Collapsing Taylor Mode Automatic Differentiation

Felix Dangel*, Tim Siebert*, Marius Zeinhofer, Andrea Walther

MLAftermath

March 10, 2026

Automatic Differentiation

- AD exploits chain rule and derivatives of "simple" functions to compute derivatives of "complex" functions:

$$(g \circ h)' = g'(h)h'$$

- The two main methods are Reverse Mode (aka Backpropagation) and Forward Mode.

Taylor Mode Automatic Differentiation

Higher-order derivatives can be computed via nested first-order AD or Taylor Mode AD.

- We undestand the input x as smooth curve $x : \mathbb{R} \to \mathbb{R}^n$ and define

$$x_0 := x(t_0)$$
$$x_k := \frac{d^k}{dt^k} x(t) \Big|_{t_0}$$

- Goal: Compute $\frac{d^K}{dt^K} f(x(t)) \Big|_{t_0}$

Taylor Mode AD

- $f_0 = f(x_0)$

Taylor Mode AD

- $f_0 = f(x_0)$
- $f_1 = f'(x_0)x_1$

Taylor Mode AD

- $f_0 = f(x_0)$
- $f_1 = f'(x_0)x_1$
- $f_2 = f''(x_0)x_1^2 + f'(x_0)x_2$

Taylor Mode AD

- $f_0 = f(x_0)$
- $f_1 = f'(x_0)x_1$
- $f_2 = f''(x_0)x_1^2 + f'(x_0)x_2$
- $f_3 = f'''(x_0)x_1^3 + 3f''(x_0)x_1x_2 + f'(x_0)x_3$

Taylor Mode AD

- $f_0 = f(x_0)$
- $f_1 = f'(x_0)x_1$
- $f_2 = f''(x_0)x_1^2 + f'(x_0)x_2$
- $f_3 = f'''(x_0)x_1^3 + 3f''(x_0)x_1x_2 + f'(x_0)x_3$

  $\vdots$
- $f_K = \sum_{\pi \in \mathsf{part}(K)} \nu(\pi) f^{(|\pi|)}(x_0)(\otimes_{i \in \pi} x_i)$

Taylor Mode AD

For $f = (g \circ h)(x)$ we have the following propagation:

$$\begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_K \end{pmatrix} \to \begin{pmatrix} h_0 = h(x_0) \\ h_1 = h'(x_0)(x_1) \\ h_2 = h''(x_0)(x_1, x_1) + h'(x_0)(x_2) \\ \vdots \\ h_K = \sum_{\pi \in \mathsf{part}(K)} \nu(\pi) h^{(|\pi|)}(x_0)(\otimes_{i \in \pi} x_i) \end{pmatrix}$$

$$\to \begin{pmatrix} g_0 = g(h_0) \\ g_1 = g'(h_0)(h_1) \\ g_2 = g''(h_0)(h_1, h_1) + g'(h_0)(h_2) \\ \vdots \\ g_K = \sum_{\pi \in \mathsf{part}(K)} \nu(\pi) g^{(|\pi|)}(h_0)(\otimes_{i \in \pi} h_i) \end{pmatrix} = \begin{pmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_K \end{pmatrix}$$

Higher-Order Derivatives

- Higher-order derivatives are required in the training of Physics informated networks (PINN).
- Goal of a PINN is to find weights θ such that the neural network f_θ fullfils a PDE

$$\begin{cases} \Delta f_\theta(x) = g(x) & x \in \Omega \\ f_\theta(x) \quad = 0 & x \in \partial\Omega \end{cases}$$

- To train a PINN the following loss is used

$$L(\theta) = \frac{1}{N} \sum_{i=1}^{N} \left(\Delta f_\theta(x_i) - g(x_i)\right)^2$$

Higher-Order Derivatives

We consider higher-order derivatives occuring in structured operators:

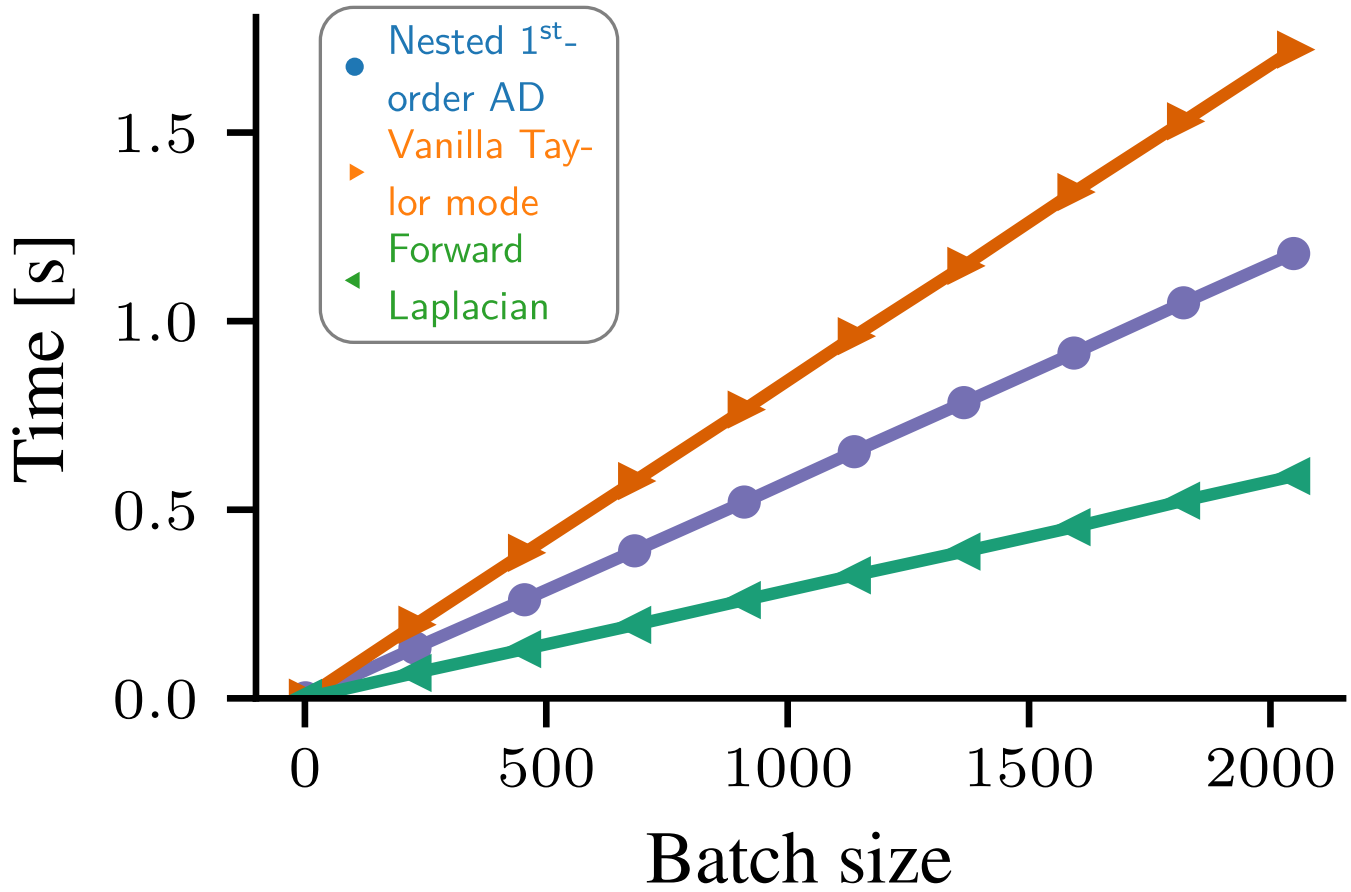
- Laplacian: $\Delta f(x) = \sum_{r=1}^{n} \frac{\partial^2 f(x)}{\partial x_r^2} = \sum_{r=1}^{n} f''(x)(e_r, e_r)$
- Biharmonic Operator:
$$\Delta^2 f(x) = \sum_{r_1=1}^{n} \sum_{r_2=1}^{n} \frac{\partial^4 f(x)}{\partial x_{r_1}^2 \partial x_{r_2}^2} = \sum_{r_1=1}^{n} \sum_{r_2=1}^{n} f^{(4)}(x)(e_{r_1}, e_{r_1}, e_{r_2}, e_{r_2})$$

