

Algorithmic Derivatives of Electromagnetic Shower Simulations

Max Ahle

Scientific Computing Group,
University of Kaiserslautern-Landau
& SIVERT Research Training Group

Fourth MODE Workshop
Valencia, Sep 22th, 2024

Outline:

- 1 Autodiff
- 2 Technical Challenges
- 3 Math Challenges



Algorithmic Differentiation (AD) / Diff. Programming

- Set of techniques to **evaluate derivatives of computer-implemented functions**.
- Useful e. g. for **gradient-based optimization** of a computationally heavy loss function, finding optimal design parameters, model parameters etc.
- **Forward mode** of AD with a single AD input x : For each number a handled by the primal program, keep track of $\dot{a} = \frac{\partial a}{\partial x}$, augmenting all real-arithmetic operations:

$$c = a + b \rightsquigarrow \dot{c} = \dot{a} + \dot{b}$$

$$c = a \cdot b \rightsquigarrow \dot{c} = \dot{a} \cdot b + a \cdot \dot{b}$$

etc. Run-time and memory performance asymptotics match those of numerical differentiation, but AD is exact.


- **Reverse mode of AD**: More complicated. Allows to compute a gradient of a single AD output w. r. t. many AD inputs in one stroke.
 \rightsquigarrow Extremely useful for optimization.

AD Tools

AD tools identify real-arithmetic operations in the primal program and insert the appropriate AD logic. Different mechanisms have been reported:

- Source Transformation
- Operator Overloading, e. g. CoDiPack, ADOL-C, Torch
- Hooking into the compiler, e. g. Clad by V. Vassilev.
- Dynamic binary instrumentation of machine code, Derivgrind from RPTU (M. Aehle's PhD topic).
- Hardware

Video (7 min) about Derivgrind+LibreOffice Calc:

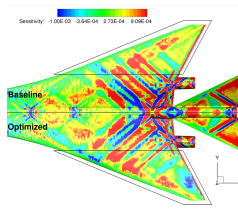
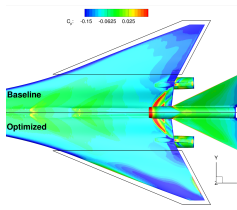
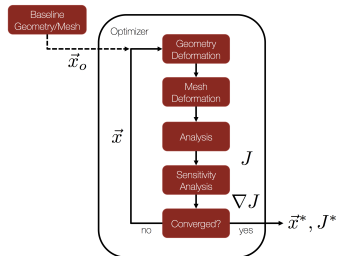
<https://t1p.de/tt4ne> 



AD-powered Aerospace Design Optimization

AD of simulation code provides gradient of objective function (e. g. drag) w. r. t. design parameters (e. g. airfoil shape).

The gradient points into the direction of steepest ascent. Use this to iteratively improve a given baseline design.¹



Pressure coefficient (left) and drag surface sensitivity (right) for the baseline (upper half) and optimized shape (lower half) of an aircraft at supersonic cruise conditions.²

¹Flow chart by T. Economou, https://su2code.github.io/tutorials/Inviscid_2D_Unconstrained_NACA0012/

²T. Albring, M. Sagebaum, N. R. Gauger 2016. Efficient Aerodynamic Design using the Discrete Adjoint Method in SU2. AIAA 2016-3518.

Can AD be useful for HEP?

- There were (gradient-free) optimization studies already while designing the LHC.³
- Studies concerned with single experiments have shown ample room for optimization.⁴
- No differentiated version of Geant4 yet.
- To make it work, we need to solve **technical and mathematical challenges**.

³S. Russenschuck, T. Tortschanoff. Mathematical Optimization of Superconducting Accelerator Magnets, IEEE Trans. on Magnetics 30 (5) 1994.

