SYSTEMS DEVELOPMENT FOR COMPUTATIONAL SCIENCE

LECTURE 12

Fabian Wermelinger

Harvard University

CS107 / AC207

Tuesday, October 11th 2022

LAST TIME

- Example of Newton's method and numerical schemes to approximate derivatives.
- Evaluation trace
- The computational graph
- Computing derivatives of one variable using the forward mode

TODAY

Main topics: ***Automatic differentiation: forward mode, higher dimensions, dual numbers***

Details:

- Beyond the basics:
  - Computing derivatives in higher dimensions using the forward mode
  - The Jacobian in forward mode
  - What the forward mode actually computes
- Review of complex numbers
- Dual numbers in forward mode AD

FORWARD MODE AD: RECAP

Recall from last time: forward mode AD

- Break down a complicated function $f(x)$ into *elementary operations*. These include addition, multiplication, sign negation, transcendental functions, trigonometric functions, etc.
- Breaking down the problem into smaller pieces led us to define *intermediate results* v_j which are functions that *depend* on the *independent* variables x (the input).
- We defined a new differential operator for a *directional derivative* and called it $D_p(\cdot)$. The *chain rule* applies to this operator in the same way it does for d/dt for example.
- ***Forward mode AD:*** evaluate v_j and $D_p v_j$ *simultaneously* as you evaluate the elementary pieces of a complicated $f(x)$ *from the inside out*.

FORWARD MODE AD: RECAP

Notation I: the vector $p \in \mathbb{R}^m$ is called the ***seed vector***. We have introduced it when we defined our directional derivative:

$$D_p y_i \stackrel{\text{def}}{=} (\nabla y_i)^\top p = \sum_{j=1}^{m} \frac{\partial y_i}{\partial x_j} p_j .$$

This definition is just a weighted sum (inner product) of derivatives with respect to the independent variables. The "direction" is given by the seed vector p.

The seed vector allows us to *cherry-pick a certain derivative of interest* (choose a "direction"). If we were interested in $\partial y_i / \partial x_1$ we would choose $p_1 = 1$ and $p_k = 0 \ \forall k \neq 1$. We can even choose a weighted combination of derivatives $\partial y_i / \partial x_j$ if we needed to.

We are free to choose the seed vector p.

FORWARD MODE AD: RECAP

Notation II: in the literature you may come across the notation $\dot{v}_j$ to denote the directional derivative of v_j, instead of the notation $D_p v_j$ that we have used here. In physics, the "dot" notation refers to differentiation with respect to time, which can only advance in one direction. Our direction is given by the m-dimensional vector p for which the notation $D_p v_j$ is more expressive.
Read it as: *derivative of v_j in direction of p*.

→ of course you are free to choose whichever notation you are most comfortable with.

FORWARD MODE AD: HIGHER DIMENSIONS

So far we have been looking at a scalar function $f(x)$ with a single argument $x \in \mathbb{R}$. In the following slides we extend our discussion to:

- Multivariate scalar function $f(x) : \mathbb{R}^m \mapsto \mathbb{R}$
- Multivariate vector function $f(x) : \mathbb{R}^m \mapsto \mathbb{R}^n$

The math we covered up to here remains exactly the same, what changes is the number of inputs and outputs in the computational graph.

FORWARD MODE AD: HIGHER DIMENSIONS

We start by looking at the case $f(x):\mathbb{R}^m \mapsto \mathbb{R}$, where $x \in \mathbb{R}^m$.

- We deal with more than one *input* $x = [x_1, x_2, \ldots, x_m]^\mathsf{T}$.
- This means we have m *independent* variables. If you recall the table for the primal and tangent traces, we have m gray rows instead of just one. Similarly, the computational graph will have m input nodes on the left side.
- The direction $p \in \mathbb{R}^m$ has m components too.
- ***Examples for such functions:*** temperature field in 3D, Ray-tracing example in 3D, Neural networks, optimization problems and so on.