Forward Laplacian

- Li et al. ([1]) found a more efficient propagation schemes to compute the Laplacian of $y(x) = (h_L \circ \cdots \circ h_1)(x)$.
- Leveraging the structure of the operator was key:

$$\begin{bmatrix} \mathbf{x} \\ \nabla \mathbf{x} \\ \Delta \mathbf{x} \end{bmatrix} \to \cdots \to \begin{bmatrix} \mathbf{h}^l \\ \nabla \mathbf{h}^l \\ \Delta \mathbf{h}^l \end{bmatrix} \to \cdots \to \begin{bmatrix} y \\ \nabla y \\ \Delta y \end{bmatrix} \quad ([1])$$

Forward Laplacian



([2])

Forward Laplacian

Can we generalize this?

Setting

Given a function $f : \mathbb{R}^n \to \mathbb{R}^m$ that is decomposable into $f = g \circ h$.

- Goal: Compute a differential operator of the form

$$\sum_{r=1}^{R} f^{(K)}(x)(v_{1_r}, \ldots, v_{K_r}).$$

- Via: Structure exploitation of the operator and Taylor mode.
- Example: Laplacian ($R = n$, $K = 2$, $v_{i_r} = e_r$)

Faà di Bruno Formula

To this end, we understand the input x as smooth curve $x : \mathbb{R} \to \mathbb{R}^n$ and define

$$x_0 := x(t_0)$$
$$x_k := \frac{d^k}{dt^k} x(t)\Big|_{t_0}.$$

The Faà di Bruno ([3]) formula of $(g \circ h)(x(t))$ is given by

$$\frac{d^K}{dt^K}(g \circ h)(x)(t)\Big|_{t_0} = \sum_{\pi \in \mathsf{part}(K)} \nu(\pi) g^{(|\pi|)}(h_0)(\otimes_{i \in \pi} h_i).$$

Taylor Mode AD

$$\begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_K \end{pmatrix} \to \begin{pmatrix} h_0 = h(x_0) \\ h_1 = h'(x_0)(x_1) \\ h_2 = h''(x_0)(x_1, x_1) + h'(x_0)(x_2) \\ \vdots \\ h_K = \sum_{\pi \in \mathsf{part}(K)} \nu(\pi) h^{(|\pi|)}(x_0)(\otimes_{i \in \pi} x_i) \end{pmatrix}$$

$$\to \begin{pmatrix} g_0 = g(h_0) \\ g_1 = g'(h_0)(h_1) \\ g_2 = g''(h_0)(h_1, h_1) + g'(h_0)(h_2) \\ \vdots \\ g_K = \sum_{\pi \in \mathsf{part}(K)} \nu(\pi) g^{(|\pi|)}(h_0)(\otimes_{i \in \pi} h_i) \end{pmatrix} = \begin{pmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_K \end{pmatrix}$$

Taylor Mode AD

$$\begin{pmatrix} x_0 \\ x_{1,r} \\ x_{2,r} \end{pmatrix} \rightarrow \begin{pmatrix} h_0 = h(x_0) \\ h_{1,r} = h'(x_0)(x_{1,r}) \\ h_{2,r} = h''(x_0)(x_{1,r}, x_{1,r}) + h'(x_0)(x_{2,r}) \end{pmatrix}$$

$$\rightarrow \begin{pmatrix} g_0 = g(h_0) \\ g_{1,r} = g'(h_0)(h_{1,r}) \\ g_{2,r} = g''(h_0)(h_{1,r}, h_{1,r}) + g'(h_0)(h_{2,r}) \end{pmatrix} = \begin{pmatrix} f_0 \\ f_{1,r} \\ f_{2,r} \end{pmatrix}$$