⁴T. Dorigo, Geometry optimization of a muon-electron scattering detector, Physics Open 4 (2020) 100022

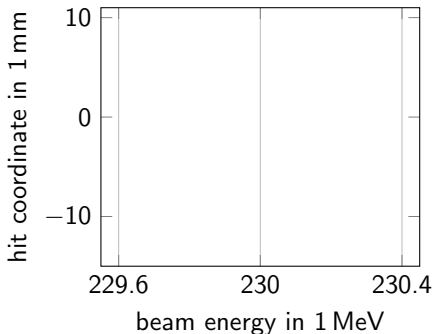
Technical Challenges

Geant4 is a large and complex codebase.

- Can we apply an AD tool with reasonable development efforts?
- How does it affect run-time and memory consumption?

GATE/Geant4 Setup

- GATE is a medical imaging toolkit built on top of Geant4.
- In our setup, a **single proton** passes through a human head and the Bergen pCT calorimeter.

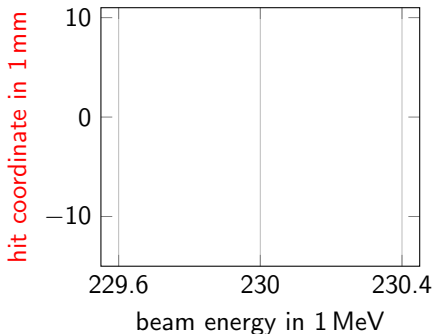


- First tracking layer.
- ◆ Second tracking layer.

M. Aehle et al. Exploration of Differentiability in a Proton CT Simulation Framework. Phys. Med. Biol., 2023.

GATE/Geant4 Setup

- GATE is a medical imaging toolkit built on top of Geant4.
- In our setup, a single proton passes through a human head and the Bergen pCT calorimeter.
- **AD outputs** (“derivative of...”): x-coordinates of hits in the first two “tracking layers” of the detector.

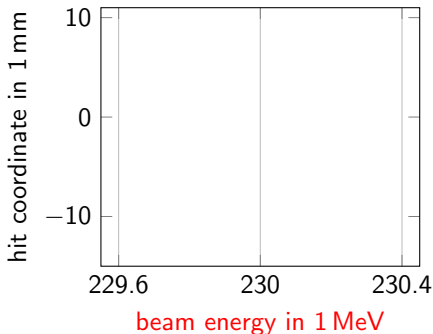


- First tracking layer.
- ◆ Second tracking layer.

M. Aehle et al. Exploration of Differentiability in a Proton CT Simulation Framework. Phys. Med. Biol., 2023.

GATE/Geant4 Setup

- GATE is a medical imaging toolkit built on top of Geant4.
- In our setup, a single proton passes through a human head and the Bergen pCT calorimeter.
- AD outputs (“derivative of...”): x-coordinates of hits in the first two “tracking layers” of the detector.
- AD input (“with respect to...”): beam energy.

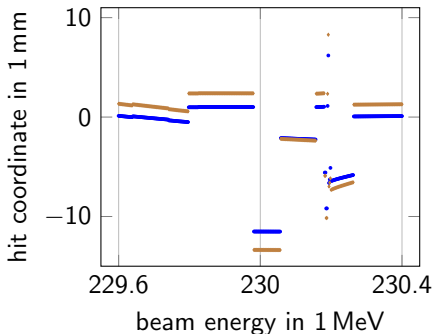


- First tracking layer.
- ◆ Second tracking layer.

M. Aehle et al. Exploration of Differentiability in a Proton CT Simulation Framework. Phys. Med. Biol., 2023.

GATE/Geant4 Setup

- GATE is a medical imaging toolkit built on top of Geant4.
- In our setup, a single proton passes through a human head and the Bergen pCT calorimeter.
- AD outputs (“derivative of...”):
x-coordinates of hits in the first two “tracking layers” of the detector.
- AD input (“with respect to...”):
beam energy.
- The seed of the random number generator was fixed.

