Computing Higher-Order Derivatives with AD

Andrea Walther
Institut für Mathematik
Humboldt-Universität zu Berlin

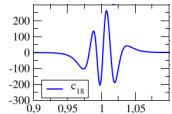
28th EuroAD Workshop
December 8, 2025

Outline

Where are Higher-Order Derivatives Needed?

- Simulation of complex system
  - definition, e.g., tooth gears
    O. Vogel: Accurate Gear Tooth Contact and Sensitivity Computation for Hypoid Bevel Gears (2009)

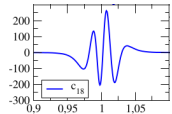
  - simulation, e.g., quantum chromodynamics
    M. Wagner, B.-J. Schaefer und A. Walther: On the efficient computation of high-order derivatives for implicitly defined functions (2010)



Where are Higher-Order Derivatives Needed?

- Simulation of complex system
  - definition, e.g., tooth gears
    O. Vogel: Accurate Gear Tooth Contact and Sensitivity Computation for Hypoid Bevel Gears (2009)

  - simulation, e.g., quantum chromodynamics
    M. Wagner, B.-J. Schaefer und A. Walther: On the efficient computation of high-order derivatives for implicitly defined functions (2010)



- Sensitivity analysis
- Variational Monte Carlo
- Physics-informed neural networks (PINNs)

Physics-Informed Neural Networks (PINNs)

Idea:
PINNs integrate *physical laws* (e.g. PDEs) into the training process of neural networks.

Goal:
Approximate a solution $u_\theta(x, t)$ of a PDE $\mathcal{N}[u_\theta(x, t)] = 0$, by minimizing

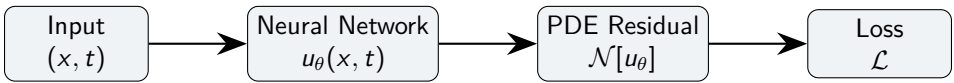
$$\mathcal{L} = \mathcal{L}_{\text{data}} + \mathcal{L}_{\text{PDE}}.$$

Physics-Informed Neural Networks (PINNs)

Idea:
PINNs integrate *physical laws* (e.g. PDEs) into the training process of neural networks.

Goal:
Approximate a solution $u_\theta(x, t)$ of a PDE $\mathcal{N}[u_\theta(x, t)] = 0$, by minimizing

$$\mathcal{L} = \mathcal{L}_{\text{data}} + \mathcal{L}_{\text{PDE}}.$$

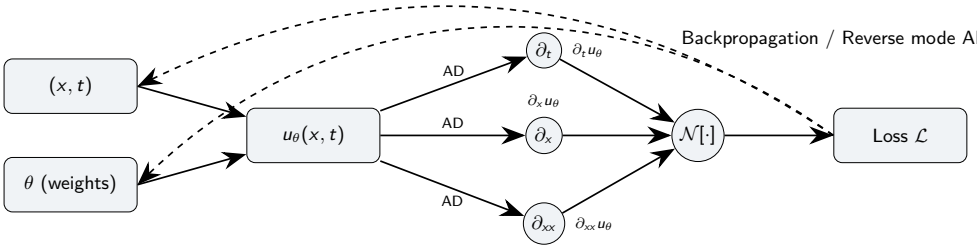


Key advantage: Leverages known physics to reduce data demand and improve generalization.

Derivatives for PINNs I

PINNs need spatial/temporal derivatives to inserted them into the PDE operator $\mathcal{N}$

$$\mathcal{N}[u_\theta] = \frac{\partial u_\theta}{\partial t} - \nu \frac{\partial^2 u_\theta}{\partial x^2} + u_\theta \frac{\partial u_\theta}{\partial x} = 0$$



Derivatives for PINNs II

Hence

1. calculation of derivatives for PDE (higher order!)

then

2. backpropagation (=reverse mode AD) applied to these derivatives

Derivatives for PINNs II

Hence

1. calculation of derivatives for PDE (higher order!)

   approaches:

   - nested AD
   - multivariate Taylor approach
   - univariate Taylor approach

   then

2. backpropagation (=reverse mode AD) applied to these derivatives

Derivatives for PINNs II

Hence

1. calculation of derivatives for PDE (higher order!)

   approaches:

   - nested AD
   - multivariate Taylor approach
   - univariate Taylor approach

   then

2. backpropagation (=reverse mode AD) applied to these derivatives

   $\Rightarrow$ order = order of PDE + 1!

