Compute:

$$f''(x_1) = \frac{1}{h^2}\Big(\underbrace{f(x_0)}_{=0} - 2f(x_1) + f(x_2)\Big)$$

$$= \frac{1}{h^2}\left(-2f_1 + f_2\right)$$

$$f''(x_2) = \frac{1}{h^2}\left(f_1 - 2f_2 + f_3\right)$$

$$f''(x_k) = \frac{1}{h^2}\left(f_{k-1} - 2f_k + f_{k+1}\right)$$

$$\vdots$$

$$f''(x_N) = \frac{1}{h^2}\left(f_{N-1} - 2f_N\right)$$

$$\begin{pmatrix} f''_1 \\ f''_2 \\ \vdots \\ \vdots \\ \\ f''_N \end{pmatrix} = \frac{1}{h^2}\begin{pmatrix} -2 & 1 & 0 & \cdots & & & 0 \\ 1 & -2 & 1 & 0 & \cdots & & 0 \\ 0 & 1 & -2 & 1 & 0 & \cdots & 0 \\ \vdots & 0 & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & & & & 1 & 0 \\ \vdots & & \cdots & 0 & 1 & -2 & 1 \\ 0 & 0 & \cdots & & 0 & 1 & -2 \end{pmatrix}\begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ \vdots \\ \\ f_N \end{pmatrix}$$

$\Rightarrow$ $\boxed{\vec{f}'' = \frac{1}{h^2} A \vec{f}}$ (matrix multiplication)

can *invert* : $\vec{f} = h^2 A^{-1} \vec{f}''$ (finite difference method for PDEs)

<u>nontrivial b.c.s</u>

$$f(x_0) = f_0 \quad , \quad f(x_{N+1}) = f_{N+1}$$

$$\rightarrow \quad f_1'' \simeq \frac{1}{h^2} (f_0 - 2 f_1 + f_2)$$

$$f_N'' \simeq \frac{1}{h^2} (f_{N-1} - 2 f_N + f_{N+1})$$

$$\Rightarrow \quad \underbrace{\begin{pmatrix} f_1'' - \frac{1}{h^2} f_0 \\ f_2'' \\ \vdots \\ f_{N-1}'' \\ f_N'' - \frac{1}{h^2} f_{N+1} \end{pmatrix}}_{= \vec{f}'' - \frac{1}{h^2} \vec{\Delta}} = h^2 A \vec{f}$$

$\vec{\Delta}$ ~ sparse vector of b.c.s

$$\Leftrightarrow \quad \vec{f} = h^2 A^{-1} \left(\vec{f}'' - \frac{1}{h^2} \vec{\Delta} \right)$$

Automatic differentiation

F.D. are inexact and <u>slow</u>

Computing $\nabla f(x_1, \dots, x_N)$ takes $\mathcal{O}(N)$ function evaluations

Symbolic methods are cumbersome / may be unavailable

~~manual differentia~~

AD yields reasonably fast, <u>exact</u> gradients / derivatives in $\mathcal{O}(1)$ function evaluations

<u>Idea</u> (Forward mode)

Introduce "dual numbers" ~~(Grassmann numbers)~~

of the form $v + \varepsilon \dot{v}$, where $\varepsilon^2 = 0$

with rules

$$\begin{cases} (v + \varepsilon \dot{v}) + (w + \varepsilon \dot{w}) = v + w + \varepsilon(\dot{v} + \dot{w}) \\ (v + \varepsilon \dot{v})(w + \varepsilon \dot{w}) = vw + \varepsilon(\dot{v} w + v \dot{w}) \\ f(v + \varepsilon \dot{v}) = f(v) + \varepsilon f'(v) \dot{v} \end{cases}$$

