

 **MATHEMATICS**

# Reverse mode differentiation vs. forward mode differentiation - where are the benefits?

Asked 8 years, 10 months ago Modified 1 year, 8 months ago Viewed 14k times

37

According to [Wikipedia](#) forward mode differentiation is preferred when  $f : \mathbb{R}^n \mapsto \mathbb{R}^m$ ,  $m >> n$ . I cannot see any computational benefits. Let us take simple example:  $f(x, y) = \sin(xy)$ . We can visualize it as graph with four nodes and 3 edges. Top node is  $\sin(xy)$ , node one level below is  $xy$  and two initial nodes are  $x$  and  $y$ . Derivatives on nodes are  $\cos(xy)$ ,  $x$ , and  $y$ . For both reverse and forward mode differentiation we have to compute these derivatives. How is reverse mode differentiation is computationally superior here?

 derivatives

[Share](#) [Cite](#) [Follow](#)

edited Jan 31, 2023 at 11:08

 HelloWorld  
749 3 17

asked Mar 20, 2017 at 17:56

 user1700890  
507 1 4 13

**2 Answers**

Sorted by: Highest score (default)

76

The chain rule states that to compute the Jacobian of an operation we should multiply the Jacobians of all sub-operations together. The difference between forward- and reverse-mode auto-differentiation is the *order* in which we multiply those Jacobians.

In your case you only have two sub-operations:  $xy$  and  $\sin()$ , leading to only one matrix multiplication, so it isn't really instructive. However, let's consider an operation with 3 sub-operations. Take the function:



$$\mathbf{y} = f(\mathbf{x}) = r(q(p(\mathbf{x})))$$



where  $\mathbf{x}$  and  $\mathbf{y}$  are vectors of different lengths. We can break this down into:



$$\mathbf{a} = p(\mathbf{x}), \quad \mathbf{b} = q(\mathbf{a}), \quad \mathbf{y} = r(\mathbf{b}).$$

This gives us the Jacobian

$$\underbrace{\frac{\partial \mathbf{y}}{\partial \mathbf{x}}}_{|\mathbf{y}| \times |\mathbf{x}|} = \underbrace{\frac{\partial r(\mathbf{b})}{\partial \mathbf{b}}}_{|\mathbf{y}| \times |\mathbf{b}|} \underbrace{\frac{\partial q(\mathbf{a})}{\partial \mathbf{a}}}_{|\mathbf{b}| \times |\mathbf{a}|} \underbrace{\frac{\partial p(\mathbf{x})}{\partial \mathbf{x}}}_{|\mathbf{a}| \times |\mathbf{x}|},$$

with the size of each matrix noted below the matrix. The time taken to compute each of those intermediate Jacobians is fixed, but the *order* in which we multiply them together changes the number of operations required to do so. Forward-mode auto-differentiation would compute

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \frac{\partial r(\mathbf{b})}{\partial \mathbf{b}} \left( \frac{\partial q(\mathbf{a})}{\partial \mathbf{a}} \frac{\partial p(\mathbf{x})}{\partial \mathbf{x}} \right),$$

which involves  $|\mathbf{x}| \cdot |\mathbf{a}| \cdot |\mathbf{b}| + |\mathbf{x}| \cdot |\mathbf{b}| \cdot |\mathbf{y}|$  multiplications\*, which simplifies to  $|\mathbf{x}| \cdot |\mathbf{b}| \cdot (|\mathbf{a}| + |\mathbf{y}|)$ . In contrast, reverse-mode auto-differentiation would compute

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \left( \frac{\partial r(\mathbf{b})}{\partial \mathbf{b}} \frac{\partial q(\mathbf{a})}{\partial \mathbf{a}} \right) \frac{\partial p(\mathbf{x})}{\partial \mathbf{x}},$$

which involves  $|\mathbf{y}| \cdot |\mathbf{a}| \cdot |\mathbf{b}| + |\mathbf{y}| \cdot |\mathbf{a}| \cdot |\mathbf{x}|$  multiplications, simplifying to  $|\mathbf{y}| \cdot |\mathbf{a}| \cdot (|\mathbf{b}| + |\mathbf{x}|)$ .