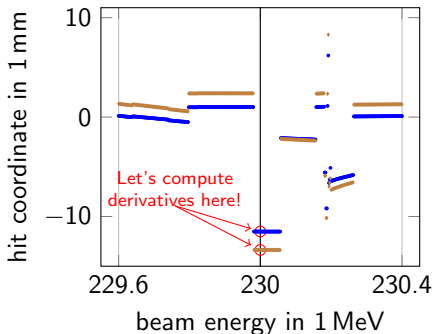


- First tracking layer.
- ◆ Second tracking layer.

M. Aehle et al. Exploration of Differentiability in a Proton CT Simulation Framework. Phys. Med. Biol., 2023.

GATE/Geant4 Setup

- GATE is a medical imaging toolkit built on top of Geant4.
- In our setup, a single proton passes through a human head and the Bergen pCT calorimeter.
- AD outputs (“derivative of...”):
x-coordinates of hits in the first two “tracking layers” of the detector.
- AD input (“with respect to...”):
beam energy.
- The seed of the random number generator was fixed.



- First tracking layer.
- ◆ Second tracking layer.

M. Aehle et al. Exploration of Differentiability in a Proton CT Simulation Framework. Phys. Med. Biol., 2023.

Code Modifications in GATE/Geant4

- Mark the input variable (beam energy):

```
if (command == pIonCmd) {
    pSourcePencilBeam->SetIonParameter(newValue);
}
if (command == pEnergyCmd) {
    double energy = pEnergyCmd->GetNewDoubleValue(newValue);
+   double one = 1.0;
+   DG_SET_DOTVALUE(&energy, &one, sizeof(double));
    pSourcePencilBeam->SetEnergy(energy);
}
```

- Mark the output variables (hit positions):

```
if (m_rootHitFlag) m_treeHit->Fill();
+ float pos = *(float*)(m_treeHit->GetBranch("posX")->GetAddress());
+ float pos_d;
+ DG_GET_DOTVALUE(&pos, &pos_d, sizeof(float));
+ std::cout << "pos_d=" << pos_d << "\n";
```

- Replace G4Log → std::log.

Code Modifications in GATE/Geant4

- Mark the input variable (beam energy):

```
if (command == pIonCmd) {
    pSourcePencilBeam->SetIonParameter(newValue);
}
if (command == pEnergyCmd) {
    double energy = pEnergyCmd->GetNewDoubleValue(newValue);
+   double one = 1.0;
+   DG_SET_DOTVALUE(&energy, &one, sizeof(double));
    pSourcePencilBeam->SetEnergy(energy);
}
```

- Mark the output variables (hit positions):

```
if (m_rootHitFlag) m_treeHit->Fill();
+ float pos = *(float*)(m_treeHit->GetBranch("posX")->GetAddress());
+ float pos_d;
+ DG_GET_DOTVALUE(&pos, &pos_d, sizeof(float));
+ std::cout << "pos_d=" << pos_d << "\n";
```

- Replace G4Log → std::log.

↪ With these changes, Derivgrind computes correct derivatives of Geant4!

Technical Challenges

Geant4 is a large and complex codebase.

- Can we apply an AD tool with reasonable development efforts?
- How does it affect run-time and memory consumption?

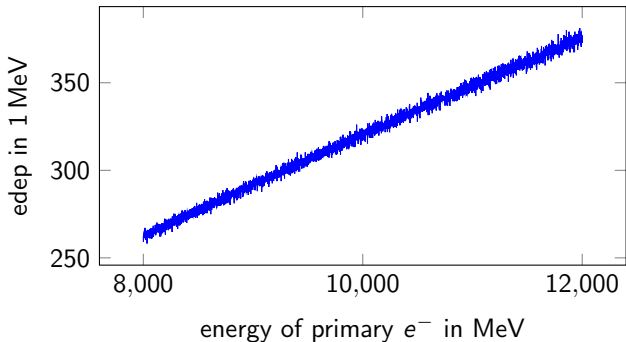
Technical Challenges

Geant4 is a large and complex codebase.

- Can we apply an AD tool with reasonable development efforts?
Yes, Derivgrind needs changes in ~ 10 lines of code.
- How does it affect run-time and memory consumption?
Derivgrind's forward/reverse mode gives a factor of 65/120 and scales the memory consumption by 2/3.

