Most materials in the discussion here follow from the paper (Baydin et al., 2018)
Derivative Calculation I

- From Baydin et al. (2018) there are four types of methods
- Deriving the explicit form
  Example: consider

  \[ f(x_1, x_2) = \ln x_1 + x_1 x_2 - \sin x_2 \]

  We calculate

  \[
  \frac{\partial f(x_1, x_2)}{\partial x_1} = \frac{1}{x_1} + x_2
  \]
Derivative Calculation II

- Numerical way by finite difference
  \[ f(x + h) - f(x) \]
  \[ h \]
  with a small \( h \)
- Symbolic way: using tools to get an explicit form
- Automatic differentiation (AD): topic of this set of slides
- Back-propagation is a special case of automatic differentiation
So you can roughly guess that in automatic differentiation, chain rules are repeated applied.
Consider the function

\[ f(x_1, x_2) = \ln x_1 + x_1 x_2 - \sin x_2 \]

Forward mode to compute the function value

\[
\begin{align*}
\nu_{-1} &= x_1 = 2 \\
\nu_0 &= x_2 = 5 \\
\nu_1 &= \ln \nu_{-1} = \ln 2 \\
\nu_2 &= \nu_{-1} \times \nu_0 = 2 \times 5 \\
\nu_3 &= \sin \nu_0 = \sin 5 \\
\nu_4 &= \nu_1 + \nu_2 = 0.693 + 10 \\
\nu_5 &= \nu_4 - \nu_3 = 10.693 + 0.959 \\
y &= \nu_5 = 11.652
\end{align*}
\]
Forward Mode of AD II

- See also the computational graph

\[ f(x_1, x_2) \]
Each $v_i$ comes from a simple operation

For computing

\[ \frac{\partial f}{\partial x_1} \]

we let

\[ \dot{v}_i = \frac{\partial v_i}{\partial x_1} \]

and apply the chain rule

Forward derivative calculation:
Forward Mode of AD IV

\[
\begin{align*}
\dot{v}_1 &= \dot{x}_1 &= 1 \\
\dot{v}_0 &= \dot{x}_2 &= 0 \\
\dot{v}_1 &= \dot{v}_{-1}/v_{-1} &= 1/2 \\
\dot{v}_2 &= \dot{v}_{-1} \times v_0 + \dot{v}_0 \times v_{-1} &= 1 \times 5 + 0 \times 2 \\
\dot{v}_3 &= \dot{v}_0 \times \cos v_0 &= 0 \times \cos 5 \\
\dot{v}_4 &= \dot{v}_1 + \dot{v}_2 &= 0.5 + 5 \\
\dot{v}_5 &= \dot{v}_4 - \dot{v}_3 &= 5.5 - 0 \\
\dot{y} &= \dot{v}_5 &= 5.5
\end{align*}
\]
For example,

\[ v_1 = \ln v_{-1} \]

\[
\frac{\partial v_1}{\partial x_1} = \frac{1}{v_{-1}} \times \frac{\partial v_{-1}}{\partial x_1}
\]

\[ = \frac{\dot{v}_{-1}}{v_{-1}} \]
Consider

\[ f : \mathbb{R}^n \rightarrow \mathbb{R}^m \]

so that

\[
\begin{bmatrix}
y_1 \\
\vdots \\
y_m
\end{bmatrix}
= f(x_1, \ldots, x_n)
\]

The Jacobian is

\[
\begin{bmatrix}
\frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_1}{\partial x_n} \\
\vdots & \ddots & \vdots \\
\frac{\partial y_m}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_n}
\end{bmatrix}
\]
If we initialize

$$\dot{x} = \left[0, \ldots, 0, 1, 0, \ldots, 0\right]^T$$

then

$$\begin{bmatrix}
\frac{\partial y_1}{\partial x_i} \\
\vdots \\
\frac{\partial y_m}{\partial x_i}
\end{bmatrix}$$

can be calculated in one forward pass.
But this means we need $n$ forward passes for the whole Jacobian.

In many optimization methods we do not need the whole Jacobian. Instead we need Jacobian-vector products.

That is,

$$
\mathbf{Jr} = \begin{bmatrix}
\frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_1}{\partial x_n} \\
\vdots & \ddots & \vdots \\
\frac{\partial y_m}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_n}
\end{bmatrix}
\begin{bmatrix}
r_1 \\ \\
\vdots \\ \\
r_n
\end{bmatrix}
$$
This can be calculated in one pass by initializing with $\dot{x} = r$

Now $\dot{v}_i$ is changed from

$$\dot{v}_i = \frac{\partial v_i}{\partial x_1}$$

to

$$\dot{v}_i = \frac{\partial v_i}{\partial x_1} r_1 + \cdots + \frac{\partial v_i}{\partial x_n} r_n$$
Jacobian Calculation by Forward Mode V

For example, if

\[ v_2 = v_{-1} \times v_0, \]

then we still have

\[ \dot{v}_2 = \dot{v}_{-1} \times v_0 + \dot{v}_0 \times v_{-1} \]

We will see examples of using Jacobian-vector products later in discussing Newton methods.
The discussion shows that the forward mode is efficient for
\[ f : \mathbb{R} \rightarrow \mathbb{R}^m \]
by one pass.

But for the other extreme
\[ f : \mathbb{R}^n \rightarrow \mathbb{R}, \]
to calculate the gradient
\[ \nabla f = \left[ \frac{\partial y}{\partial x_1} \ldots \frac{\partial y}{\partial x_n} \right]^T \]
we need \( n \) passes.
This is not efficient

Subsequently we will consider another way for AD: reverse mode
Cheng-Hung Liu helped to draw the computational graph and prepare the tables.