Assuming for simplicity that the dimensionality of variables monotonically increases or decreases through the computation, in the case that  $|\mathbf{y}| \geq |\mathbf{b}| \geq |\mathbf{a}| \geq |\mathbf{x}|$ , we can see that forward-mode auto-differentiation results in the same or fewer operations, since  $\frac{|\mathbf{y}| \cdot |\mathbf{a}|}{|\mathbf{x}| \cdot |\mathbf{b}|} \geq \frac{|\mathbf{y}| + |\mathbf{a}|}{|\mathbf{x}| + |\mathbf{b}|}$ , hence  $|\mathbf{y}| \cdot |\mathbf{a}| \cdot (|\mathbf{b}| + |\mathbf{x}|) \geq |\mathbf{x}| \cdot |\mathbf{b}| \cdot (|\mathbf{a}| + |\mathbf{y}|)$ . Similarly, if  $|\mathbf{y}| \leq |\mathbf{b}| \leq |\mathbf{a}| \leq |\mathbf{x}|$  then reverse-mode auto-differentiation results in the same or fewer operations.

This means that reverse-mode auto-differentiation (a.k.a. back propagation) will usually be faster when  $f : \mathbb{R}^n \mapsto \mathbb{R}^m, m \ll n$ , i.e. the cost function output is low dimensional (e.g. a scalar loss function), but the input is high dimensional (e.g. pixels in an image, or words in an input text), as is generally the case in neural network training.

This reasoning on whether forward- or reverse-mode is preferable extends to longer chains of Jacobians. However, exceptions can occur, e.g. when the lowest dimensionality of variables occurs neither at the function input or output, but somewhere in between. In such cases, the optimal ordering of matrix multiplications won't be fully forward- or reverse-mode, but a hybrid scheme.

I've discussed theoretical/idealized considerations, but there are practical considerations too. For example, reverse-mode auto-differentiation requires a forward pass through the code to compute values, then a reverse pass to compute the derivatives. A trace of the values needs to be stored during the forward pass, in order to compute the reverse pass. This increases the complexity of implementing and running reverse-mode auto-differentiation.

\*The number of scalar multiplications required to multiply two matrices of sizes  $a \times b$  and  $b \times c$  is  $a \cdot b \cdot c$ .

Share Cite Follow

edited May 4, 2024 at 20:53

answered Feb 19, 2019 at 18:34



user664303

982 8 11

1  $|\mathbf{x}|$  is the length of  $\mathbf{x}$ , i.e. number of elements;  $\cdot$  just means multiply. Regarding the order, yes, things inside parentheses are computed first. – user664303 Aug 14, 2020 at 19:32

1 @GuilhermeParreira I have updated the question to explain this. – user664303 Aug 28, 2020 at 20:05

1 Now I understood! Thank you!! – Guilherme Parreira Feb 23, 2021 at 21:16

1 @Tom I've improved the reasoning to demonstrate that forward mode is in general faster for monotonically increasing dimensionality, and reverse mode is faster in the opposite case. For other situations, please note my comments in the penultimate paragraph. – user664303 May 4, 2024 at 20:49

1 @Apricity Which inequality? This one?:  $(|\mathbf{y}| \cdot |\mathbf{a}|) / (|\mathbf{x}| \cdot |\mathbf{b}|) \geq (|\mathbf{y}| + |\mathbf{a}|) / (|\mathbf{x}| + |\mathbf{b}|)$ . Are you asking for a proof of that? – user664303 Jul 21, 2024 at 17:19



An analogy might help. Let  $\mathbf{A}$ ,  $\mathbf{B}$ , and  $\mathbf{C}$  be matrices with dimensions such that  $\mathbf{ABC}$  is well defined. There are two obvious ways to compute this product, represented by  $(\mathbf{AB})\mathbf{C}$  and  $\mathbf{A}(\mathbf{BC})$ . Which of those will require fewer multiplications and additions depends on the dimensions of the matrices. For example, if  $\mathbf{C}$  has width 1 then the second form will be faster, or at least no slower. It's efficient to multiply by thin or short matrices early.

13



If  $\mathbf{A}$ ,  $\mathbf{B}$ , and  $\mathbf{C}$  correspond to Jacobians of various steps in the computation graph, then I believe  $\mathbf{C}$  having width 1 corresponds to the case when  $m = 1$ .



Share Cite Follow

answered Jul 6, 2021 at 1:15



David Roodman

333 2 4

2 > It's efficient to multiply by thin or short matrices early. This finally clicks! – Rounak Datta May 16, 2023 at 9:09 

## Start asking to get answers

Find the answer to your question by asking.

[Ask question](#)

## Explore related questions

[derivatives](#)

See similar questions with these tags.