FORWARD MODE AD: HIGHER DIMENSIONS

Example: 2-Dimensional input $x \in \mathbb{R}^m$ with $m = 2$

Consider the independent coordinates $x = [x_1, x_2]^\top$ with

$$f(x) = x_1 x_2$$

It is easy to compute the gradient right away:

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \end{bmatrix} = \begin{bmatrix} x_2 \\ x_1 \end{bmatrix}$$

FORWARD MODE AD: HIGHER DIMENSIONS

Example: 2-Dimensional input $x \in \mathbb{R}^m$ with $m = 2$

- The primal trace for this function consists of simply *one intermediate variable:*

$$f(x) = v_1 = v_{-1} v_0 = x_1 x_2$$

- The tangent trace requires the computation of $D_p v_1$, but now we have *more than just one coordinate for the direction* $\rightarrow p = [p_1, p_2]^\intercal$:

$$D_p v_1 = (\nabla v_1)^\intercal p = \frac{\partial v_1}{\partial x_1} p_1 + \frac{\partial v_1}{\partial x_2} p_2 = x_2 p_1 + x_1 p_2$$

- ***How to choose*** p ***if you are interested in*** $\frac{\partial f}{\partial x_1}$***?*** $\rightarrow p = [1, 0]^\intercal$
- ***How to choose*** p ***if you are interested in*** $\frac{\partial f}{\partial x_2}$***?*** $\rightarrow p = [0, 1]^\intercal$

FORWARD MODE AD: HIGHER DIMENSIONS

Example: 2-Dimensional input $x \in \mathbb{R}^m$ with $m = 2$

Let's now add a sinusoidal on top:

$$f(x) = \sin(x_1 x_2)$$

- The primal trace consists of *two intermediate variables*:

$$v_1 = v_{-1} v_0 = x_1 x_2$$

$$f(x) = v_2 = \sin(v_1)$$

- From the previous slide you know that:

$$D_p v_1 = x_2 p_1 + x_1 p_2$$

Discuss with your neighbors the following items:

1. What is the computational graph for this problem?
2. What is the value of $D_p v_2$?

FORWARD MODE AD: HIGHER DIMENSIONS

We can see that what the ***forward mode AD*** really computes is:

$$\nabla f \cdot p$$

The most general form is a ***vector function*** $f(x) : \mathbb{R}^m \mapsto \mathbb{R}^n$. The gradient of such function ∇f is an *outer product* that is called the ***Jacobian***. Think of the Jacobian as a matrix, where the elements of that matrix are given by the first order partial derivatives $\frac{\partial f_i}{\partial x_j}$.

Forward mode AD computes the inner product of the Jacobian with the seed vector p

$$J \cdot p,$$

where $J \in \mathbb{R}^{n \times m}$ and $p \in \mathbb{R}^m$. Think of this as projecting the Jacobian in the direction given by p.

FORWARD MODE AD: HIGHER DIMENSIONS

Example for $f(x) : \mathbb{R}^m \mapsto \mathbb{R}^n$ with $m = 2$ and $n = 2$:

Let $x = [x_1, x_2]^\intercal$. Consider the vector function given by:

$$f(x) = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = \begin{bmatrix} x_1 x_2 + \sin(x_1) \\ x_1 + x_2 + \sin(x_1 x_2) \end{bmatrix}$$

The gradient (Jacobian) for this function is not hard to compute:

$$\nabla f = J = \begin{bmatrix} \partial f_1 / \partial x_1 & \partial f_1 / \partial x_2 \\ \partial f_2 / \partial x_1 & \partial f_2 / \partial x_2 \end{bmatrix}$$

$$= \begin{bmatrix} x_2 + \cos(x_1) & x_1 \\ 1 + x_2 \cos(x_1 x_2) & 1 + x_1 \cos(x_1 x_2) \end{bmatrix}$$

FORWARD MODE AD: HIGHER DIMENSIONS

Example for $f(x) : \mathbb{R}^m \mapsto \mathbb{R}^n$ with $m = 2$ and $n = 2$:

Let $x = [x_1, x_2]^\intercal$. Consider the vector function given by:

$$f(x) = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = \begin{bmatrix} x_1 x_2 + \sin(x_1) \\ x_1 + x_2 + \sin(x_1 x_2) \end{bmatrix}$$

What is the computational graph for this problem?