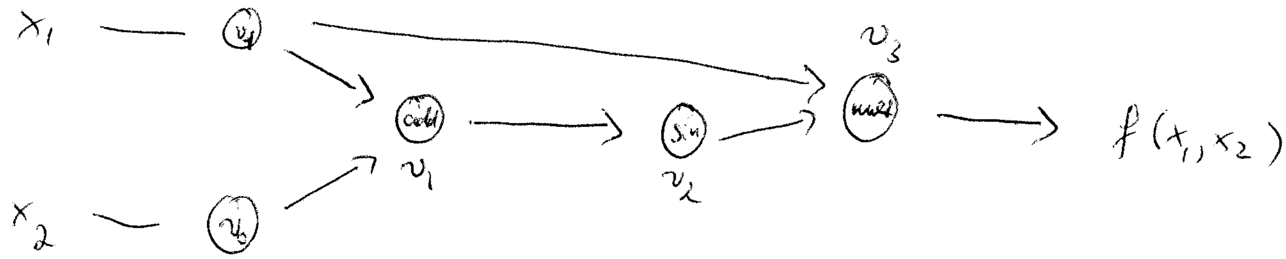
The chain rule works as expected:

$$g(f(v + \varepsilon \dot{v})) = g(f(v) + \varepsilon f'(v) \dot{v})$$

$$= g(f(v)) + \varepsilon g'(f(v)) f'(v) \dot{v}$$

Equivalent to decomposing function evaluations into computational graphs

Ex: $f(x_1, x_2) = x_1 \sin(x_1 + x_2)$



Define:

$$v_{-1} = x_1$$

$$v_0 = x_2$$

$$v_1 = v_{-1} + v_0$$

$$v_2 = \sin(v_1)$$

$$v_3 = v_{-1} \cdot v_2$$

Compute derivatives

$$\dot{v}_{-1} = 0$$

$$\dot{v}_0 = 1$$

$$\dot{v}_1 = \dot{v}_{-1} + \dot{v}_0$$

$$\dot{v}_2 = \dot{v}_1 \cos(v_1)$$

$$\dot{v}_3 = \dot{v}_{-1} v_2 + v_{-1} \dot{v}_2$$

$$\Rightarrow \frac{\partial f}{\partial x_2}$$

omputationally

- implement operators $+$, $*$, $\sin$, $\cos$, $\exp$, ..
that work on dual numbers

- evaluate compositions using

$$f(x+\varepsilon) = f(x) + \varepsilon f'(x)$$

extract prefactor of ε to find derivative in <u>one</u>

function evaluation.

ery efficient for vector ~~scalar~~ functions of one ~~many~~ variable

$$f: \mathbb{R} \longrightarrow \mathbb{R}^n$$

ften we have scalar functions of many variables

$$f: \mathbb{R}^n \longrightarrow \mathbb{R}$$

hen, use "reverse mode AD"

Forward mode is exact but still slow

for $f(x_1, \ldots, x_N)$ (need to compute $f(x_1+\varepsilon, x_2, \ldots, x_N)$,

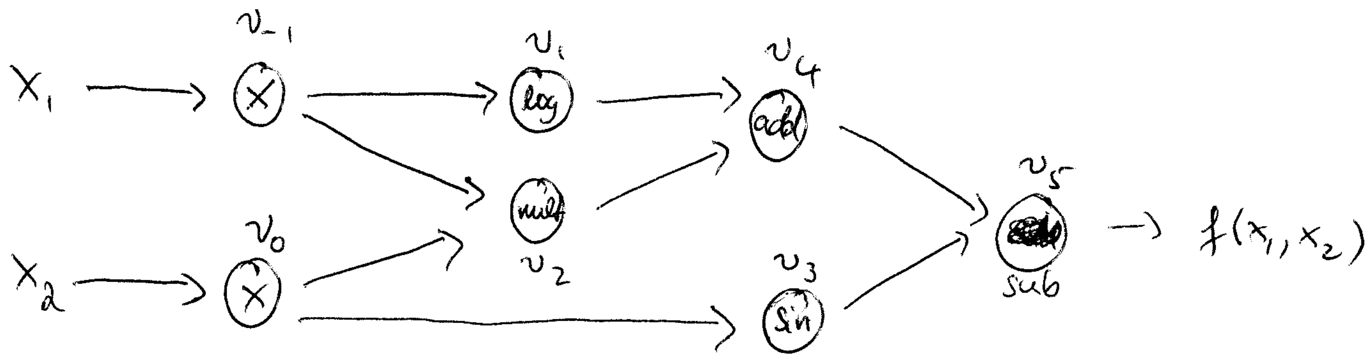
$\ldots$ $f(x_1, \ldots, x_{N-1}, x_N+\varepsilon)$

