Forward and Reverse Mode Automatic Differentiation

Suppose you have simple program

$$f(x, y) \coloneqq xy + \sin x$$

which you wish to differentiate.

First, transform the expression into Static Single Assignment (SSA) form, where each step is a single atomic operation whose derivative is known.

$$\boxed{\begin{array}{ll} x \coloneqq \star & \rightsquigarrow a, b \\ y \coloneqq \star & \rightsquigarrow a \\ a \coloneqq xy & \rightsquigarrow f \\ b \coloneqq \sin x & \rightsquigarrow f \\ f \coloneqq a + b & \rightsquigarrow \star \end{array}}$$

The notation $\alpha \rightsquigarrow \beta$ means the value of α is directly referenced by β below. We write $\alpha = \star$ to indicate that α is a free input, and $\alpha \rightsquigarrow \star$ if it is a final output.

Forward mode

The forward mode derivative program is formed simply by finding the differential of each step.

$$\boxed{\begin{array}{l} \mathrm{d}x \coloneqq \star \\ \mathrm{d}y \coloneqq \star \\ \mathrm{d}a \coloneqq y\,\mathrm{d}x + x\,\mathrm{d}y \\ \mathrm{d}b \coloneqq \cos x\,\mathrm{d}x \\ \mathrm{d}f \coloneqq \mathrm{d}a + \mathrm{d}b \end{array}}$$

For any given $\mathrm{d}x$ and $\mathrm{d}y$, we can directly compute $\mathrm{d}f$. For example, if $(\mathrm{d}x, \mathrm{d}y) = (1, 0)$, then $\mathrm{d}f$ evaluates to $\partial f/\partial x$. To find $\partial f/\partial y$, we need to evaluate the forward pass again, with $(\mathrm{d}x, \mathrm{d}y) = (0, 1)$.

Forward mode requires **one evaluation** of the program **per input** variable.

Implementing forward mode

In practice, functions can be evaluated in forward mode efficiently by simply interweaving the normal program statements with their differentials.

$$\boxed{\begin{array}{l} \text{function } (x, y; \mathrm{d}x, \mathrm{d}y)\ \{ \\ \qquad a \coloneqq xy \\ \qquad \mathrm{d}a \coloneqq y\,\mathrm{d}x + x\,\mathrm{d}y \\ \qquad b \coloneqq \sin x \\ \qquad \mathrm{d}b \coloneqq \cos x\,\mathrm{d}x \\ \qquad f \coloneqq a + b \\ \qquad \mathrm{d}f \coloneqq \mathrm{d}a + \mathrm{d}b \\ \} \to (f; \mathrm{d}f) \end{array}}$$

This function computes both f and the differential $\mathrm{d}f$ in one pass.

Forward mode functor

The essence of forward mode differentiation is the functor

$$\mathcal{F}\{f\}(x, \mathrm{d}x) = (f(x), \mathbb{D}f[x](\mathrm{d}x))$$

where $\mathbb{D}f[x](\mathrm{d}x)$ is the directional derivative of f at x in the $\mathrm{d}x$ direction, or

$$\mathbb{D}f[x](\mathrm{d}x) \coloneqq \lim_{\varepsilon \to 0} \frac{f(x + \varepsilon\,\mathrm{d}x) - f(x)}{\varepsilon}$$

where we assume the domain of f is a vector space.

This functor has a simple composition law

$$\mathcal{F}\{g \circ f\} = \mathcal{F}\{g\} \circ \mathcal{F}\{f\}$$
$$\mathcal{F}\{g \circ f\}(x, \mathrm{d}x) = (g(f(x)), \mathbb{D}g[f(x)](\mathbb{D}f[x](\mathrm{d}x)))$$

which makes it easy to find the derivative of larger programs.

$$\mathcal{F}\left\{\begin{array}{l} (x, y) \mapsto \{ \\ \qquad a \coloneqq xy \\ \qquad b \coloneqq \sin x \\ \qquad f \coloneqq a + b \\ \} \to (f) \end{array}\right\} = \begin{array}{l} (x, y, \mathrm{d}x, \mathrm{d}y) \mapsto \{ \\ \qquad (a, \mathrm{d}a) \coloneqq \mathcal{F}\{xy\}(x, y, \mathrm{d}x, \mathrm{d}y) = (xy, y\,\mathrm{d}x + x\,\mathrm{d}y) \\ \qquad (b, \mathrm{d}b) \coloneqq \mathcal{F}\{\sin x\}(x, \mathrm{d}x) = (\sin x, \cos x\,\mathrm{d}x) \\ \qquad (f, \mathrm{d}f) \coloneqq \mathcal{F}\{a + b\}(a, b, \mathrm{d}a, \mathrm{d}b) = (a + b, \mathrm{d}a + \mathrm{d}b) \\ \} \to (f, \mathrm{d}f) \end{array}$$