Mathematical Challenges

Now we'll switch from a single low-energy particle to millions of high-energy events, and average:

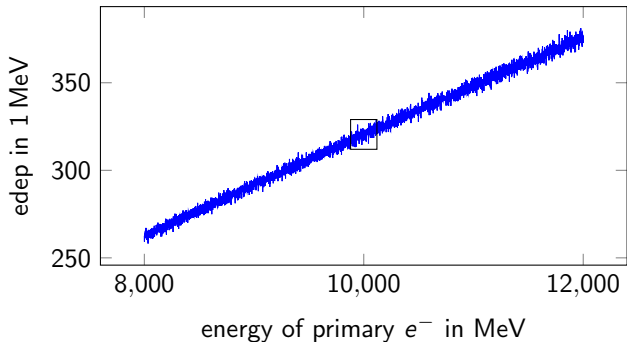


Energy deposition of an EM shower in some layer of a sampling calorimeter, depending on the energy of the primary particle. Simulated with G4HepEm/HepEmShow, 1k events per data point.

Even though the “noise” has a low magnitude, its derivative can have a large magnitude! The derivative can also be zero with probability 1...

Mathematical Challenges

Now we'll switch from a single low-energy particle to millions of high-energy events, and average:

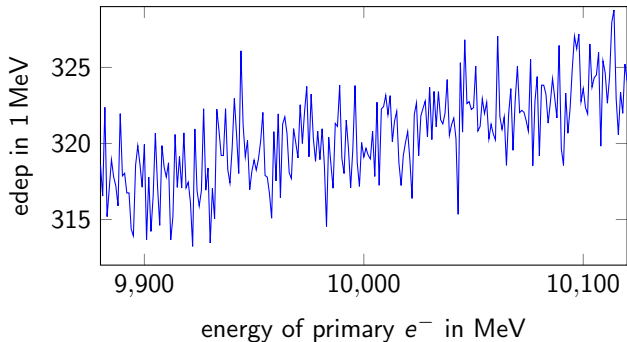


Energy deposition of an EM shower in some layer of a sampling calorimeter, depending on the energy of the primary particle. Simulated with G4HepEm/HepEmShow, 1k events per data point.

Even though the “noise” has a low magnitude, its derivative can have a large magnitude! The derivative can also be zero with probability 1...

Mathematical Challenges

Now we'll switch from a single low-energy particle to millions of high-energy events, and average:



Energy deposition of an EM shower in some layer of a sampling calorimeter, depending on the energy of the primary particle. Simulated with G4HepEm/HepEmShow, 1k events per data point.

Even though the “noise” has a low magnitude, its derivative can have a large magnitude! The derivative can also be zero with probability 1...

Mathematical Challenges

- Is the variance of the derivative sufficiently low? Otherwise we'd have to average over too many events to get a trustworthy result. . .
- When we average the derivative where it exists, do we get the derivative of the averages?
- Can the AD derivatives be useful for optimization?

G4HepEm and HepEmShow



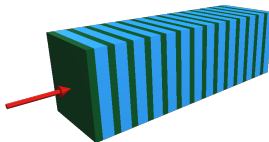
- R & D project by Mihaly Novak, Jonas Hahnfeld, Ben Morgan
- Simulation of electromagnetic showers (e^- , e^+ , γ)
- Isolates the required data and functionalities from Geant4, well documented, customizable.

github.com/mnovak42/g4hepem

github.com/mnovak42/hepemshow



- Created by Mihaly Novak
- Self-contained application using G4HepEm but not Geant4.
- Simulates electromagnetic showers in a sandwich calorimeter.

