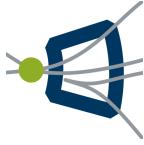


Data Analysis Block Course: Statistics and Machine learning



Exercise: Forward and Reverse Automatic Differentiation by Hand

Prof. Dr. Mikhail Mikhasenko

Assistants: Hendric Jonas, Christoph Tönnis



25 September 2026

[Indico schedule](#)

[RUB Moodle](#)

Learning goals

After this exercise, you should be able to

- explain what a **dual number** is and why dual numbers are useful for **forward-mode** automatic differentiation,
- propagate values and derivatives through a computation using dual numbers,
- compute **directional derivatives** and **Jacobian** columns with forward mode,
- explain the idea of a **computational graph** and an **adjoint** in **reverse-mode automatic** differentiation,
- propagate adjoints backwards through a computation graph,
- compute gradients and Jacobian rows with reverse mode,
- explain the practical difference between forward and reverse AD.

The calculations in this sheet are designed to be done **by hand**. No programming is required.

Part I - Forward-mode AD with dual numbers

1. What are dual numbers?

Automatic differentiation (AD) computes derivatives by applying the chain rule to the elementary operations of a calculation. For a composition, the chain rule gives

$$(f \circ q)'(x) = f'(q(x))q'(x) \quad (1)$$

A **dual number** $\hat{x}$ has the form

$$\hat{x} = x + \dot{x}\varepsilon \quad (2)$$

where $x, \dot{x} \in \mathbb{R}$ and ε is a formal symbol with the special property

$$\varepsilon^2 = 0 \quad \varepsilon \neq 0 \quad (3)$$

The ordinary value x is called the **primal value**. The coefficient $\dot{x}$ is the **tangent**: it tracks the derivative along a chosen input direction. The dot is notation for this derivative component (not necessarily a time derivative). Choosing the initial tangent is called setting an **input seed**. **Forward mode** carries primal values and tangents together from inputs to outputs.

If we then evaluate a differentiable function f using its value and first derivative for each elementary operation together with $\varepsilon^2 = 0$, we obtain

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

For the special choice $\dot{x} = 1$ we obtain

$$f(x + \varepsilon) = f(x) + f'(x)\varepsilon \quad (4)$$

so the coefficient of ε is directly $f'(x)$.

2. Arithmetic with dual numbers

Let

$$\hat{a} = a + \dot{a}\varepsilon \quad \hat{b} = b + \dot{b}\varepsilon \quad (5)$$

be two dual numbers. We define addition and multiplication rules

- **Addition:**

$$\hat{a} + \hat{b} = (a + b) + (\dot{a} + \dot{b})\varepsilon \quad (6)$$

- **Multiplication:**

$$\hat{a}\hat{b} = (a + \dot{a}\varepsilon)(b + \dot{b}\varepsilon) \quad (7)$$

Expanding the product gives

$$ab + (a\dot{b} + \dot{a}b)\varepsilon \quad (8)$$

which is exactly the product rule in differentiation.

Some useful elementary functions

Here $\log$ denotes the natural logarithm with base e .

$$\begin{aligned} \exp(a + \dot{a}\varepsilon) &= e^a + \dot{a} e^a \varepsilon \\ \sin(a + \dot{a}\varepsilon) &= \sin(a) + \dot{a} \cos(a) \varepsilon \\ \log(a + \dot{a}\varepsilon) &= \log(a) + \frac{\dot{a}}{a} \varepsilon \quad a > 0 \\ (a + \dot{a}\varepsilon)^2 &= a^2 + 2a \dot{a} \varepsilon \end{aligned} \quad (9)$$

3. Very simple 1D example

Consider a function

$$f(x) = x^2 + 3x$$

and compute $f(2)$ and $f'(2)$ using dual numbers. For this we choose the dual number

$$\hat{x}_0 = 2 + \varepsilon$$

with tangent $\dot{x}_0 = 1$.

We evaluate the parts of the function $f(x)$ separately

$$\begin{aligned}\hat{x}_0^2 &= (2 + \varepsilon)^2 = 4 + 4\varepsilon \\ 3\hat{x}_0 &= 6 + 3\varepsilon\end{aligned}$$

Therefore the whole function $f(\hat{x}_0)$ is

$$f(\hat{x}_0) = 10 + 7\varepsilon$$

Hence

$$f(2) = 10 \quad f'(2) = 7$$

The important point is that the value and the derivative were propagated at the same time.

