Adjoints for Roundoff Sensitivity Analysis

David Bindel

6 Aug 2026

The Beginnings of Modern Reverse Mode

Seppo Linnainmaa of Helsinki says the idea came to him on a sunny afternoon in a Copenhagen park in 1970. He used it as a tool for **estimating the effects of arithmetic rounding errors on the results of complex expressions**.
— A. Griewank, "Who Invented the Reverse Mode of Differentiation?"

Many successors to Linnainmaa (1970/1976), includng Iri (1984), Christianson (1992), Langlois (2001), Thierry and Langlois (2009), more recent systems like ADAPT and SATIRE.

The Exact Setup

Given inputs x, intermediates t, output $y = f(x)$, computation is:

$$F(t, x) = 0, \quad y = w^T t$$

Often we choose $F(t, x) = t - G(t, x)$ where G is strictly lower triangular in t. For example: $y = -x_1 + \sqrt{x_1^2 - x_2}$ involves

$$\begin{aligned} t_1 &= G_1 = x_1^2 \\ t_2 &= G_2 = t_1 - x_2 \\ t_3 &= G_3 = \sqrt{t_2} \\ t_4 &= G_4 = t_3 - x_1 \\ y &= e_4^T t = t_4 \end{aligned}$$

The Approximate Setup

Given inputs x, intermediates $\hat{t}$, output $\hat{y} = \hat{f}(\hat{x})$, computation is:

$$F(\hat{t}, \hat{x}) = \rho, \quad \hat{y} = w^T \hat{t}$$

Often we choose $F(\hat{t}, \hat{x}) = \hat{t} - G(\hat{t}, \hat{x})$ where G is strictly lower triangular in $\hat{t}$. For example: $y = -x_1 + \sqrt{x_1^2 - x_2}$ involves

$$\begin{aligned}
\hat{t}_1 &= G_1 = \hat{x}_1^2 + \rho_1 \\
\hat{t}_2 &= G_2 = \hat{t}_1 - \hat{x}_2 + \rho_2 \\
\hat{t}_3 &= G_3 = \sqrt{\hat{t}_2} + \rho_3 \\
\hat{t}_4 &= G_4 = \hat{t}_3 - \hat{x}_1 + \rho_4 \\
\hat{y} &= e_4^T \hat{t} = \hat{t}_4
\end{aligned}$$

Same slide, except with some hats and ρ!

Bounding Intermediate Errors

Actual computation

$$F(\hat{t}, \hat{x}) = \rho, \quad \hat{y} = w^T \hat{t}$$

For explicit computation $F(t, x) = t - G(t, x)$, have elementwise bound:

$$|\rho| \leq D|t|\epsilon_{\mathrm{mach}}$$

Use D to model varying precision, Sterbenz subtraction.

For *implicit* computations, ρ incorporates both *roundoff* and *residual error* terms from solvers.

Linearization and Elimination

Consider reference computation and perturbed computation:

$$\begin{aligned} F(t, x) &= 0, & y &= w^T t, \\ F(\hat{t}, \hat{x}) &= \rho, & \hat{y} &= w^T \hat{t}. \end{aligned}$$

Linearize about (x, t) to get

$$\begin{bmatrix} F_{,t} & F_{,x} \\ w^T & 0 \end{bmatrix} \begin{bmatrix} \Delta t \\ \Delta x \end{bmatrix} = \begin{bmatrix} \rho \\ \Delta y \end{bmatrix}.$$

Eliminate Δt and write $\lambda = F_{,t}^{-T} w$ to get

$$-\lambda^T F_{,x} \Delta x = \Delta y - \lambda^T \rho.$$

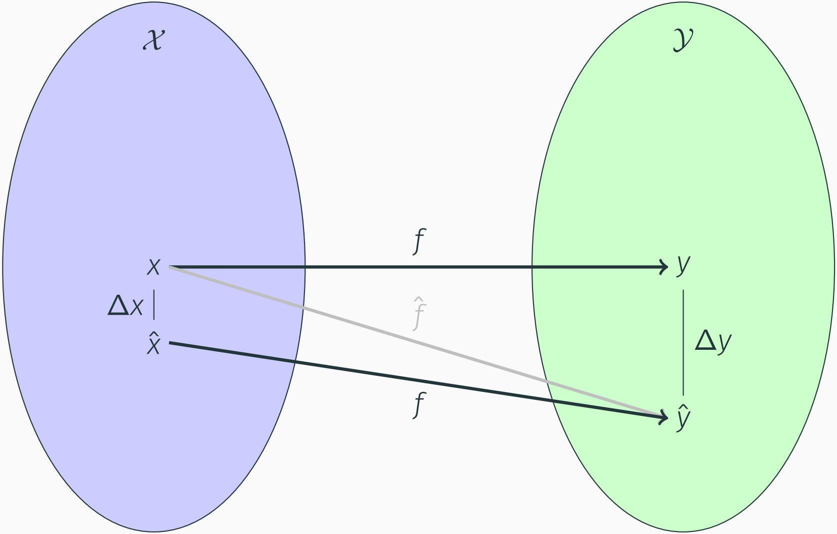
Linearized Analyses

Several consequences from the expression:

$$-\lambda^T F_{,x} \Delta x = \Delta y - \lambda^T \rho.$$

	Assume	Consequence
Reverse mode	$\rho = 0$	$\nabla_x f(x) = -(F_{,x})^T \lambda$
Forward error	$\Delta x = 0$	$\Delta y = \lambda^T \rho.$

Backward Error Picture



Linearized Analyses

Standard consequences from the expression:

$$-\lambda^T F_{,x} \Delta x = \Delta y - \lambda^T \rho.$$

	Assume	Consequence
Reverse mode	$\rho = 0$	$\nabla_x y = -(F_{,x})^T \lambda$
Forward error	$\Delta x = 0$	$\Delta y = \lambda^T \rho$
Backward error	$\Delta y = 0$	$\lambda^T F_{,x} \Delta x = -\lambda^T \rho$

Our focus: (linearized) backward error analysis picture

$$f(\hat{x}) = f(x) - \lambda^T F_{,x} = \hat{f}(x) = f(x) + \lambda^T \rho.$$

Adjoints and Backward Error

Minimal norm backward error is

$$\Delta x = (u^T)^\dagger(\lambda^T \rho), \quad u = (F_{,x})^T \lambda$$

In the scalar case, $\|\Delta x\| = |\lambda^T \rho| / \|u\|$.

Define $v = \lambda \odot Dt$ to get

$$\begin{aligned} \frac{\|\Delta y\|}{\|y\|} &\leq \frac{\|v\|_1}{\|y\|} \epsilon_{\mathrm{mach}} \\ \frac{\|\Delta x\|}{\|x\|} &\leq \frac{\|v\|_1}{\|u\| \|x\|} \epsilon_{\mathrm{mach}} \end{aligned}$$

Replace $\|v\|_1$ with $\|v\|_2$ if we model rounding errors as iid random perturbations.

Ongoing Work: Finding Problem Inputs

Measure backward stability at x via

$$\beta_p = \frac{\|v\|_p}{\|u\|\|x\|}, \quad p \in \{1, 2\}.$$

This is strictly involved in a *bound* on backward error:

$$\frac{\|\Delta x\|}{\|x\|} \leq \beta_1 \epsilon_{\mathrm{mach}}.$$

But we can compute gradients of β_p with one more forward and backward pass (and one Hessian pass)!

Now maximize $\beta_1(x)$ or $\beta_2(x)^2$ to find problems inputs.

Questions?

A few questions for all of you:

- Has the backward error approach been combined with reverse mode autodiff before? I haven't found it, but the literature is scattered...
- Similarly, has gradient-based search for bad cases (with autodiff in this style) been done before? Higham (1993) did direct search, but I've found no derivative-based approaches from citing papers.
- Does anyone have a review they like on reverse mode differentiation and error analysis?

(And, of course, I'm happy to take your questions as well!)

Backup: $\nabla_x \beta$

$$d = \begin{cases} (F_{,t})^{-1}\left(\frac{F_{,x}u}{\|u\|^2} + \frac{(Dt \odot s)}{\|v\|_1}\right), & \text{for the case } \beta_1 \\ (F_{,t})^{-1}\left(\frac{F_{,x}u}{\|u\|^2} + \frac{(Dt \odot v)}{\|v\|^2}\right), & \text{for the case } \beta_2, \end{cases} \tag{1}$$

$$g_i = -\sum_i \lambda_i F_{i,tt} d, \quad a_i = -\sum_i \lambda_i F_{i,xt} d, \tag{2}$$

$$h_i = -\sum_i \lambda_i F_{i,tx} u, \quad b_i = -\sum_i \lambda_i F_{i,xx} u \tag{3}$$

$$c = g - h + D^T \begin{cases} -(\lambda \odot s)/\|v\|_1, & \text{for the case } \beta_1 \\ (\lambda \odot v)/\|v\|^2, & \text{for the case } \beta_2 \end{cases} \tag{4}$$

$$\nabla_x \beta = a - b - (F_{,x})^T (F_{,t})^{-T} c - \frac{x}{\|x\|^2} \tag{5}$$