Nested AD

Idea: Apply backpropagation repeatedly!

For $F : \mathbb{R}^n \mapsto \mathbb{R}^m$, one obtains

$$F \implies \begin{aligned} y &= F(x) \\ \bar{x}^\top &+= \bar{y}^\top F'(x) \end{aligned} \implies \begin{aligned} \tilde{\bar{y}} &+= F'(x)\tilde{\bar{x}} \\ \tilde{x}^\top &+= \bar{y}^\top F''(x)\tilde{\bar{x}} + \tilde{y}^\top F'(x) \end{aligned} \implies \cdots$$

input $x, \bar{y}$, output $y, \bar{x}$ — input $x, \bar{y}, \tilde{y}$, output $\tilde{\bar{y}}, \tilde{x}$

Nested AD

Idea: Apply backpropagation repeatedly!

For $F : \mathbb{R}^n \mapsto \mathbb{R}^m$, one obtains

$$F \implies \begin{matrix} y & = F(x) \\ \bar{x}^\top & += \bar{y}^\top F'(x) \end{matrix} \implies \begin{matrix} \tilde{\bar{y}} & += F'(x)\tilde{\bar{x}} \\ \tilde{x}^\top & += \bar{y}^\top F''(x)\tilde{\bar{x}} + \tilde{y}^\top F'(x) \end{matrix} \implies \cdots$$

input $x, \bar{y}$, output $y, \bar{x}$ — input $x, \bar{y}, \tilde{y}$, output $\tilde{\bar{y}}, \tilde{x}$

- often used in ML context, e.g., Shi, Hu, Lin, Kawaguchil: Stochastic Taylor Derivative Estimator: Efficient amortization for arbitrary differential operators (2024)
- computational complexity is $\mathcal{O}(n^d)$ with $n =$ dimension of x, $d =$ highest derivative order
  In the AD book:

  Rule 10: Never go back more than once!
- implementation might be difficult

Required Derivative Tensors

Given:

$$y \quad = \quad f(x) \quad \text{with } f : \mathcal{D} \subset \mathbb{R}^{\bar{n}} \mapsto \mathbb{R},$$

i.e., $m = 1$ for simplicity.

Required:

$$\nabla_S^k f(x) \equiv \frac{\partial^k}{\partial z^k} f(x + Sz)\Big|_{z=0} \in \mathbb{R}^{n^k}$$

for $S \in \mathbb{R}^{\bar{n} \times n}$, $z \in \mathbb{R}^n$ and $k = 1, \ldots, d$.

Required Derivative Tensors

Given:

$$y \quad = \quad f(x) \quad \text{with } f : \mathcal{D} \subset \mathbb{R}^{\bar{n}} \mapsto \mathbb{R},$$

i.e., $m = 1$ for simplicity.

Required:

$$\nabla_S^k f(x) \equiv \frac{\partial^k}{\partial z^k} f(x + Sz)\bigg|_{z=0} \in \mathbb{R}^{n^k}$$

for $S \in \mathbb{R}^{\bar{n} \times n}$, $z \in \mathbb{R}^n$ and $k = 1, \ldots, d$.

Examples:

$$\frac{\partial}{\partial z} f(x + Sz)\bigg|_{z=0} = (f'(x) s_j)_{j=1,\ldots,n} \in \mathbb{R}^n, \qquad \frac{\partial^2}{\partial z^2} f(x + Sz)\bigg|_{z=0} = (f''(x)\, s_i\, s_j)_{i=1,\ldots,n}^{j=1,\ldots,n} \in \mathbb{R}^{n \times n}$$

Required Derivative Tensors

Given:

$$y \quad = \quad f(x) \quad \text{with } f : \mathcal{D} \subset \mathbb{R}^{\bar{n}} \mapsto \mathbb{R},$$

i.e., $m = 1$ for simplicity.