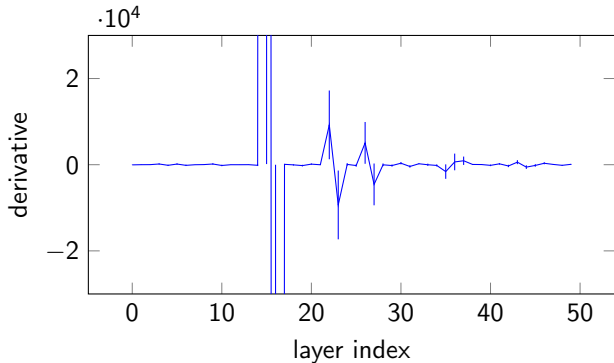


$\frac{\partial (\text{Energy Deposition})}{\partial (\text{Primary Particle Energy})}$ for the fully detailed simulation

Applied the AD tool CoDiPack (took ~ 3 days).

$\frac{\partial(\text{Energy Deposition})}{\partial(\text{Primary Particle Energy})}$ for the fully detailed simulation

Applied the AD tool CoDiPack (took ~ 3 days).

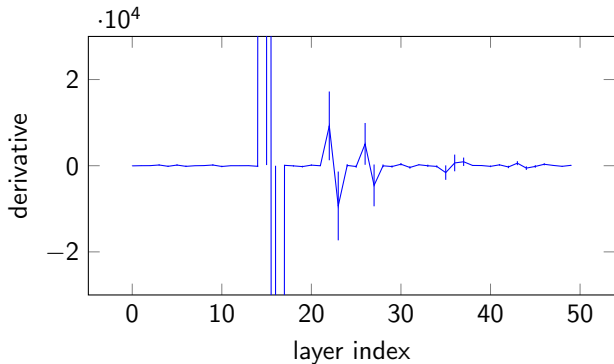


AD at 10 GeV

Look at the vertical axis range. These noisy derivatives can hardly be useful.

$\frac{\partial(\text{Energy Deposition})}{\partial(\text{Primary Particle Energy})}$ for the fully detailed simulation

Applied the AD tool CoDiPack (took ~ 3 days).

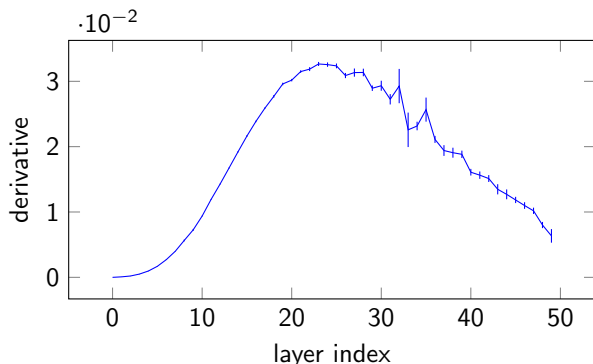


AD at 10 GeV

Look at the vertical axis range. These noisy derivatives can hardly be useful.

\rightsquigarrow **Deactivate multiple scattering** in the simulation.

$\frac{\partial (\text{Energy Deposition})}{\partial (\text{Primary Particle Energy})}$ with simplifications

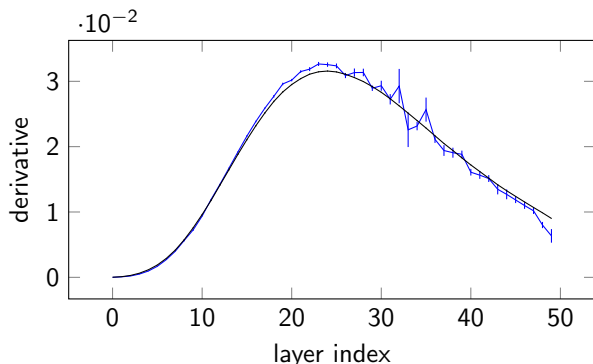


AD at 10 GeV

multiple scattering
disabled

This looks much better!

