Pullback / vjp rule for Linear system solving

$$f(\underline{\underline{A}}, \underline{b}) = \{ \text{solve} \quad \underline{\underline{A}}\,\underline{x} = \underline{b} \quad \text{for } \underline{x} \}=: \underline{x}$$

$\underline{\underline{A}} \in \mathbb{R}^{N \times N}$
$\underline{b} \in \mathbb{R}^N$
$\underline{x} \in \mathbb{R}^N$

- Numerical solution to PDEs
- Optimization Methods
- ⋮

e.g.: direct methods (LU, cholesky ...)
iterative methods (CG, GMRES, multigrid ...)

task: backward propagate cotangent information on the output $\bar{\underline{x}} \in \mathbb{R}^N$ to the inputs $\bar{\underline{\underline{A}}} \in \mathbb{R}^{N \times N}$ & $\bar{\underline{b}} \in \mathbb{R}^N$

↳ without reverse-mode AD through the numerical solver (=unrolling)

in general

$$\bar{\underline{\underline{A}}} = \bar{\underline{x}}^T \frac{\partial f}{\partial \underline{\underline{A}}} \qquad \bar{\underline{b}}^T = \bar{\underline{x}}^T \frac{\partial f}{\partial \underline{b}}$$

$=$ ↑ hmm. this is not a nice quantity

– Index notation

primal / forward: $A_{ij}\, x_j = b_i$

now the pullbacks:

$$\bar{A}_{ik} = \bar{x}_j \frac{\partial f_j}{\partial A_{ik}} = \bar{x}_j \frac{\partial x_j}{\partial A_{ik}} \quad (2)$$

$$\bar{b}_k = \bar{x}_j \frac{\partial f_j}{\partial b_k} \quad (1)$$

(1) $\frac{\partial x_j}{\partial b_k}$ —implicit differentiation→ $\frac{\partial}{\partial b_k}(A_{ij}\, x_j) = \frac{\partial}{\partial b_k}(b_i)$

$$A_{ij} \frac{\partial x_j}{\partial b_k} = \delta_{ik}$$

$$\Leftrightarrow \quad \frac{\partial x_j}{\partial b_k} = A^{-1}_{ij}\, \delta_{ik} = A^{-1}_{kj}$$

(2) $\frac{\partial x_j}{\partial A_{ik}}$ —implicit differentiation→ $\frac{\partial}{\partial A_{ik}}(A_{ij}\, x_j) = \frac{\partial}{\partial A_{ik}}(b_i)$

$$\frac{\partial A_{ij}}{\partial A_{ik}} x_j + A_{ij} \frac{\partial x_j}{\partial A_{ik}} = 0$$

$$\delta_{ik}\, \delta_{jl}\, x_j + A_{ij} \frac{\partial x_j}{\partial A_{ik}} = 0$$

$$\Leftrightarrow \quad \frac{\partial x_j}{\partial A_{ik}} = -A^{-1}_{ij}\, \delta_{ik}\, \delta_{jl}\, x_j$$

$$\frac{\partial x_j}{\partial A_{ik}} = -A^{-1}_{kj}\, x_l$$

plug back in

(1) $\bar{b}_k = \bar{x}_j \frac{\partial x_j}{\partial b_k} = \bar{x}_j\, A^{-1}_{kj}$

$$\Leftrightarrow \quad A_{kj}\, \bar{b}_k = \bar{x}_j$$

(2) $\bar{A}_{ik} = -\bar{x}_j \underbrace{A^{-1}_{kj}}\, x_l = -\bar{b}_k\, x_l$

back to symbolic notation

$$\underline{\lambda} = \{ \text{solve} \quad \underline{\underline{A}}^T \underline{\lambda} = \bar{\underline{x}} \quad \text{for } \lambda \}$$

λ ... is the adjoint variable $\in \mathbb{R}^N$

$$\bar{\underline{b}} = \underline{\lambda}$$

$$\bar{\underline{\underline{A}}} = -\underline{\lambda}\, \underline{x}^T \quad \text{outer product}$$

Full pullback rule

$$B(f_1, (\underline{\underline{A}}, \underline{b},), (\bar{\underline{x}},)) = ((\underbrace{\underline{f}(\underline{\underline{A}}, \underline{b})}_{\underline{x}}, (\underbrace{-\underline{\lambda}\, \underline{x}^T}_{\bar{\underline{\underline{A}}}}, \underbrace{\underline{\lambda}}_{\bar{\underline{b}}},)))$$

Alternatively

this is not sparse (we need sparse outer products)

$$B(f_1, (\underline{\underline{A}}, \underline{b},), (\bar{\underline{x}},)) = ((\underbrace{\underline{\underline{A}}^{-1} \underline{b}}_{\underline{x}},), (\underbrace{-\bar{\underline{b}}\, \underline{x}^T}_{\bar{\underline{\underline{A}}}}, \underbrace{\underline{\underline{A}}^{-T} \bar{\underline{x}}}_{\bar{\underline{b}}},)))$$

requires another linear system solve, but with the transposed primal matrix, i.e.. $\underline{\underline{A}}^T \in \mathbb{R}^{N \times N}$

↳ if primal pass was done with direct solver, we could re-use its factorization, in the adjoint form

↳ or use adjoint form of preconditioner for an iterative adjoint solve