Reverse mode

The reverse mode is slightly more confusing: from the SSA form, starting from bottom to top, find the *derivative operator* $\frac{\partial}{\partial \alpha}$ for each variable $\alpha \rightsquigarrow \beta_1, \ldots, \beta_k$ using the chain rule to write it in terms of the variables it affects:

$$\frac{\partial}{\partial \alpha} = \frac{\partial \beta_1}{\partial \alpha}\frac{\partial}{\partial \beta_1} + \cdots + \frac{\partial \beta_k}{\partial \alpha}\frac{\partial}{\partial \beta_2}$$

Then, write $\partial \alpha$ in place of $\frac{\partial}{\partial \alpha}$, and view it as a real variable instead of an operator.

$$\boxed{\begin{array}{l} \partial f \coloneqq \star \\ \partial b \coloneqq \partial f \\ \partial a \coloneqq \partial f \\ \partial y \coloneqq x\,\partial a \\ \partial x \coloneqq y\,\partial a + \cos x\,\partial b \end{array}}$$

Then, to find $\frac{\partial}{\partial x} f(x, y)$, we assign $\partial f = 1$ and substitute from top to bottom to obtain ∂x and ∂y.

We set $\partial f = 1$ because, when viewed as an operator applied to the function in question, $\frac{\partial}{\partial f} f(x, y) = 1$. The final value of ∂x is then $\frac{\partial}{\partial x} f(x, y)$.

Reverse mode requires **one evaluation** of the program **per output** variable.

Implementing reverse mode

We can write the reverse mode derivative program by including both the normal steps and the derivative steps. However, this time the steps can't all be run at once because the derivative steps run in reverse order, so must be run after all the normal steps. Therefore, we put the derivative steps in a callback function $\mathbb{J}$ so they can be run later, after the current function and any following functions have been run.

$$\boxed{\begin{array}{l} \text{function } (x, y)\ \{ \\ \qquad a \coloneqq xy \\ \qquad b \coloneqq \sin x \\ \qquad f \coloneqq a + b \\ \qquad \mathbb{J} \coloneqq (\partial f) \mapsto \{ \\ \qquad\qquad \partial b \coloneqq \partial f \\ \qquad\qquad \partial a \coloneqq \partial f \\ \qquad\qquad \partial y \coloneqq x\,\partial a \\ \qquad\qquad \partial x \coloneqq y\,\partial a + \cos x\,\partial b \\ \qquad \} \to (\partial x, \partial y) \\ \} \to (f, \mathbb{J}) \end{array}}$$

Reverse mode functor

The essence of reverse mode differentiation is the functor

$$\mathcal{R}\{f\}(x) = (f(x), \mathbb{D}f[x]^*)$$

where $\partial f \mapsto \mathbb{D}f[x]^*(\partial f)$ is the adjoint operator of $\mathrm{d}x \mapsto \mathbb{D}f[x](\mathrm{d}x)$, satisfying

$$\langle \mathbb{D}f[x]^*(\partial f), \mathrm{d}x \rangle = \langle \partial f, \mathbb{D}f[x](\mathrm{d}x) \rangle$$

for some inner product $\langle\ \ ,\ \ \rangle$.

This functor has the composition law

$$\mathcal{R}\{g \circ f\}(x) = (g(f(x)), \mathbb{D}f[x]^* \circ \mathbb{D}g[f(x)]^*)$$

Graphs

Program flow can be visualised like this:

$$\begin{tikzcd} x \arrow[r] \arrow[drr] & xy = a \arrow[dr] & \\ & & a + b = f \\ y \arrow[uur] & \sin x = b \arrow[ur] & \end{tikzcd}$$

The forward mode program has the same flow:

$$\begin{tikzcd} (x, \mathrm{d}x) \arrow[r] \arrow[drr] & \mathcal{F}\{xy\} = (a, \mathrm{d}a) \arrow[dr] & \\ & & \mathcal{F}\{a + b\} = (f, \mathrm{d}f) \\ (y, \mathrm{d}y) \arrow[uur] & \mathcal{F}\{\sin x\} = (b, \mathrm{d}b) \arrow[ur] & \end{tikzcd}$$

The reverse mode program, however, propagates derivatives backwards, as if all the arrows are reversed.

Further reading

https://rufflewind.com/2016-12-30/reverse-mode-automatic-differentiation