individually)

olution: Reverse-mode AD

<u>Ex</u>: $f(x_1, x_2) = \log x_1 + x_1 x_2 - \sin(x_2)$

omputational graph:



~~each node, we~~

each node, we are interested in $\frac{\partial f}{\partial v_i} = \bar{v}_i$

cause $\frac{\partial f}{\partial v_5} = \frac{\partial f}{\partial f} = 1$, this is our recursion start.

The chain rule is used to construct all derivatives in <u>one</u> backward pass.

$$\bar{v}_5 = \frac{\partial f}{\partial v_5} = \frac{\partial f}{\partial f} = 1$$

$$\bar{v}_4 = \frac{\partial f}{\partial v_4} = \frac{\partial f}{\partial v_5}\frac{\partial v_5}{\partial v_4} = \bar{v}_5 \underbrace{\frac{\partial v_5}{\partial v_4}}_{= \frac{\partial}{\partial v_4}(v_4 - v_3) = 1} = \bar{v}_5 = 1$$

$$\bar{v}_3 = \frac{\partial f}{\partial v_3} = \frac{\partial f}{\partial v_5}\frac{\partial v_5}{\partial v_3} = \bar{v}_5 \underbrace{\frac{\partial}{\partial v_3}(v_4 - v_3)}_{= -1} = -\bar{v}_5 = -1$$

$$\bar{v}_2 = \frac{\partial f}{\partial v_2} = \frac{\partial f}{\partial v_4}\frac{\partial v_4}{\partial v_2} = \bar{v}_4 \underbrace{\frac{\partial}{\partial v_2}(v_1 + v_2)}_{= 1} = \bar{v}_4 = 1$$

$$\bar{v}_1 = \frac{\partial f}{\partial v_1} = \frac{\partial f}{\partial v_4}\frac{\partial v_4}{\partial v_1} = \bar{v}_4 \frac{\partial}{\partial v_1}(v_1 + v_2) = \bar{v}_4 = 1$$

$$\bar{v}_0 = \frac{\partial f}{\partial v_0} = \frac{\partial f}{\partial v_3}\frac{\partial v_3}{\partial v_0} + \frac{\partial f}{\partial v_2}\frac{\partial v_2}{\partial v_0} = \bar{v}_3 \frac{\partial}{\partial v_0}(\sin v_0) + \bar{v}_2 \frac{\partial}{\partial v_0}(v_0 \cdot v_{-1})$$

$$= -\cos v_0 + v_{-1} = \frac{\partial f}{\partial x_2}$$

$$\bar{v}_{-1} = \frac{\partial f}{\partial v_{-1}} = \frac{\partial f}{\partial v_1}\frac{\partial v_1}{\partial v_{-1}} + \frac{\partial f}{\partial v_2}\frac{\partial v_2}{\partial v_{-1}} = \bar{v}_1 \frac{\partial}{\partial v_{-1}}(v_0 v_{-1}) + \bar{v}_2 \frac{\partial}{\partial v_{-1}}(\log v_{-1})$$

$$= v_0 + \frac{1}{v_{-1}} = \frac{\partial f}{\partial x_1}$$

→ compute <u>all</u> $\frac{\partial f}{\partial x_i}$ in one single pass ($O(1)$)

Efficient implementation is a little trickier.