FORWARD MODE AD: HIGHER DIMENSIONS

Example for $f(x):\mathbb{R}^m \mapsto \mathbb{R}^n$ ***with*** $m=2$ ***and*** $n=2$***:***

In the following we aim to compute the directional derivative $D_p v_5 = D_p f_1$ which is the directional derivative of the *first component* f_1 of our vector function $f(x)$.

- By drawing the computational graph we should have found that $v_5 = v_1 + v_2$. (Note that a computational graph can be drawn in multiple ways.)
- Taking the derivative of v_5:

$$D_p v_5 = (\nabla v_5)^\intercal p = \underbrace{\left(\frac{\partial v_5}{\partial v_1}\nabla v_1 + \frac{\partial v_5}{\partial v_2}\nabla v_2\right)}_{\text{chain rule}}^\intercal p$$

$$= (\nabla v_1 + \nabla v_2)^\intercal p = D_p v_1 + D_p v_2$$

FORWARD MODE AD: HIGHER DIMENSIONS

Example for $f(x) : \mathbb{R}^m \mapsto \mathbb{R}^n$ with $m = 2$ and $n = 2$:

- Taking the derivative of v_5:

$$D_p v_5 = (\nabla v_1 + \nabla v_2)^\intercal p = D_p v_1 + D_p v_2$$

- We need $D_p v_1$ and $D_p v_2$ → computational graph $v_1 = v_{-1} v_0$:

$$D_p v_1 = D_p(v_{-1} v_0) = \underbrace{v_0 D_p v_{-1} + v_{-1} D_p v_0}_{\text{product rule}}$$

- But we know $v_{-1} = x_1$ and $v_0 = x_2$:

$$D_p v_{-1} = (\nabla v_{-1})^\intercal p = \begin{bmatrix} \frac{\partial x_1}{\partial x_1} \\ \frac{\partial x_1}{\partial x_2} \end{bmatrix}^\intercal p = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \end{bmatrix} = p_1$$

FORWARD MODE AD: HIGHER DIMENSIONS

Example for $f(x) : \mathbb{R}^m \mapsto \mathbb{R}^n$ with $m = 2$ and $n = 2$:

- Taking the derivative of v_5:

$$D_p v_5 = (\nabla v_1 + \nabla v_2)^\top p = D_p v_1 + D_p v_2$$

- We need $D_p v_1$ and $D_p v_2$ → computational graph $v_1 = v_{-1} v_0$:

$$D_p v_1 = D_p(v_{-1} v_0) = \underbrace{v_0 D_p v_{-1} + v_{-1} D_p v_0}_{\text{product rule}}$$

- If you do the same math for $D_p v_0$ you find:

$$\boxed{D_p v_1 = x_2 p_1 + x_1 p_2}$$

FORWARD MODE AD: HIGHER DIMENSIONS

***Example for** $f(x) : \mathbb{R}^m \mapsto \mathbb{R}^n$ **with** $m = 2$ **and** $n = 2$:*

- Taking the derivative of v_5:

$$D_p v_5 = (\nabla v_1 + \nabla v_2)^\intercal p = D_p v_1 + D_p v_2$$

- We find $D_p v_2$ with $v_2 = \sin(v_{-1})$ by following the *exact same procedure*. By doing so we find the solution:

$$\boxed{D_p v_5 = D_p f_1 = \big(x_2 + \cos(x_1)\big) p_1 + x_1 p_2}$$

- For the second component f_2 of the vector function the procedure is again identical. For our computational graph:

$$\boxed{D_p v_6 = D_p f_2 = \big(1 + x_2 \cos(x_1 x_2)\big) p_1 + \big(1 + x_1 \cos(x_1 x_2)\big) p_2}$$