Required:

$$\nabla_S^k f(x) \equiv \frac{\partial^k}{\partial z^k} f(x + Sz)\bigg|_{z=0} \in \mathbb{R}^{n^k}$$

for $S \in \mathbb{R}^{\bar{n} \times n}$, $z \in \mathbb{R}^n$ and $k = 1, \ldots, d$.

Examples:

$$\frac{\partial}{\partial z} f(x + Sz)\bigg|_{z=0} = (f'(x) s_j)_{j=1,\ldots,n} \in \mathbb{R}^n, \qquad \frac{\partial^2}{\partial z^2} f(x + Sz)\bigg|_{z=0} = (f''(x)\, s_i\, s_j)_{i=1,\ldots,n}^{j=1,\ldots,n} \in \mathbb{R}^{n \times n}$$

For third and higher derivative tensors, the matrix-vector notation is no longer sufficient!

Multivariate Taylor Approach

AD approach:

1. start with $\nabla_S x = S \in \mathbb{R}^{\bar{n} \times n}$ and $\nabla_S^k x = 0$, $1 \leq k \leq d$
2. propagate $\nabla_S^k w$ for all intermediates w according to AD, i.e., well-known rules

$$\begin{aligned} \nabla_S v &= u \, \nabla_S w + w \, \nabla_S u \\ \nabla_S^2 v &= u \, \nabla_S^2 w + \nabla_S u (\nabla_S w)^\top + \nabla_S w (\nabla_S u)^\top + w \, \nabla_S^2 u \\ &\dots \end{aligned}$$

3. extract $\nabla_S^k f(x), \; k = 0, \dots, d$

Multivariate Taylor Approach

AD approach:

1. start with $\nabla_S x = S \in \mathbb{R}^{\bar{n} \times n}$ and $\nabla_S^k x = 0$, $1 \leq k \leq d$
2. propagate $\nabla_S^k w$ for all intermediates w according to AD, i.e., well-known rules

$$\begin{aligned} \nabla_S v &= u \, \nabla_S w + w \, \nabla_S u \\ \nabla_S^2 v &= u \, \nabla_S^2 w + \nabla_S u (\nabla_S w)^\top + \nabla_S w (\nabla_S u)^\top + w \, \nabla_S^2 u \\ &\dots \end{aligned}$$

3. extract $\nabla_S^k f(x), \; k = 0, \dots, d$

Several papers by R. Neidinger (starting 1995)

Handling Derivative Tensors: Memory Requirement

Alternatives:

- store derivative tensors as n^k array
- exploit symmetry: all derivative tensors have

$$\binom{n+k-1}{k} = \frac{n \cdot (n+1) \cdots (n+k-1)}{1 \cdot 2 \cdots k} \approx \frac{n^k}{k!}$$

distinct elements $\Rightarrow$ Exploiting symmetry reduces memory requirement by $k!$

Handling Derivative Tensors: Memory Requirement

Alternatives:

- store derivative tensors as n^k array
- exploit symmetry: all derivative tensors have

$$\binom{n+k-1}{k} = \frac{n \cdot (n+1) \cdots (n+k-1)}{1 \cdot 2 \cdots k} \approx \frac{n^k}{k!}$$

distinct elements $\Rightarrow$ Exploiting symmetry reduces memory requirement by $k!$

But: Access more complicate and may be irregular
Note: Does not apply for $n = 1$!

Handling Derivative Tensors: Operations

$\nabla_S^k f(\boldsymbol{x}),\ k = 0, \ldots, d$ are polynomials of degree d in n variables, same holds for intermediates

Assume $w = v * u$ is part of evaluation of f

$\Rightarrow$ number of multiplications for $0 \le k \le d$:

$$\binom{2n+k-1}{k} = \sum_{j=0}^{k} \binom{n+j-1}{j} \binom{n+k-j-1}{k-j}$$

$\Rightarrow$ total number of iterations:

$$\binom{2n+d}{d} = \sum_{k=0}^{d} \binom{2n+k-1}{k}$$

Univariate Taylor Approach I

Now: $n = 1$, i.e., think in Taylor polynomials!

