AD und Parametersensitivitäten für ODE

Hier noch einmal die drei Argumente, warum AD für Parametersensitivitäten für die numerische Lösung von ODE's

$$\left\{ \begin{array}{rcl} \dot{u} &=& f(t,u,p), \\ u(0) &=& u_0, \end{array} \right. \qquad \text{genauer} \qquad \left\{ \begin{array}{rcl} \dot{u}(t,p) &=& f\big(t,u(t,p),p\big), \\ u(0,p) &=& u_0 \end{array} \right. \tag{1}$$

(damit auch DAE's) mit Schrittweitensteuerung leider **nicht** funktioniert, wenn man AD vor dem Solver mit p als aktive Variablen einschaltet, und hinterher $\partial u/\partial p$ abgreift.

(Äquivalent zur Sensitivität der Lösung u bzgl. Parametern p ist das Problem der Sensitivität der Lösung u bzgl. des Anfangswertes u_0. Siehe [1].)

Argument 1

Soweit waren wir letztes Mal nicht mehr gekommen, als Sie zu Besuch waren. Machen wir doch einfach ein Gedankenexperiment, was passiert, wenn wir *von aussen* durch Finite Differenzen Parametersensitivitäten berechnen wollen. Der Einfachheit halber seien die absolute, sowie die relative Solvertoleranz beide identisch, etwa gleich $\textsc{tol} > 0$. Dann haben wir

$$\begin{aligned} \frac{1}{\delta p}\big(u_{\mathrm{TOL}}(t,p+\delta p) - u_{\mathrm{TOL}}(t,p)\big) &= \frac{1}{\delta p}\Big(\big(u(t,p+\delta p) + \mathcal{O}(\textsc{tol})\big) - \big(u(t,p) + \mathcal{O}(\textsc{tol})\big)\Big) \\ &= \frac{1}{\delta p}\big(u(t,p+\delta p) - u(t,p)\big) + \mathcal{O}(\textsc{tol}/\delta p) \\ &= \frac{\partial u}{\partial p}(t,p) + \mathcal{O}(\delta p) + \mathcal{O}(\textsc{tol}/\delta p). \end{aligned}$$

Die beiden $\mathcal{O}(\textsc{tol})$-Terme heben sich natürlich nicht weg, da die numerischen Fehler i.a. unkorreliert sind. An dem Beispiel erkennt man, was die beste Wahl für δp ist, nämlich $\delta p = c\sqrt{\textsc{tol}}$, damit hat man

$$\frac{1}{\delta p}\big(u_{\mathrm{TOL}}(t,p+\delta p) - u_{\mathrm{TOL}}(t,p)\big) = \frac{\partial u}{\partial p}(t,p) + \mathcal{O}(\sqrt{\textsc{tol}}).$$

Siehe [1]. Selbst bei recht stringenter Toleranz $\textsc{tol} = 1.0\textsc{e}^{-6}$ führt das schon zu einer grossen Parameterstörung $\delta p \approx 1.0\textsc{e}^{-3}$. Nun, bei AD, da haben wir $\delta p \to 0$, was zu $\mathcal{O}(\infty)$ führt, der GAU, siehe Beispiel harmonischer Oszillator HOSZ_AD_TOL_XX.gif.

Argument 2

Zu $\textsc{tol} > 0$ ist die Funktion

$$(t,p,u_0) \mapsto u_{\mathrm{TOL}}(t,p,u_0),$$

welche einem gegebenem Startwert u_0 und Parametersatz p die numerische Lösung zu gegebener Toleranz $\textsc{tol}$ zu einem Zeitpunkt t (gemäss (irgend-)eines Integrationsverfahrens mit Schrittweitensteuerung) zuordnet, überhaupt *nicht differenzierbar* nach p! Hier gibt es einen Haufen IF's, THEN's uns ELSE's. Siehe [1].

Argument 3

Das kleine (numerische) Rauschen der numerischen Lösung um die wahre analytische Lösung ist *extrem* sensitiv bzgl. kleiner Parameterstörungen. Siehe Beispiel harmonischer Oszillator HOSZ_rauschen_im_hinteren_bereich*.gif.
[Eigentlich ist dies kein neues Argument, sondern drückt nur 'anschaulicher' die Tatsache aus, dass die beiden $\mathcal{O}(\textsc{tol})$-Terme aus Argument 1 oben, zueinander unkorreliert sind und sich daher im Dreizeiler oben nicht wegheben.]