FORWARD MODE AD: HIGHER DIMENSIONS

Example for $f(x) : \mathbb{R}^m \mapsto \mathbb{R}^n$ with $m = 2$ and $n = 2$:

- Component f_1:

$$\boxed{D_p v_5 = D_p f_1 = \left(x_2 + \cos(x_1)\right) p_1 + x_1 p_2}$$

- Component f_2:

$$\boxed{D_p v_6 = D_p f_2 = \left(1 + x_2 \cos(x_1 x_2)\right) p_1 + \left(1 + x_1 \cos(x_1 x_2)\right) p_2}$$

- Analytical solution from earlier slide:

$$\nabla f = J = \begin{bmatrix} \partial f_1 / \partial x_1 & \partial f_1 / \partial x_2 \\ \partial f_2 / \partial x_1 & \partial f_2 / \partial x_2 \end{bmatrix}$$

$$= \begin{bmatrix} x_2 + \cos(x_1) & x_1 \\ 1 + x_2 \cos(x_1 x_2) & 1 + x_1 \cos(x_1 x_2) \end{bmatrix}$$

FORWARD MODE AD: HIGHER DIMENSIONS

Example for $f(x) : \mathbb{R}^m \mapsto \mathbb{R}^n$ ***with*** $m = 2$ ***and*** $n = 2$**:**

We find the following:

- If we choose $p = [1, 0]^\intercal$ (the ***unit vector*** for coordinate x_1) then $D_p v_5 = \frac{\partial f_1}{\partial x_1}$ and $D_p v_6 = \frac{\partial f_2}{\partial x_1}$, exactly the *first column of J*.
- If we choose $p = [0, 1]^\intercal$ (the ***unit vector*** for coordinate x_2) then $D_p v_5 = \frac{\partial f_1}{\partial x_2}$ and $D_p v_6 = \frac{\partial f_2}{\partial x_2}$, exactly the *second column of J*.

Take-home message: We can compute the full Jacobian in forward mode AD using *m passes*, where seed vectors p are set to the m-th unit vector along coordinate x_m for the m-th pass.

COMPLEX NUMBERS

A complex number has the form:

$$z = x + \mathrm{i}y$$

with x the *real* part and y the *imaginary* part.

- The *imaginary unit* i gives the complex number $z \in \mathbb{C}$ the special property that *defines* the square root of a ***negative*** number

$$\boxed{\mathrm{i} = \sqrt{-1},}$$

  such that $\mathrm{i}^2 = -1$.
- This rather simple definition allows to solve a whole new dimension of problems → things that cannot be explained with real numbers in $\mathbb{R}$ can now be expressed with complex numbers in $\mathbb{C}$.

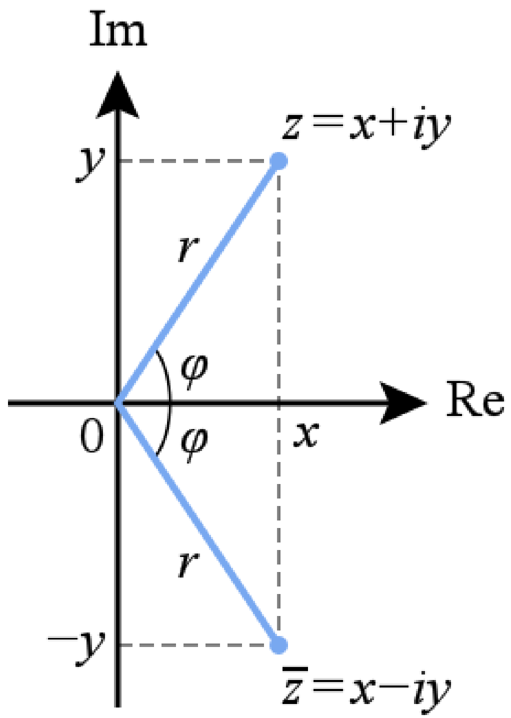
COMPLEX NUMBERS

A complex number has the form:

$$z = x + \mathrm{i}y$$

with x the *real* part and y the *imaginary* part.