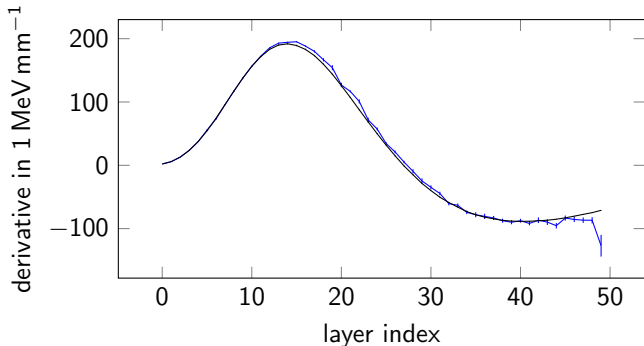
$\frac{\partial (\text{Energy Deposition})}{\partial (\text{Primary Particle Energy})}$ with simplifications



AD at 10 GeV
Diff.quot. at
9.9... 10.1 GeV
multiple scattering
disabled

This looks much better!

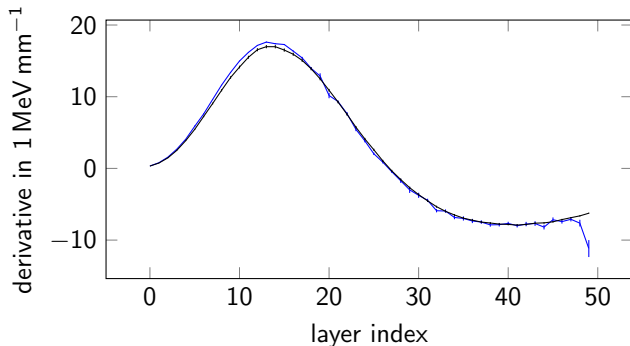
What about other AD inputs?

$$\frac{\partial(\text{Energy Deposition})}{\partial(\text{Absorber Thickness})}$$
 with simplifications


AD at 2.3 mm
 Diff.quot. at
 2.29... 2.31 mm
 multiple scattering and
 fluctuation disabled

Good agreement as well.

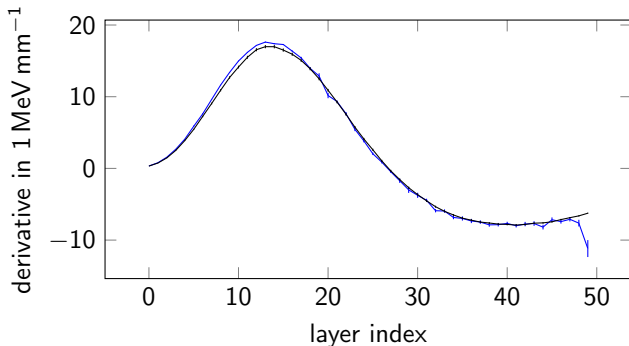
$\frac{\partial(\text{Energy Deposition})}{\partial(\text{Gap Thickness})}$ with simplifications



AD at 5.7 mm
 Diff.quot. at
 5.65... 5.75 mm
 multiple scattering and
 fluctuation disabled

Good agreement as well.

$\frac{\partial(\text{Energy Deposition})}{\partial(\text{Gap Thickness})}$ with simplifications

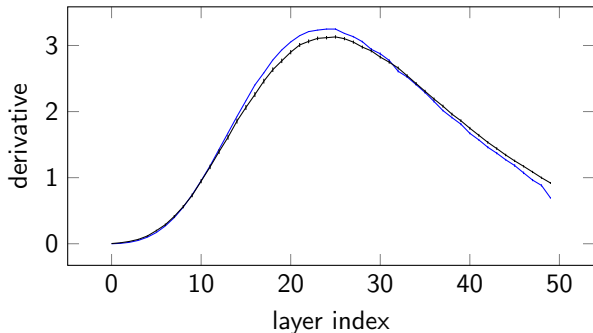


AD at 5.7 mm
Diff.quot. at
5.65... 5.75 mm
multiple scattering and
fluctuation disabled

Good agreement as well.

Going back to the $\frac{\partial(\text{Energy Deposition})}{\partial(\text{Primary Particle Energy})}$ plot: Let's reduce the step size of the difference quotient to reduce the truncation error. Let's take more events to make the error bars smaller.