4. More than one input

Suppose now that we have a function of dimension m that takes as its argument a vector with n components

$$g : \mathbb{R}^n \rightarrow \mathbb{R}^m \tag{10}$$

We can seed every input with its own tangent component:

$$\hat{\mathbf{x}} = \mathbf{x} + \dot{\mathbf{x}}\varepsilon \tag{11}$$

Forward AD then gives

$$g(\mathbf{x} + \dot{\mathbf{x}}\varepsilon) = g(\mathbf{x}) + J_g(\mathbf{x}) \dot{\mathbf{x}}\varepsilon \tag{12}$$

The **Jacobian matrix** $J_g(\mathbf{x})$ collects all first partial derivatives: its entry in row i and column j is $\partial g_i / \partial x_j$. A **partial derivative** measures the change with respect to one input while holding the others fixed. For n inputs and m outputs, the Jacobian has m rows and n columns.

A **sweep** is one traversal of the calculation in the chosen direction. One forward-mode sweep computes one **Jacobian-vector product (JVP)**

$$J_g(\mathbf{x})\dot{\mathbf{x}} \tag{13}$$

A **standard basis vector** has one entry equal to 1 and all other entries equal to 0. If we choose $\dot{\mathbf{x}}$ to be a standard basis vector, for example

$$\mathbf{e}_1 = (1, 0, 0)^T$$

then the tangent part is the first column of the Jacobian.

For a scalar-valued function, the **gradient** ∇g is the column vector of all partial derivatives. The **directional derivative** along an input direction $\mathbf{v}$ is

$$D_{\mathbf{v}}g(\mathbf{x}) = \left. \frac{d}{dt}g(\mathbf{x} + t\mathbf{v}) \right|_{t=0} = \nabla g(\mathbf{x})^T \mathbf{v} \quad (14)$$

Here $\mathbf{v}$ need not have unit length: scaling the seed scales the directional derivative.

Forward-mode exercises

Exercise 1.1 - Scalar input, scalar output

Consider the function

$$f(x) = \log(xe^x + 3) \quad (15)$$

Evaluate it and its derivative at $x = 1$ using dual-number propagation.

Exercise 1.2 - Vector input, scalar output

Consider now a function that takes a 3-dimensional vector as input

$$g(x, y, z) = xy + \sin z + y^2 \quad (16)$$

at the point $(x, y, z) = (1, 2, 0)$

1. First calculate the ordinary function value $g(1, 2, 0)$.
2. Use forward-mode AD with the three seeds

$$\mathbf{e}_x = (1, 0, 0) \quad \mathbf{e}_y = (0, 1, 0) \quad \mathbf{e}_z = (0, 0, 1)$$

to determine the full gradient $\nabla g(1, 2, 0)$.

3. Now use only one forward sweep with seed direction

$$\dot{\mathbf{x}} = (1, -1, 2)^T$$

to compute the directional derivative $D_{\dot{\mathbf{x}}}g = \nabla g^T \dot{\mathbf{x}}$.

Exercise 1.3 - Vector input, vector output

Consider the vector-valued function

$$\mathbf{h}(x, y, z) = \begin{pmatrix} h_1(x, y, z) \\ h_2(x, y, z) \end{pmatrix} = \begin{pmatrix} xy + z \\ x^2 + \sin y - z^2 \end{pmatrix}$$

Evaluate it at $(x, y, z) = (1, 0, 1)$

1. Calculate $\mathbf{h}(1, 0, 1)$.

2. Use the three input basis directions

$$\mathbf{e}_x \quad \mathbf{e}_y \quad \mathbf{e}_z$$

and dual numbers to compute the full Jacobian

$$J_{\mathbf{h}} = \begin{pmatrix} \frac{\partial h_1}{\partial x} & \frac{\partial h_1}{\partial y} & \frac{\partial h_1}{\partial z} \\ \frac{\partial h_2}{\partial x} & \frac{\partial h_2}{\partial y} & \frac{\partial h_2}{\partial z} \end{pmatrix}$$

Remember: each forward sweep gives one column of the Jacobian.