- You can think of z as a *two-dimensional* vector.
- The imaginary unit i *extends the real line* with an orthogonal *imaginary* axis.



COMPLEX NUMBERS

Complex numbers have several useful properties:

- *Complex conjugate*: $z = x + \mathrm{i}y \rightarrow z^* = x - \mathrm{i}y$
- *Magnitude*: $|z|^2 = zz^* = (x + \mathrm{i}y)(x - \mathrm{i}y) = x^2 + y^2$
- *Polar form*: $z = re^{\mathrm{i}\varphi}$
  - r: is called *radius*, $r = |z|$
  - φ: is called *angle*, $\varphi = \arctan(y/x)$
- When you compute the product

  $$z = z_1 z_2$$

  what happens to the radius and angle of z?
- Using the polar form of z, can you see why $zz^* \in \mathbb{R}$ is a real number?

TOWARDS DUAL NUMBERS

- A dual number is similar to a complex number.
- The unit ϵ that gives the dual number its special properties is defined different than the imaginary unit i for a complex number.

A dual number consists of a *real* part and a *dual* part and is written as

$$z = a + b\epsilon,$$

where $a, b \in \mathbb{R}$ and ϵ is a special (nilpotent) number with the property

$$\epsilon^2 = 0$$

and $\epsilon \neq 0 \rightarrow$ ***Note:*** *ϵ is not a real number.*

DUAL NUMBERS

Dual numbers have several useful properties:

- ***Dual conjugate***: $z = a + b\epsilon \rightarrow z^* = a - b\epsilon$
- ***Magnitude***: $|z|^2 = zz^* = (a + b\epsilon)(a - b\epsilon) = a^2$
- ***Polar decomposition***: $z = a(1 + m\epsilon) = ae^{m\epsilon}$
  - where $m = \frac{b}{a}$ for $a \neq 0$
  - $e^{m\epsilon} = 1 + m\epsilon + \frac{1}{2}(m\epsilon)^2 + \ldots = 1 + m\epsilon$

More interesting: dual numbers have the following properties for addition and multiplication:

$$z_1 + z_2 = (a_1 + b_1\epsilon) + (a_2 + b_2\epsilon) = (a_1 + a_2) + (b_1 + b_2)\epsilon$$

$$z_1 z_2 = (a_1 + b_1\epsilon)(a_2 + b_2\epsilon) = (a_1 a_2) + (a_1 b_2 + a_2 b_1)\epsilon$$

DUAL NUMBERS

$$z_1 + z_2 = (a_1 + b_1\epsilon) + (a_2 + b_2\epsilon) = (a_1 + a_2) + (b_1 + b_2)\epsilon$$
$$z_1 z_2 = (a_1 + b_1\epsilon)(a_2 + b_2\epsilon) = (a_1 a_2) + (a_1 b_2 + a_2 b_1)\epsilon$$

- Let $f(x)$ and $g(x)$ be two functions with $f'(x)$ and $g'(x)$ their *derivative* with respect to x.
- Substitute the real part $a_1 = f$ and dual part $b_1 = f'$ in z_1 and $a_2 = g$ and $b_2 = g'$ in z_2:

$$z_1 + z_2 = (f + f'\epsilon) + (g + g'\epsilon) = (f + g) + (f' + g')\epsilon$$
$$z_1 z_2 = (f + f'\epsilon)(g + g'\epsilon) = (fg) + (fg' + f'g)\epsilon$$