Let x be the path

$$x(t) \equiv \sum_{j=0}^{d} x_j t^j : \mathbb{R} \mapsto \mathbb{R}^n \qquad \text{with} \qquad x_j = \frac{1}{j!} \frac{\partial^j}{\partial t^j} x(t) \Big|_{t=0}$$

Univariate Taylor Approach I

Now: $n = 1$, i.e., think in Taylor polynomials!

Let x be the path

$$x(t) \equiv \sum_{j=0}^{d} x_j t^j : \mathbb{R} \mapsto \mathbb{R}^n \qquad \text{with} \qquad x_j = \frac{1}{j!} \frac{\partial^j}{\partial t^j} x(t) \Big|_{t=0}$$

Hence

x_j is scaled derivative at $t = 0 \quad \Rightarrow \quad x_1 =$ tangent, $x_2 =$ curvature

Univariate Taylor Approach I

Now: $n = 1$, i.e., think in Taylor polynomials!

Let x be the path

$$x(t) \equiv \sum_{j=0}^{d} x_j t^j : \mathbb{R} \mapsto \mathbb{R}^n \qquad \text{with} \qquad x_j = \frac{1}{j!} \frac{\partial^j}{\partial t^j} x(t) \Big|_{t=0}$$

Hence

$$x_j \text{ is scaled derivative at } t = 0 \quad \Rightarrow \quad x_1 = \text{tangent}, \; x_2 = \text{curvature}$$

F smooth enough $\Rightarrow$ There exist Taylor coefficients $y_j \in \mathbb{R}$, $0 \le j \le d$, with

$$y(t) = \sum_{j=0}^{d} y_j t^j + \mathcal{O}(t^{d+1}).$$

Univariate Taylor Approach II

Using the chain rule, one obtains:

$$
\begin{aligned}
y_0 &= F(x_0) \\
y_1 &= F'(x_0)\, x_1 \\
y_2 &= F'(x_0)\, x_2 + \tfrac{1}{2} F''(x_0)\, x_1 x_1 \\
y_3 &= F'(x_0)\, x_3 + F''(x_0)\, x_1 x_2 + \tfrac{1}{6} F'''(x_0)\, x_1 x_1 x_1 \\
&\dots
\end{aligned}
$$

Univariate Taylor Approach II

Using the chain rule, one obtains:

$$
\begin{aligned}
y_0 &= F(x_0) \\
y_1 &= F'(x_0)\, x_1 \\
y_2 &= F'(x_0)\, x_2 + \tfrac{1}{2} F''(x_0)\, x_1 x_1 \\
y_3 &= F'(x_0)\, x_3 + F''(x_0)\, x_1 x_2 + \tfrac{1}{6} F'''(x_0)\, x_1 x_1 x_1 \\
&\dots
\end{aligned}
$$

Implementation?

- naive way: qubic computational cost in highest degree d
- Faà Di Bruno (1857!): tailored approach (quadratic computational cost in d)

Univariate Taylor Approach II

Using the chain rule, one obtains:

$$\begin{aligned} y_0 &= F(x_0) \\ y_1 &= F'(x_0)\, x_1 \\ y_2 &= F'(x_0)\, x_2 + \tfrac{1}{2} F''(x_0)\, x_1 x_1 \\ y_3 &= F'(x_0)\, x_3 + F''(x_0)\, x_1 x_2 + \tfrac{1}{6} F'''(x_0)\, x_1 x_1 x_1 \\ &\dots \end{aligned}$$

Implementation?