$\frac{\partial (\text{Energy Deposition})}{\partial (\text{Primary Particle Energy})}$ with simplifications, many events



AD at 10 GeV
 Diff.quot. at
 9.995... 10.005 GeV
 multiple scattering and
 fluctuation disabled

The $\sim 5\%$ deviation is **small but statistically significant**.

Hypothesis: The 5% bias has to do with this \rightarrow
 There is a novel method⁵ to handle some of these
 constructs; implementation will be lots of work.

```

double p = /* diff'able */;
if( rng->flat() < p ){
    // ... do something ...
}
    
```

⁵G. Arya et al. Automatic Differentiation of Programs with Discrete Randomness. NeurIPS 2022.

Gradient-Based Optimization Problem

Given a

- primary energy e and
- absorber thickness a ,

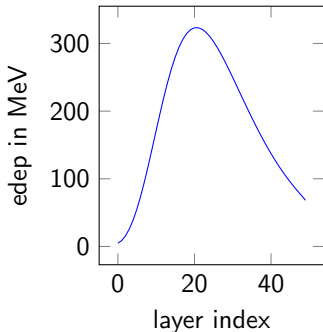
HepEmShow computes the resulting edep distribution $f_i(e, a)$ in the layers $i = 0, \dots, 49$.

**Given $f(e^*, a^*)$ (e. g. measured),
can we infer e^*, a^* ?**

↪ For the loss function

$$L(e, a) = \|f(e, a) - f(e^*, a^*)\|_2^2,$$

find $\min_{e, a} L(e, a)$!



Gradient-Based Optimization Setup

$$\underbrace{\frac{\partial L}{\partial(e, a)}(e_i, a_i)}_{2 \times 1} = \underbrace{\frac{\partial L}{\partial f}(f(e_i, a_i))}_{1 \times 50} \cdot \underbrace{\frac{\partial f}{\partial(e, a)}(e_i, a_i)}_{50 \times 2}$$

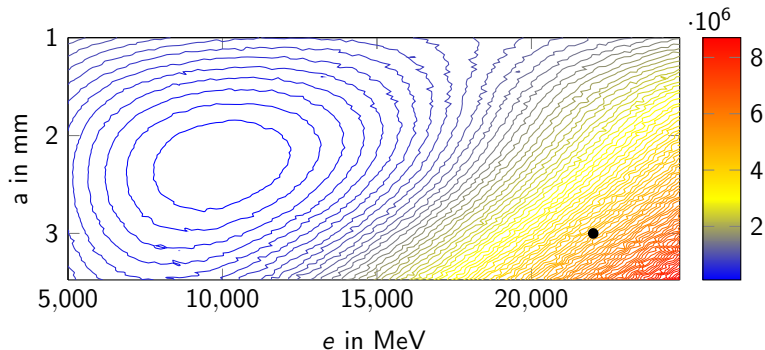
Starting with some (e_0, a_0) , iteratively,

- 1 Evaluate $f(e_i, a_i)$, i. e. run HepEmShow without AD,
- 2 Evaluate $\frac{\partial L}{\partial f}(f(e_i, a_i)) = 2(f(e_i, a_i) - f(e^*, a^*))^T$,
- 3 Evaluate the vector-jacobian product (vjp) with $\frac{\partial f}{\partial(e, a)}(e, a)$, i. e. run HepEmShow with reverse-mode AD.
- 4 Gradient descent step with step-sizes λ_e, λ_a :