Ausweg 1

'Interne Differentiation'. Das erweiterte Anfangswertproblem

$$\left\{ \begin{array}{lcllcl} \dfrac{\partial}{\partial t}\Big(u(t,p)\Big) & = & f\big(t,u(t,p),p\big), & u(0,p) & = & u_0 \\ \dfrac{\partial}{\partial t}\left(\dfrac{\partial u}{\partial p}(t,p)\right) & = & \dfrac{\partial f}{\partial u}\big(t,u(t,p),p\big)\dfrac{\partial u}{\partial p}(t,p) + \dfrac{\partial f}{\partial p}\big(t,u(t,p),p\big), & \dfrac{\partial u}{\partial p}(0,p) & = & 0 \end{array} \right. \tag{2}$$

solven. Siehe [1]. D.h. die Parametersensitivitätsgleichungen simultan mitintegrieren. Das ist die sauberste Vorgehensweise. Der Solver sorgt durch seine Schrittweitensteuerung dafür, dass die Lösung $(u, \partial u/\partial p)$ bis auf TOL genau berechnet wird. Insbesondere wird auch $\partial u/\partial p$ 'kontrolliert' integriert. Die Schrittweiten, die der Solver für das erweiterte System nimmt, sind natürlich generell kleiner als die für das System (1). Die Jacobimatrizen $\partial f/\partial u$ und $\partial f/\partial p$ können natürlich am geweiligen Arbeitspunkt $u(t,p)$ durch AD bestimmt werden!

Ausweg 2

Konstante Zeitschrittweite. Keine Schrittweitensteuerung verwenden, sondern mit konstanter Schrittweite integrieren. Haben wir ausprobiert. Funktioniert wunderbar. Aber: Keine gute Idee für die Praxis...

Ausweg 3

'Step size freeze'. Siehe [1]. System (1) zu gegebener Toleranz TOL solven, sich dann die Zeitschritte

$$0 = t_0 < \ldots < t_N = T \tag{3}$$

des Solvers merken/einfrieren. Dann im folgenden die Schrittweitensteuerung des Solvers *abstellen* und den Solver mit den Zeitschritten (3) fahren. [Vorher natrlich AD aktivieren mit p als aktiven Variablen.] Die Zeitschritte sind damit *a priori* fest vorgegeben und hängen *nicht* mehr von p ab. Haben wir noch nicht ausprobiert. Müsste (*Muss*!) aber funktionieren, weil es mit konstanter (p-unabhängiger) Schrittweite auch funktioniert.

Ausweg 4

AD mit p als aktive Variablen anstellen. Im Solver jedesmal, nachdem der Solver sich einen neuen Zeitschritt Δt (ein *adouble* in ADOL-C-Sprache, ein *deriv* in ADMAT-Sprache) errechnet, wieder ein normales *double* machen. [Oder äquivalent: Die Ableitungen Δt.dot auf Null setzten.] Das haben wir auf Ihr anraten noch ausprobiert. Funktioniert! (Entspricht wahrscheinlich dem *Step-Size-Freeze*-Vorgehen.)

Ausweg 5

???

Beispiel (linearer) harmonischer Oszillator

Hier $u = (x, v)$ und $p = (r, \omega)$

$$\left\{ \begin{array}{lcl} \dot{x} & = & v \\ \dot{v} & = & -2rv - \omega^2 x \end{array} \right. , \qquad x(0) = x_0, \qquad v(0) = 0$$

Die analytische Lösung ist

$$x(t,r,\omega) = \frac{x_0}{\Omega} e^{-rt}\big(\Omega\cos(\Omega t) + r\sin(\Omega t)\big), \qquad \Omega = \sqrt{\omega^2 - r^2}$$

bekanntlich eine gedämpfte harmonische Schwingung. Daraus ergeben sich $v(t,r,\omega) = \dot{x}(t,r,\omega)$ sowie $u(t,r,\omega)$, ferner alle Parametersensitivitäten $\partial u/\partial r$ und $\partial u/\partial \omega$. Letztere klingen ebenfalls exponentiell wie die Lösung $u = (x, v)$ mit der Zeit ab.

[Siehe dazu auch den Plot HOSZ_analytische_loesung_und_sensitivitaeten.gif.]

Alle Rechnungen haben wir mit MATLAB und dessen ODE Löser ode45 durchgeführt. Dies ist eine Implementierung von DoPri5 (explizites eingebettetes Runge-Kutta-Verfahren) von Dormand und Prince, implementiert von Shampine. Wir haben ferner das AD-Tool ADMAT 2.0 im einfachen (tapeless) Vorwärtsmodus (von Coleman und Verma) verwendet, da die Zahl der Outputs (ganze Zeitreihen) viel grösser als die Zahl der Inputs (hier nur zwei Parameter r und ω) ist.

References

[1] Hairer E., Noersett S. P., Wanner G.: *Solving ordinary differential equations I. Nonstiff problems.* Springer Series in Computational Mathematics, Vol. 8, 1993.