- naive way: qubic computational cost in highest degree d
- Faà Di Bruno (1857!): tailored approach (quadratic computational cost in d)

Hence:

$$\mathrm{OPS}(x_0, \dots, x_d \to y_0, \dots, y_d) \sim d^2\, \mathrm{OPS}(x_0 \to y_0)$$

Univariate Taylor Mode: Example

From

$$\begin{aligned} y_0 &= F(x_0) \\ y_1 &= F'(x_0)\, x_1 \\ y_2 &= F'(x_0)\, x_2 + \tfrac{1}{2} F''(x_0)\, x_1 x_1 \\ y_3 &= F'(x_0)\, x_3 + F''(x_0)\, x_1 x_2 + \tfrac{1}{6} F'''(x_0)\, x_1 x_1 x_1 \end{aligned}$$

it follows for $x_1 = 1, x_2 = 0, x_3 = 0$ that

$$\begin{aligned} y_0 &= F(x_0) \\ y_1 &= F'(x_0)\, x_1 \\ y_2 &= \tfrac{1}{2} F''(x_0)\, x_1 x_1 \\ y_3 &= \tfrac{1}{6} F'''(x_0)\, x_1 x_1 x_1 \end{aligned}$$

i.e., scaled Taylor coefficients of F!

Univariate Taylor Approach: Complexity

One can show:
$\binom{2n+d-1}{d}$ univariate Taylor polynomials required for derivative tensors $\nabla_S^k f(x),\ k = 0, \ldots, d$

Univariate Taylor Approach: Complexity

One can show:
$\binom{2n+d-1}{d}$ univariate Taylor polynomials required for derivative tensors $\nabla_S^k f(x),\ k = 0, \ldots, d$

Storage: Simple d-dimensional array per univariate polynomial

Univariate Taylor Approach: Complexity

One can show:
$\binom{2n+d-1}{d}$ univariate Taylor polynomials required for derivative tensors $\nabla_S^k f(x),\ k = 0, \ldots, d$

Storage: Simple d-dimensional array per univariate polynomial

Operation count: Assume $w = v * u$ is part of evaluation of f

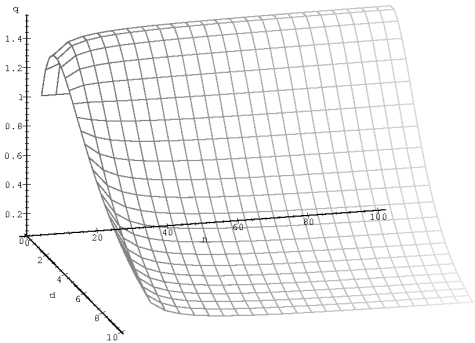
$\Rightarrow$ number of multiplications for $0 \le k \le d$:

$$\frac{(d+2)(d+1)}{2} = \binom{d+2}{d}$$

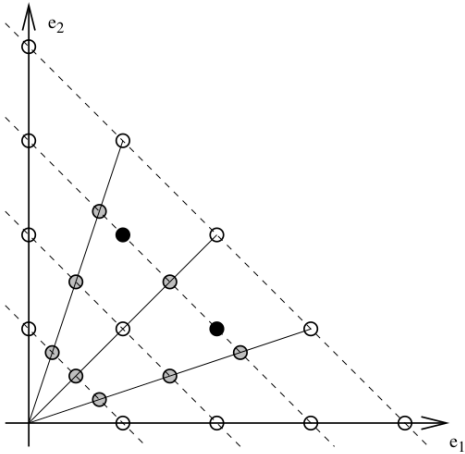
$\Rightarrow$ total number of iterations:

$$\binom{n+d-1}{d}\binom{d+2}{d} = q(d,n)\binom{2n+d}{d}$$

The Function $q(.,.)$



Choosing the Taylor Polynomials



$$n = 2, d = 4$$

$$\Rightarrow$$

$$\binom{n+d-1}{d} = \binom{5}{4} = 5$$

Calculating the Missing Values

With

$$f_{\boldsymbol{i}}(\boldsymbol{x}) \;=\; \sum_{\boldsymbol{0}<\boldsymbol{k}\leq\boldsymbol{i}} \binom{\boldsymbol{i}}{\boldsymbol{k}} (-1)^{|\boldsymbol{i}-\boldsymbol{k}|} f^{(|\boldsymbol{i}|)}(\boldsymbol{x};\boldsymbol{S}\boldsymbol{k}),\;\; f^{(m)}(\boldsymbol{x};\boldsymbol{S}\boldsymbol{k}) = \sum_{|\boldsymbol{j}|=d} \left(\frac{|\boldsymbol{k}|}{d}\right)^{m} \binom{d\boldsymbol{k}/|\boldsymbol{k}|}{\boldsymbol{j}} f^{(m)}(\boldsymbol{x};\boldsymbol{S}\boldsymbol{j})$$