3. Use a single forward sweep with

$$\mathbf{v} = (1, 2, -1)^T$$

to compute $J_{\mathbf{h}}\mathbf{v}$

Part II - Reverse-mode AD / Backward AD

1. The basic idea

Forward mode propagates derivatives together with the values from input to output.

A **computational graph** represents a calculation as nodes for inputs and intermediate results, with directed edges showing which values each operation uses. An **intermediate variable** stores the result of one of these operations. A **local derivative** is the derivative of an operation's output with respect to one of its inputs.

Reverse mode proceeds as follows:

1. Perform an ordinary forward pass and store the intermediate values.
2. Start from the output.
3. Propagate sensitivities backwards through the computational graph.

For an intermediate variable v , define its **adjoint** as

$$\bar{v} = \frac{\partial L}{\partial v} \tag{17}$$

where L is the final scalar output whose derivative we want. The adjoint measures the sensitivity of this output to v . The bar notation is common in reverse-mode AD.

Initialize all adjoints to zero. The **output seed** specifies the starting adjoint at the output. For the derivative of the scalar output itself, set

$$\bar{L} = \frac{\partial L}{\partial L} = 1 \tag{18}$$

Then apply the chain rule locally, one operation at a time, in reverse order.

2. Local backward rules

Suppose an intermediate variable c is computed from earlier variables. The local backward rules are

- **Addition:** For $c = a + b$ we obtain

$$\bar{a} += \bar{c} \quad \bar{b} += \bar{c} \quad (19)$$

Accumulation, written $\bar{a} += \bar{c}$, means replacing $\bar{a}$ by its current value plus $\bar{c}$. If one variable influences the output through several paths, all contributions to its adjoint must be added.

- **Multiplication:** For $c = ab$ we obtain

$$\bar{a} += \bar{c}b \quad \bar{b} += \bar{c}a \quad (20)$$

Some useful elementary functions

$$\begin{aligned} c = e^a &\Rightarrow \bar{a} += \bar{c}e^a \\ c = \sin a &\Rightarrow \bar{a} += \bar{c}\cos a \\ c = \log a &\Rightarrow \bar{a} += \bar{c}\frac{1}{a} \\ c = a^2 &\Rightarrow \bar{a} += \bar{c}2a \end{aligned} \quad (21)$$

3. Very simple 1D reverse-mode example

Consider the function

$$f(x) = (x + 1)^2$$

and compute its derivative at $x = 2$. For this we break the function into elementary operations

$$\begin{aligned} v_1 &= x + 1 \\ v_2 &= v_1^2 \\ f &= v_2 \end{aligned}$$

Forward pass

At $x = 2$ we obtain the intermediate values

$$v_1 = 3 \quad v_2 = 9$$

Backward pass

Start with the output seed

$$\bar{v}_2 = 1$$

because $f = v_2$, so $\partial f / \partial v_2 = 1$.

Since $v_2 = v_1^2$ we get

$$\bar{v}_1 = \bar{v}_2 2v_1 = 6$$

and since $v_1 = x + 1$ we obtain

$$\bar{x} = \bar{v}_1 = 6$$

Hence

$$f'(2) = 6$$

In one dimension this may look more complicated than ordinary differentiation. Its advantage becomes clear when a function has many inputs but only one scalar output.

4. Reverse mode for several inputs

For a scalar-valued function

$$f : \mathbb{R}^n \rightarrow \mathbb{R} \tag{22}$$

one reverse sweep gives the full gradient ∇f , with derivatives with respect to all inputs at once. A **loss** measures the error of a model's predictions. In machine learning it is usually a scalar, even when the model has many parameters.

For a vector-valued function

$$\mathbf{f} : \mathbb{R}^n \rightarrow \mathbb{R}^m \tag{23}$$

we must first choose an output seed $\mathbf{w} \in \mathbb{R}^m$. Reverse mode then computes

$$J_{\mathbf{f}}^T \mathbf{w} \tag{24}$$

The **vector-Jacobian product (VJP)** is $\mathbf{w}^T J_{\mathbf{f}}$. With column-vector notation, we write its transpose $J_{\mathbf{f}}^T \mathbf{w}$. The output seed $\mathbf{w}$ assigns a starting adjoint to each output; equivalently, we differentiate the scalar $L = \mathbf{w}^T \mathbf{f}$.