Taylor Mode AD

$$\begin{pmatrix} x_0 \\ x_{1,r} \\ x_{2,r} \end{pmatrix} \to \begin{pmatrix} h_0 = h(x_0) \\ h_{1,r} = h'(x_0)(x_{1,r}) \\ h_{2,r} = h''(x_0)(x_{1,r}, x_{1,r}) + h'(x_0)(x_{2,r}) \end{pmatrix}$$

$$\to \begin{pmatrix} g_0 = g(h_0) \\ g_{1,r} = g'(h_0)(h_{1,r}) \\ g_{2,r} = g''(h_0)(h_{1,r}, h_{1,r}) + g'(h_0)(h_{2,r}) \end{pmatrix} = \begin{pmatrix} f_0 \\ f_{1,r} \\ f_{2,r} \end{pmatrix}$$

$$\xrightarrow[\text{sum}]{} \sum_{r=1}^{n} f_{2,r}$$

Taylor Mode AD

$$\begin{pmatrix} x_0 \\ x_{1,r} \\ x_{2,r} \end{pmatrix} \rightarrow \begin{pmatrix} h_0 = h(x_0) \\ h_{1,r} = h'(x_0)(x_{1,r}) \\ h_{2,r} = h''(x_0)(x_{1,r}, x_{1,r}) + h'(x_0)(x_{2,r}) \end{pmatrix}$$

$$\rightarrow \begin{pmatrix} g_0 = g(h_0) \\ g_{1,r} = g'(h_0)(h_{1,r}) \\ g_{2,r} = g''(h_0)(h_{1,r}, h_{1,r}) + g'(h_0)(h_{2,r}) \end{pmatrix} = \begin{pmatrix} f_0 \\ f_{1,r} \\ f_{2,r} \end{pmatrix}$$

$$\underset{\text{sum}}{\rightarrow} \sum_{r=1}^{n} f_{2,r} \underset{\{x_{1,r}=e_r, x_{2,r}=0\}}{=} \sum_{r=1}^{n} f''(x_0)(e_r, e_r)$$

Taylor Mode AD

$$\begin{pmatrix} x_0 \\ x_{1,r} \\ x_{2,r} \\ \vdots \\ x_{K,r} \end{pmatrix} \to \begin{pmatrix} h_0 = h(x_0) \\ h_{1,r} = h'(x_0)(x_{1,r}) \\ h_{2,r} = h''(x_0)(x_{1,r}, x_{1,r}) + h'(x_0)(x_{2,r}) \\ \vdots \\ h_{K,r} = \sum_{\pi \in \mathsf{part}(K)} \nu(\pi) h^{(|\pi|)}(x_0)(\otimes_{i \in \pi} x_{i,r}) \end{pmatrix}$$

$$\to \begin{pmatrix} g_0 = g(h_0) \\ g_{1,r} = g'(h_0)(h_{1,r}) \\ g_{2,r} = g''(h_0)(h_{1,r}, h_{1,r}) + g'(h_0)(h_{2,r}) \\ \vdots \\ g_{K,r} = \sum_{\pi \in \mathsf{part}(K)} \nu(\pi) g^{(|\pi|)}(h_0)(\otimes_{i \in \pi} h_{i,r}) \end{pmatrix} = \begin{pmatrix} f_0 \\ f_{1,r} \\ f_{2,r} \\ \vdots \\ f_{K,r} \end{pmatrix} \to \cdots \to \sum_r f_{K,r}$$

Exploit Structure

$$\sum_{r=1}^{R} f_{K,r} = \sum_{r=1}^{R} g_{K,r}$$

Exploit Structure

$$\sum_{r=1}^{R} f_{K,r} = \sum_{r=1}^{R} g_{K,r} = \sum_{r=1}^{R} \sum_{\pi \in \mathsf{part}(K)} \nu(\pi) g^{(|\pi|)}(h_0)(\otimes_{i \in \pi} h_{i,r})$$

Exploit Structure

$$\sum_{r=1}^{R} f_{K,r} = \sum_{r=1}^{R} g_{K,r} = \sum_{r=1}^{R} \sum_{\pi \in \text{part}(K)} \nu(\pi) g^{(|\pi|)}(h_0)(\otimes_{i \in \pi} h_{i,r})$$

The set $\text{part}(K)$ contains $\tilde{\pi} := \{K\}$.