one obtains

$$\left.\frac{\partial^{|\boldsymbol{i}|} f(\boldsymbol{x}+\boldsymbol{S}\boldsymbol{z})}{\partial z_1^{\boldsymbol{i}_1}\partial z_2^{\boldsymbol{i}_2}\ldots\partial z_n^{\boldsymbol{i}_n}}\right|_{\boldsymbol{z}=\boldsymbol{0}} = \sum_{|\boldsymbol{j}|=d} f^{(|\boldsymbol{i}|)}(\boldsymbol{x};\boldsymbol{S}\boldsymbol{j}) \underbrace{\sum_{\boldsymbol{0}<\boldsymbol{k}\leq\boldsymbol{i}} (-1)^{|\boldsymbol{i}-\boldsymbol{k}|} \binom{\boldsymbol{i}}{\boldsymbol{k}} \binom{d\boldsymbol{k}/|\boldsymbol{k}|}{\boldsymbol{j}} \left(\frac{|\boldsymbol{k}|}{d}\right)^{|\boldsymbol{i}|}}_{\equiv\, c_{\boldsymbol{i},\boldsymbol{j}}}$$

A. Griewank, J. Utke, A. Walther: Evaluating higher derivative tensors by forward propagation of univariate Taylor series. Mathematics of Computation (2000)

Application: Quantum Chromodynamics (QCD)

linear Sigma Model (LσM) given by

$$\Omega(T, \mu; \sigma_q, \sigma_s) \;=\; U(\sigma_q, \sigma_s) + \Omega_{\bar{q}q}(T, \mu; \sigma_q, \sigma_s)$$

$$\left.\frac{\partial \Omega(T, \mu; \sigma_q, \sigma_s))}{\partial \sigma_i}\right|_{\substack{\sigma_q = \bar{\sigma}_q, \\ \sigma_s = \bar{\sigma}_s}} = 0 \quad i = q, s \qquad \Rightarrow \qquad \bar{\sigma}_i \;=\; \bar{\sigma}_i(T, \mu) \qquad i = q, s \; .$$

Application: Quantum Chromodynamics (QCD)

linear Sigma Model (LσM) given by

$$\Omega(T,\mu;\sigma_q,\sigma_s) \;=\; U(\sigma_q,\sigma_s) + \Omega_{\bar{q}q}(T,\mu;\sigma_q,\sigma_s)$$

$$\left.\frac{\partial\Omega(T,\mu;\sigma_q,\sigma_s))}{\partial\sigma_i}\right|_{\substack{\sigma_q=\bar{\sigma}_q,\\ \sigma_s=\bar{\sigma}_s}} = 0 \quad i=q,s \qquad \Rightarrow \qquad \bar{\sigma}_i \;=\; \bar{\sigma}_i(T,\mu) \qquad i=q,s\ .$$

grand potential evaluated for fixed $\bar{\sigma}_i, i=q,s$:

$$\bar{\Omega}(T,\mu) \;= \Omega\left(T,\mu;\bar{\sigma}_q(T,\mu),\bar{\sigma}_s(T,\mu)\right) = \textstyle\sum_{n=0}^{\infty} c_n(T)\left(\frac{\mu}{T}\right)^n$$

with

$$c_n(T) = \frac{1}{n!}\left.\frac{\partial^n\left(p(T,\mu)/T^4\right)}{\partial\left(\mu/T\right)^n}\right|_{\mu=0}\ .$$