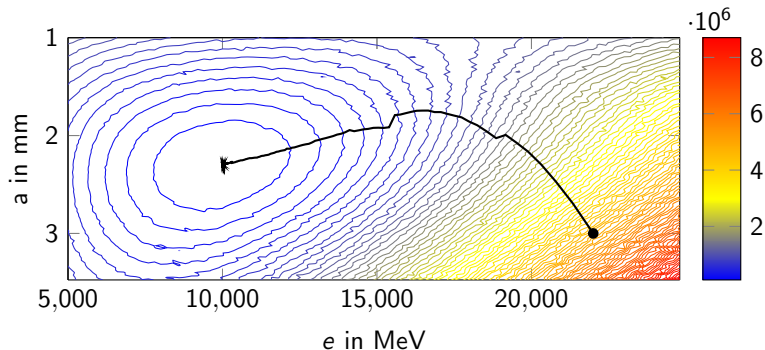
$$e_{i+1} = e_i - \lambda_e \cdot \frac{\partial L}{\partial e}(e_i, a_i)$$

$$a_{i+1} = a_i - \lambda_a \cdot \frac{\partial L}{\partial a}(e_i, a_i)$$

Gradient-Based Optimization Results

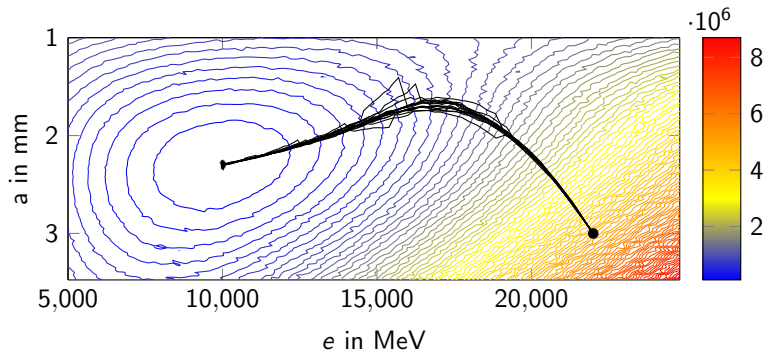


Gradient-Based Optimization Results



350 gradient descent steps with $\lambda_e = 1$, $\lambda_a = 10^{-7} \text{ mm}^2 \text{ MeV}^{-2}$, 1000 events per iteration. Starting from $e_0 = 22\,000 \text{ MeV}$, $a_0 = 3 \text{ mm}$, converging to the minimizer $e^* = 10\,000 \text{ MeV}$, $a^* = 2.3 \text{ mm}$.

Gradient-Based Optimization Results



350 gradient descent steps with $\lambda_e = 1$, $\lambda_a = 10^{-7} \text{ mm}^2 \text{ MeV}^{-2}$, 1000 events per iteration. Starting from $e_0 = 22,000 \text{ MeV}$, $a_0 = 3 \text{ mm}$, converging to the minimizer $e^* = 10,000 \text{ MeV}$, $a^* = 2.3 \text{ mm}$.

Paths are stochastic, of course.

Mathematical Challenges

- Is the variance of the derivative sufficiently low? Otherwise we'd have to average over too many events to get a trustworthy result. . .
- When we average the derivative where it exists, do we get the derivative of the averages?
- Can the AD derivatives be useful for optimization?

Mathematical Challenges

- Is the variance of the derivative sufficiently low? Otherwise we'd have to average over too many events to get a trustworthy result. . .
When disabling multiple scattering in an electromagnetic simulation, the variance is OK.
- When we average the derivative where it exists, do we get the derivative of the averages?
Not exactly, but up to 5 %, which is fine for optimization.
- Can the AD derivatives be useful for optimization?
Yes!

Summary

Preprint: Aehle et al., Optimization of Electromagnetic Shower Simulations using Pathwise Algorithmic Derivatives

Integrating AD capabilities in HEP detector simulations works in principle, and can be worthwhile

- from an application perspective: to understand how design parameters affect objective functions, and enable efficient gradient-based optimization; probably there are more applications than that;
- from a AD research perspective: Not much work done so far regarding AD of Monte-Carlo codes.

The community should always keep **alternative approaches** in mind:

- numerical differentiation
- gradient-free optimization

Contact Information

Max Ahle, `max.ahle@scicomp.uni-kl.de`,

Nicolas R. Gauger, `nicolas.gauger@scicomp.uni-kl.de`

Scientific Computing Group

University of Kaiserslautern-Landau (RPTU)