DUAL NUMBERS

- Substitute the real part $a_1 = f$ and dual part $b_1 = f'$ in z_1 and $a_2 = g$ and $b_2 = g'$ in z_2:

$$z_1 + z_2 = (f + f'\epsilon) + (g + g'\epsilon) = (f + g) + (f' + g')\epsilon$$
$$z_1 z_2 = (f + f'\epsilon)(g + g'\epsilon) = (fg) + (fg' + f'g)\epsilon$$

- ***Observe:***

1. ***Adding dual numbers together resembles the linearity of addition and results in adding the functions in the real part and adding the derivatives in the dual part.***
2. ***Multiplication results in multiplication of the functions in the real part and the product rule for the derivatives in the dual part.***

DUAL NUMBERS

- Think of f and g as *intermediate results* v_i and v_j in the primal trace of forward mode AD.
- Then their derivatives f' and g' correspond to the tangent trace $D_p v_i$ and $D_p v_j$, respectively.
- In forward mode AD, we always evaluate v_j and $D_p v_j$ ***simultaneously*** (in each row of the trace table we compute both). *We carry the primal trace and tangent trace forward as a **pair***.
- A dual number is therefore a very useful structure in forward mode AD. It can be used to encode the ***primal trace in the real part*** and the ***tangent trace in the dual part***:

$$z_j = v_j + D_p v_j \epsilon$$

DUAL NUMBERS

Example: evaluate a node v_3 in a computational graph

- Assume the relationship for v_3 is as follows. The notation we have used so far to express this in the evaluation trace table is:

$$v_3 = v_1 v_2$$

- We now write this relationship with ***dual numbers***

$$z_3 = z_1 z_2,$$

where $z_1 = v_1 + D_p v_1 \epsilon$ and $z_2 = v_2 + D_p v_2 \epsilon$.

- From the dual number product rule on the previous slide:

$$z_3 = \underbrace{v_1 v_2}_{\text{primal trace}} + \underbrace{(v_2 D_p v_1 + v_1 D_p v_2)}_{\text{tangent trace}} \epsilon,$$

DUAL NUMBERS

- Up to here, dual numbers have most of the properties we require in forward mode automatic differentiation.
- The last requirement we have on dual numbers to be useful for AD is that we can use them with the ***chain rule***.
- We must prove that the chain rule holds for a function $f(z)$, where $z = a + b\epsilon$ is now a *dual number*. *The tool we use for this is a Taylor series expansion.*
- We can expand any analytic function $f(z)$, where $z = a + b\epsilon$ is a dual number, using a Taylor series expansion

$$f(z) = \sum_{\kappa=0}^{\infty} \frac{f^{(\kappa)}(\zeta)}{\kappa!}(z-\zeta)^{\kappa}.$$

- Let $\zeta = a + 0\epsilon$ be the dual number around which we want to expand the series for $f(z)$. The point ζ is *arbitrary and we choose it to be in the close* ***neighborhood*** *of z*. The real part of ζ is the same as for z and ϵ *is small.*

DUAL NUMBERS

- We can expand any analytic function $f(z)$, where $z = a + b\epsilon$ is a dual number, using a Taylor series expansion

$$f(z) = \sum_{\kappa=0}^{\infty} \frac{f^{(\kappa)}(\zeta)}{\kappa!}(z-\zeta)^{\kappa}.$$

- Let $\zeta = a + 0\epsilon$ be the dual number around which we want to expand the series for $f(z)$. The point ζ is *arbitrary and we choose it to be in the close* ***neighborhood*** *of z*. The real part of ζ is the same as for z and ϵ *is small.*
- Expanding the series we find:

$$f(z) = f(a+b\epsilon) = \sum_{\kappa=0}^{\infty} \frac{f^{(\kappa)}(\zeta)}{\kappa!}(z-\zeta)^{\kappa} = \sum_{\kappa=0}^{\infty} \frac{f^{(\kappa)}(a)}{\kappa!}(b\epsilon)^{\kappa}$$