To reconstruct the complete Jacobian, use one reverse sweep per output basis vector.

Reverse-mode exercises

Use exactly the same three functions as in the forward-mode section.

Exercise 2.1 - Scalar input, scalar output

Consider the function

$$f(x) = \log(xe^x + 3) \tag{25}$$

and evaluate its derivative at $x = 1$ using reverse-mode AD.

1. Write the function as a sequence of elementary intermediate variables.
2. Perform the forward pass and record all intermediate values.

3. Initialize all adjoints to zero, then set the output adjoint equal to 1.
4. Propagate all adjoints backwards.
5. Determine $\bar{x} = f'(1)$.

Exercise 2.2 - Three inputs, scalar output

Consider the function

$$g(x, y, z) = xy + \sin z + y^2 \quad (26)$$

at the point $(x, y, z) = (1, 2, 0)$. Use the intermediate variables

$$\begin{aligned} a &= xy \\ b &= \sin z \\ c &= y^2 \\ d &= a + b + c \end{aligned}$$

The scalar output is $g = d$.

1. Perform the forward pass and record a, b, c, d .
2. Initialize all adjoints to zero, then set $\bar{d} = 1$ and propagate backwards.
3. Determine the gradient $\nabla g(1, 2, 0)$ and compare it with Exercise 1.2.
4. Explain why the adjoint of y receives two contributions.

Exercise 2.3 - Three inputs, two outputs

Consider the vector-valued function

$$\mathbf{h}(x, y, z) = \begin{pmatrix} xy + z \\ x^2 + \sin y - z^2 \end{pmatrix}$$

at the point $(x, y, z) = (1, 0, 1)$. Reverse mode needs an output seed because the output is not scalar.

1. **First output:** Use the output seed

$$\mathbf{w}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

to propagate backwards only from h_1 and compute $J_{\mathbf{h}}^T \mathbf{w}_1$.

2. **Second output:** Use the output seed

$$\mathbf{w}_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

to compute $J_{\mathbf{h}}^T \mathbf{w}_2$.

3. **Full Jacobian:** Use the two results to reconstruct the complete Jacobian.

4. **General output seed:** Without calculating the Jacobian from scratch, use

$$\mathbf{w} = \begin{pmatrix} 3 \\ -1 \end{pmatrix}$$

to calculate $J_{\mathbf{h}}^T \mathbf{w}$.

Short conceptual questions

1. Why does setting $\varepsilon^2 = 0$ cause the coefficient of ε to behave like a derivative?

2. Consider a function

$$f : \mathbb{R}^{100} \rightarrow \mathbb{R}$$

How many basis-direction forward sweeps would be needed to obtain the full gradient?
How many reverse sweeps?

3. Consider a function

$$f : \mathbb{R} \rightarrow \mathbb{R}^{100}$$

Which mode would naturally be more efficient for constructing the full Jacobian?

4. What is an adjoint $\bar{v}$?

5. Why do reverse-mode updates use accumulation such as

$$\bar{x} += \dots$$

instead of simply assigning one value to $\bar{x}$?

Key concepts

Concept	Meaning in this exercise
Automatic differentiation (AD)	Computing derivatives by applying the chain rule.
Loss	A scalar measure of how poorly a model fits its target.
Dual number	$x + \dot{x}\varepsilon$, where $\varepsilon \neq 0$ and $\varepsilon^2 = 0$.
Input seed	The initial tangent or input direction for a forward sweep.
Directional derivative	Rate of change along $\mathbf{x} + t\mathbf{v}$; for a scalar output, $\nabla g^T \mathbf{v}$.
Forward mode / JVP	Propagates values and tangents to compute $J\mathbf{v}$.
Sweep / pass	One traversal of the calculation in a given direction.
Computational graph	Nodes and directed dependencies representing a calculation.
Adjoint	$\bar{v} = \partial L / \partial v$: sensitivity of the chosen scalar output L to v .
Output seed	Initial output adjoints; 1 for a scalar output, or weights $\mathbf{w}$ for several outputs.
Reverse mode / VJP	Propagates adjoints backwards to compute $J^T \mathbf{w}$, the transpose of $\mathbf{w}^T J$.
Accumulation	Adding all contributions to an adjoint when a variable affects the output through several paths.