlich benötigten Operatoren und Funktionen programmiert wurden) mit dem Ergebnis

$$\begin{aligned}&(q(5), q'(5)) \\ &= (-0.844540587580294, 0.618202155258725).\end{aligned}$$

Eine umfassende Klasse von entsprechenden Operatoren und Funktionen wurde von C. Taylor [5] programmiert. Eine erste (eingeschränktere) Klasse dieser Art wurde, ohne Bezug auf duale Zahlen zu nehmen, von L.B. Rall in [3] entwickelt.

Die hier über die duale Arithmetik eingeführte Automatische Differentiation lässt sich in naheliegender Weise auf eine automatische Berechnung der Koeffizienten von Taylorpolynomem zu gegebenen Entwicklungspunkten verallgemeinern. Ferner existieren verschiedene Verfahren zur automatischen Berechnung von partiellen Ableitungen beliebiger Ordnung für entsprechend oft differenzierbare Funktionen von mehreren Veränderlichen (siehe z. B. [1]).

Literatur

[1] A. Griewank, A. Walther, *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiations*, SIAM, 2008

[2] B. Hornfeck, *Algebra*, Walter de Gruyter & Co, Berlin, 1969

[3] L.B. Rall, *The Artihmetic of Differentiation*, University of Wisconsin - Madison, Mathematics Research Center, Technical Summary Report 2688, 1984

[4] E. Study, *Geometrie der Dynamen, Die Zusammensetzung von Kräften und verwandte Gegenstände der Geometrie*, B.G. Teubner, 1903

[5] C. Taylor, *Matlab dual number class (for automatic differentiation)*, https://gist.github.com/chris-taylor/2005955