Exploit Structure

$$\sum_{r=1}^{R} f_{K,r} = \sum_{r=1}^{R} g_{K,r} = \sum_{r=1}^{R} \sum_{\pi \in \mathsf{part}(K)} \nu(\pi) g^{(|\pi|)}(h_0)(\otimes_{i \in \pi} h_{i,r})$$

The set $\mathsf{part}(K)$ contains $\tilde{\pi} := \{K\}$. It contributes the following term to the sum

$$g'(h_0)(h_{K,r}).$$

Exploit Structure

$$\sum_{r=1}^{R} f_{K,r} = \sum_{r=1}^{R} g_{K,r} = \sum_{r=1}^{R} \sum_{\pi \in \text{part}(K)} \nu(\pi) g^{(|\pi|)}(h_0)(\otimes_{i \in \pi} h_{i,r})$$

The set $\text{part}(K)$ contains $\tilde{\pi} := \{K\}$. It contributes the following term to the sum

$$g'(h_0)(h_{K,r}).$$

- The highest coefficient occurs only in the last term.

Exploit Structure

Then we have

$$\left(\sum_{r=1}^{R} \sum_{\pi \in \mathsf{part}(K) \setminus \{\tilde{\pi}\}} \nu(\pi) g^{(|\pi|)}(h_0)(\otimes_{i \in \pi} h_{i,r})\right) + \sum_{r=1}^{R} g'(h_0)(h_{K,r}).$$

Exploit Structure

Then we have

$$\left(\sum_{r=1}^{R} \sum_{\pi \in \text{part}(K) \setminus \{\tilde{\pi}\}} \nu(\pi) g^{(|\pi|)}(h_0)(\otimes_{i \in \pi} h_{i,r})\right) + \sum_{r=1}^{R} g'(h_0)(h_{K,r}).$$

Changing the order of summation gives

$$\left(\sum_{r=1}^{R} \sum_{\pi \in \text{part}(K) \setminus \{\tilde{\pi}\}} \nu(\pi) g^{(|\pi|)}(h_0)(\otimes_{i \in \pi} h_{i,r})\right) + g'(h_0)\left(\sum_{r=1}^{R} h_{K,r}\right).$$

Exploit Structure

Then we have

$$\left(\sum_{r=1}^{R} \sum_{\pi \in \text{part}(K) \setminus \{\tilde{\pi}\}} \nu(\pi) g^{(|\pi|)}(h_0)(\otimes_{i \in \pi} h_{i,r})\right) + \sum_{r=1}^{R} g'(h_0)(h_{K,r}).$$

Changing the order of summation gives

$$\left(\sum_{r=1}^{R} \sum_{\pi \in \text{part}(K) \setminus \{\tilde{\pi}\}} \nu(\pi) g^{(|\pi|)}(h_0)(\otimes_{i \in \pi} h_{i,r})\right) + g'(h_0)\left(\sum_{r=1}^{R} h_{K,r}\right).$$

- Sum the highest coefficient first, then propagate!

Collapsed Taylor Mode

$$\begin{pmatrix} x_0 \\ x_{1,r} \\ x_{2,r} \\ \vdots \\ x_{K,r} \end{pmatrix} \to \begin{pmatrix} h_0 = h(x_0) \\ h_{1,r} = h'(x_0)(x_{1,r}) \\ h_{2,r} = h''(x_0)(x_{1,r}, x_{1,r}) + h'(x_0)(x_{2,r}) \\ \vdots \\ \sum_r h_{K,r} = \sum_r \left(\sum_{\pi \in \mathsf{part}(K) \setminus \{\tilde{\pi}\}} \nu(\pi) h^{(\pi)}(x_0)(\otimes_{i \in \pi} x_{i,r}) \right) + \sum_r h'(x_0)(x_{K,r}) \end{pmatrix}$$

$$\to \begin{pmatrix} g_0 = g(h_0) \\ g_{1,r} = g'(h_0)(h_{1,r}) \\ g_{2,r} = g''(h_0)(h_{1,r}, h_{1,r}) + g'(h_0)(h_{2,r}) \\ \vdots \\ \sum_r g_{K,r} = \sum_r \left(\sum_{\pi \in \mathsf{part}(K) \setminus \{\tilde{\pi}\}} \nu(\pi) g^{(\pi)}(h_0)(\otimes_{i \in \pi} h_{i,r}) \right) + \sum_r g'(h_0)(h_{K,r}) \end{pmatrix}$$