$$= f(a) + f'(a)b\epsilon$$

DUAL NUMBERS

- Expanding the series we find:

$$f(z) = f(a + b\epsilon) = \sum_{\kappa=0}^{\infty} \frac{f^{(\kappa)}(\zeta)}{\kappa!}(z - \zeta)^{\kappa} = \sum_{\kappa=0}^{\infty} \frac{f^{(\kappa)}(a)}{\kappa!}(b\epsilon)^{\kappa}$$

$$= f(a) + f'(a)b\epsilon$$

- ***All higher order terms vanish because of the definition*** $\epsilon^2 = 0$
- Use the dual number $z_j = v_j + D_p v_j \epsilon$ as we did before and substitute:

$$f(z_j) = f(v_j) + f'(v_j) D_p v_j \epsilon$$

$\rightarrow$ if we apply an analytic function f on the dual number z_j, the result is another dual number with:
  - Real part equal to f applied to real part of z_j
  - Dual part equal to the dual part of z_j, scaled with f' applied to real part of z_j ***(chain rule!)***

DUAL NUMBERS

- Use the dual number $z_j = v_j + D_p v_j \epsilon$ as we did before and substitute:

$$f(z_j) = f(v_j) + f'(v_j) D_p v_j \epsilon$$

- ***Recall:*** last lecture we were studying the forward primal and tangent traces of $f(x) = x - \exp\left(-2\left(\sin(4x)\right)^2\right)$. The first two *intermediate variables* are shown again below:

Forward primal trace	Forward tangent trace	Numerical value: v_j; $D_p v_j$
$v_0 = x_1 = \frac{\pi}{16}$	$D_p v_0 = 1$	1.963495e-01; 1.000000e+00
$v_1 = 4v_0$	$D_p v_1 = 4 D_p v_0$	7.853982e-01; 4.000000e+00
$v_2 = \sin(v_1)$	$D_p v_2 = \cos(v_1) D_p v_1$	7.071068e-01; 2.828427e+00

- Let $z_1 = v_1 + D_p v_1 \epsilon$ and $z_2 = \sin(z_1)$. Using the rule above, we find:

$$z_2 = \sin(z_1) = \underbrace{\sin(v_1)}_{v_2} + \underbrace{\cos(v_1) D_p v_1}_{D_p v_2} \epsilon$$

DUAL NUMBERS: EXERCISE

Given the function: $f(x) = \frac{\sin(x)}{(\cos(x))^2+1}$

Perform the following tasks:

1. Draw the computational graph for $f(x)$. The last intermediate variable is $v_5 = f(x_1)$, where x_1 is the point where we evaluate f.
2. Show that $D_p v_5$ takes the form

$$D_p v_5 = \frac{1}{v_4^2}(v_4 D_p v_1 - v_1 D_p v_4)$$

   for $v_5 = g(v_1, v_4)$ with g some function (***hint*** → chain rule).
3. Compute the last intermediate state with dual numbers $z_5 = g(z_1, z_4)$. Note that the function g is the same as in item 2 above, we just use dual numbers this time. Depending on how you draw the graph, the arguments to g may have different subscripts.

RECAP

Automatic Differentiation: Forward Mode

- Beyond the basics:
  - Computing derivatives in higher dimensions using the forward mode
  - The Jacobian in forward mode
  - What the forward mode actually computes
- Review of complex numbers
- Introduction of dual numbers

Further reading:

- P. H.W. Hoffmann, *A Hitchhiker's Guide to Automatic Differentiation*, Springer 2015, doi:10.1007/s11075-015-0067-6 (You can access this paper through the Harvard network or find it in the class repository)
- Griewank, A. and Walther, A., *Evaluating derivatives: principles and techniques of algorithmic differentiation*, SIAM 2008, Vol. 105
- Nocedal, J. and Wright, S., *Numerical Optimization*, Springer 2006, 2nd Edition