Application: Quantum Chromodynamics (QCD)

linear Sigma Model (LσM) given by

$$\Omega(T,\mu;\sigma_q,\sigma_s) \;=\; U(\sigma_q,\sigma_s) + \Omega_{\bar{q}q}(T,\mu;\sigma_q,\sigma_s)$$

$$\left.\frac{\partial\Omega(T,\mu;\sigma_q,\sigma_s))}{\partial\sigma_i}\right|_{\substack{\sigma_q=\bar{\sigma}_q,\\ \sigma_s=\bar{\sigma}_s}} = 0 \quad i=q,s \qquad \Rightarrow \qquad \bar{\sigma}_i \;=\; \bar{\sigma}_i(T,\mu) \qquad i=q,s\ .$$

grand potential evaluated for fixed $\bar{\sigma}_i, i=q,s$:

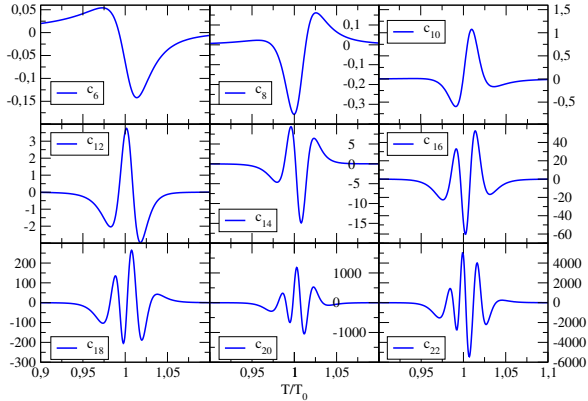
$$\bar{\Omega}(T,\mu) \;= \Omega\left(T,\mu;\bar{\sigma}_q(T,\mu),\bar{\sigma}_s(T,\mu)\right) = \textstyle\sum_{n=0}^{\infty} c_n(T)\left(\frac{\mu}{T}\right)^n$$

with

$$c_n(T) = \frac{1}{n!}\left.\frac{\partial^n\left(p(T,\mu)/T^4\right)}{\partial\left(\mu/T\right)^n}\right|_{\mu=0}\ .$$

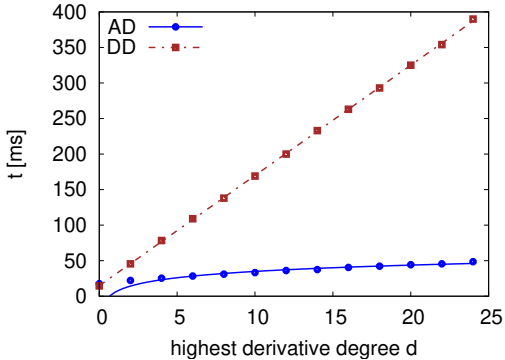
$\Rightarrow$ Use AD to compute these derivatives of an implicit function!

Calculated Taylor Coefficients



(Wagner, Walther, Schaefer 2010)

Timings for the Calculated Taylor Coefficients



(Wagner, Walther, Schaefer 2010)

Summary

- Higher-order derivatives important for PINNS and Variational Monte Carlo Simulations
- Nesting AD: In general not efficient!
- univariate Taylor approach

Summary

- Higher-order derivatives important for PINNS and Variational Monte Carlo Simulations
- Nesting AD: In general not efficient!
- univariate Taylor approach
  In the AD book:

  Rule 4: Products of transpose-Jacobians with vectors are cheap,
  but their trace is as expensive as the whole Jacobian.

Summary

- Higher-order derivatives important for PINNS and Variational Monte Carlo Simulations
- Nesting AD: In general not efficient!
- univariate Taylor approach
  In the AD book:

  Rule 4: Products of transpose-Jacobians with vectors are cheap,
  but their trace is as expensive as the whole Jacobian.

  Recently, we found out: This is not true for Hessians, see talk of Tim Siebert

Summary

- Higher-order derivatives important for PINNS and Variational Monte Carlo Simulations
- Nesting AD: In general not efficient!
- univariate Taylor approach
  In the AD book:

  > Rule 4: Products of transpose-Jacobians with vectors are cheap,
  > but their trace is as expensive as the whole Jacobian.

  Recently, we found out: This is not true for Hessians, see talk of Tim Siebert
- multivariate Taylor approach
  Might be interesting in this context, subject of future rerearch