Collapsed Taylor Mode

$$\begin{pmatrix} x_0 \\ x_{1,r} \\ x_{2,r} \\ \vdots \\ \sum_r x_{K,r} \end{pmatrix} \to \begin{pmatrix} h_0 = h(x_0) \\ h_{1,r} = h'(x_0)(x_{1,r}) \\ h_{2,r} = h''(x_0)(x_{1,r}, x_{1,r}) + h'(x_0)(x_{2,r}) \\ \vdots \\ \sum_r h_{K,r} = \sum_r \left(\sum_{\pi \in \mathsf{part}(K) \setminus \{\tilde{\pi}\}} \nu(\pi) h^{(\pi)}(x_0)(\otimes_{i \in \pi} x_{i,r}) \right) + h'(x_0) \left(\sum_r x_{K,r} \right) \end{pmatrix}$$

$$\to \begin{pmatrix} g_0 = g(h_0) \\ g_{1,r} = g'(h_0)(h_{1,r}) \\ g_{2,r} = g''(h_0)(h_{1,r}, h_{1,r}) + g'(h_0)(h_{2,r}) \\ \vdots \\ \sum_r g_{K,r} = \sum_r \left(\sum_{\pi \in \mathsf{part}(K) \setminus \{\tilde{\pi}\}} \nu(\pi) g^{(\pi)}(h_0)(\otimes_{i \in \pi} h_{i,r}) \right) + g'(h_0) \left(\sum_r h_{K,r} \right) \end{pmatrix}$$

Collapsed Taylor Mode

- Laplacian:

$$\sum_{r=1}^{n} f''(x_0)(e_r, e_r)$$

$\rightarrow K = 2, R = n, x_{1,r} = e_r, x_{2,r} = 0$

- Weighted Laplacian:

$$\sum_{r=1}^{n} f''(x_0)(v_r, v_r)$$

$\rightarrow K = 2, R = n, x_{1,r} = v_r, x_{2,r} = 0$

Collapsed Taylor Mode for Mixed-Partials

Biharmonic Operator as example for linear operator with mixed-partials:

$$\sum_{r_1=1}^{n} \sum_{r_2=1}^{n} f^{(4)}(x_0)(e_{r_1}, e_{r_1}, e_{r_2}, e_{r_2}).$$

Collapsed Taylor Mode for Mixed-Partials

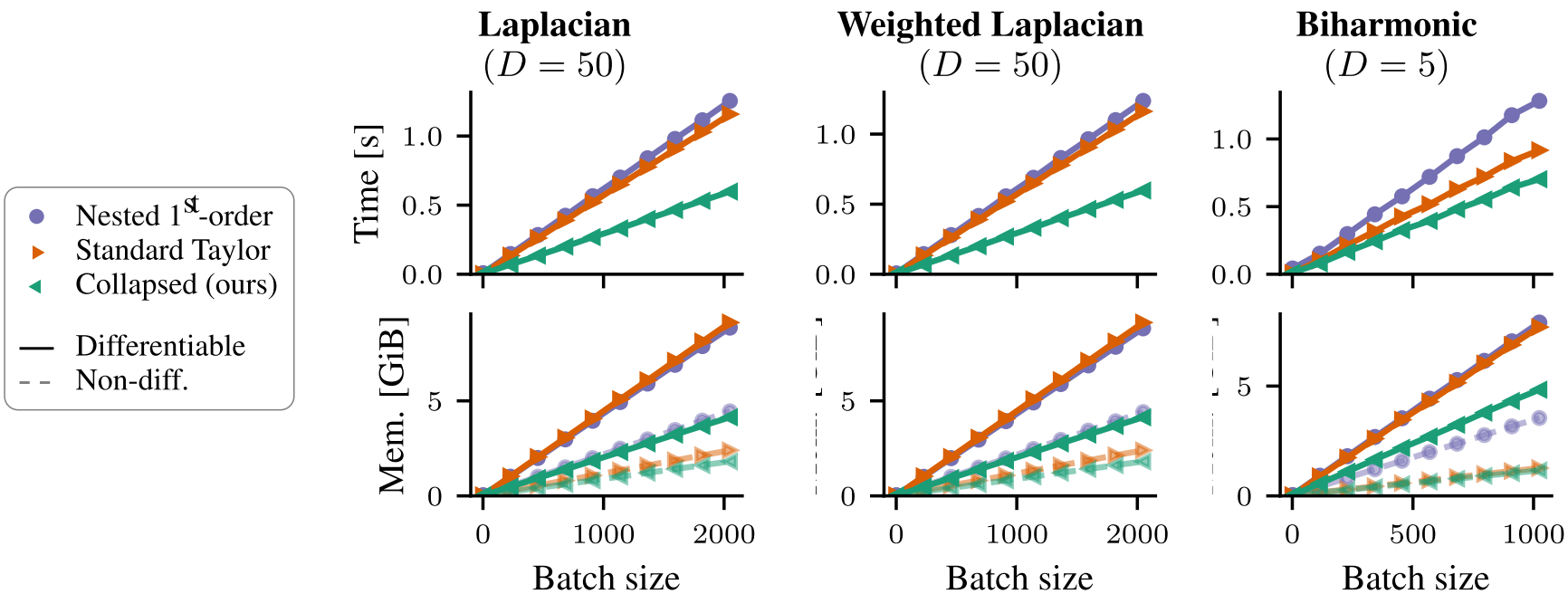
Biharmonic Operator as example for linear operator with mixed-partials:

$$\sum_{r_1=1}^{n} \sum_{r_2=1}^{n} f^{(4)}(x_0)(e_{r_1}, e_{r_1}, e_{r_2}, e_{r_2}).$$

Use interpolation formula ([4]) to enable Taylor Mode

$$\sum_{r_1=1}^{n} \sum_{r_2=1}^{n} f^{(4)}(x_0)(e_{r_1}, e_{r_1}, e_{r_2}, e_{r_2})$$
$$= \sum_{r_1=1}^{n} \sum_{r_2=1}^{n} \sum_{q \in \mathbb{N}^2, |q|=4} c_{(2,2),q} f^{(4)}(x_0) \left((q_1 e_{r_1} + q_2 e_{r_2})^{\otimes 4} \right).$$

Performance



([2])

Conclusion

- Leveraging the structure of the operator in the Faà di Bruno formula reduces runtime and memory.
- `pip install jet-for-pytorch`

Conclusion

- Leveraging the structure of the operator in the Faà di Bruno formula reduces runtime and memory.
- `pip install jet-for-pytorch`

Outlook

- Multivariate Taylor Mode could lead to even better schemes.
- Further symmetries should be exploited automatically as well.
- Numerical Properties of the collapsing method.
- Apply our method to other model types.

References I

[1] Ruichen Li, Haotian Ye, Du Jiang, Xuelan Wen, Chuwei Wang, Zhe Li, Xiang Li, Di He, Ji Chen, Weiluo Ren, and Liwei Wang. "A computational framework for neural network-based variational Monte Carlo with Forward Laplacian". In: *Nature Machine Intelligence* 6.2 (2024), pp. 209–219.

[2] Felix Dangel, Tim Siebert, Marius Zeinhofer, and Andrea Walther. "Collapsing Taylor Mode Automatic Differentiation". In: *The Thirty-ninth Annual Conference on Neural Information Processing Systems*. 2025.

[3] Michael Hardy. "Combinatorics of partial derivatives.". In: *The Electronic Journal of Combinatorics* 13.1 (2006).

References II

[4] Andreas Griewank, Jean Utke, and Andrea Walther. "Evaluating higher derivative tensors by forward propagation of univariate Taylor series". In: *Mathematics of Computation* 69 (1999).