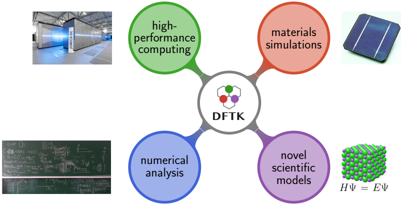
Automatic differentiation efforts in DFTK

Michael F. Herbst* and Niklas Schmitz†

*Applied and Computational Mathematics, RWTH Aachen University
†TU Berlin

7 October 2021

Slides: `https://michael-herbst.com/talks/2021.10.07_dftk_ad_update.pdf`



Contents

Modelling electronic structures

- Seek variational energy: $\min_P E(P)$
- But: Experiments can't measure energies!

- Changes in the energy are what is interesting
- Key question: How is the response to external perturbation?
- Examples:
  - Forces (response to atomic position shifts)
  - Dipole moment (response to electric field)
  - Elasticity (cross-response to lattice deformation)
  - ...
- Often directly measurable (or closely linked)

$\Rightarrow$ We care very much about derivatives

Modelling electronic structures

- Seek variational energy: $\min_P E(P)$
- But: Experiments can't measure energies!

- Changes in the energy are what is interesting
- Key question: How is the response to external perturbation?
- Examples:
  - Forces (response to atomic position shifts)
  - Dipole moment (response to electric field)
  - Elasticity (cross-response to lattice deformation)
  - ...
- Often directly measurable (or closely linked)

$\Rightarrow$ We care very much about derivatives

Modelling electronic structures

- Seek variational energy: $\min_P E(P)$
- But: Experiments can't measure energies!

- Changes in the energy are what is interesting
- Key question: How is the response to external perturbation?
- Examples:
  - Forces (response to atomic position shifts)
  - Dipole moment (response to electric field)
  - Elasticity (cross-response to lattice deformation)
  - ...
- Often directly measurable (or closely linked)

$\Rightarrow$ We care very much about derivatives

Need for automatic derivatives: Practical argument

- Many (many) models
- Need derivatives to judge usefulness of method
- Deriving / implementing analytic derivatives takes time ...
- ...so does fixing the bugs

- Even standard codes don't have all relevant derivatives
- Standard fallback: Finite differences

Need for AD: New and improved methods

- Any model building needs error control ...
- Error control needs derivatives (sensitivities)

- Data-driven model construction
- Scientific machine learning
- E.g. neural-network functionals / pseudos / ...

- Challenge: Requires *unusual* derivatives:
  - Density *vs.* XC parameters
  - Atomisation energy *vs.* pseudo parameters
  - ...

Property computation

- SCF fixed-point problem in density matrix P

$$0 = f(P, \lambda) = f_{\mathsf{FD}}\Big(H^\lambda(P)\Big) - P$$

  with
  - λ: Parameter of external perturbation
  - f_{FD}: Fermi-Dirac function
  - H^λ: Non-linear Kohn-Sham Hamiltonian
- Defines implicit function $P(\lambda)$ for density matrix
- Quantities of interest:

$$\frac{dQ(P)}{d\lambda} = \frac{\partial Q}{\partial \lambda} + \frac{\partial Q}{\partial P}\frac{\partial P}{\partial \lambda}$$

  - Forces: $Q = E, \quad \lambda = R$ (atomic displacement)
  - Polarisability: $Q =$ dipole moment, $\quad \lambda = \mathcal{E}$ (electric field)

Hellmann-Feynman theorem

$$\frac{dQ(P)}{d\lambda} = \frac{\partial Q}{\partial \lambda} + \frac{\partial Q}{\partial P}\frac{\partial P}{\partial \lambda}$$

- Special case of $Q = E$
- Recall $P_* = \text{argmin } E(P) \quad \Rightarrow \quad \left.\frac{\partial E}{\partial P}\right|_{P_*} = 0$
- Hellmann-Feynman theorem

$$\left.\frac{dE}{d\lambda}\right|_* = \left.\frac{\partial E}{\partial \lambda}\right|_*$$

- First energy derivatives are (comparatively) easy!

Response theory (1)

- If $Q \neq E$ we need $\frac{\partial P}{\partial \lambda}$
- Consider at $\lambda = \lambda_*$ and corresponding P_* and H_*:

$$
\begin{aligned}
0 &= \frac{\partial}{\partial \lambda} \left[f_{\text{FD}}\Big(H^\lambda(P)\Big) - P \right]\Big|_* \\
&= f'_{\text{FD}}(H_*) \cdot \frac{\partial H^\lambda}{\partial \lambda}\Big|_* + \frac{\partial P}{\partial \lambda}\Big|_* \cdot \frac{\partial}{\partial P} \left[f_{\text{FD}}\Big(H^\lambda(P)\Big) - P \right]\Big|_* \\
&= f'_{\text{FD}}(H_*) \cdot \frac{\partial H^\lambda}{\partial \lambda}\Big|_* + \frac{\partial P}{\partial \lambda}\Big|_* \cdot \left[f'_{\text{FD}}(H_*) \cdot \boldsymbol{K}^{\lambda_*}(P_*) - I \right]
\end{aligned}
$$

where $\boldsymbol{K}^{\lambda_*} = \dfrac{\partial H^{\lambda_*}}{\partial P}$

Response theory (2): Sternheimer equation

$$0 = f'_{\mathsf{FD}}(H_*) \cdot \left.\frac{\partial H^\lambda}{\partial \lambda}\right|_* + \left.\frac{\partial P}{\partial \lambda}\right|_* \cdot \left[f'_{\mathsf{FD}}(H_*) \cdot \boldsymbol{K}^{\lambda_*}(P_*) - I\right]$$

- Rearrange:

$$\left.\frac{\partial P}{\partial \lambda}\right|_* = -\left[f'_{\mathsf{FD}}(H_*)\boldsymbol{K}^{\lambda_*}(P_*) - I\right]^{-1} f'_{\mathsf{FD}}(H_*) \left.\frac{\partial H^\lambda}{\partial \lambda}\right|_*$$

$$= -\left[\boldsymbol{K}^{\lambda_*}(P_*) + \boldsymbol{\Omega}(H_*)\right]^{-1} \left.\frac{\partial H^\lambda}{\partial \lambda}\right|_*$$

where $\boldsymbol{\Omega}(H_*) = -\left(f'_{\mathsf{FD}}(H_*)\right)^{-1}$

- Sternheimer equation (implicit differentiation)

Example: Computing polarisabilities

- Homogeneous electric field $\lambda = \mathcal{E}$ along x-direction
- Cubic cell (length L_x)

- Hamiltonian $H^{\mathcal{E}}(P) = H_{\mathsf{DFT}}(P) - \mathcal{E}(x - L_x/2)$
- Perturbation $\frac{\partial H^{\mathcal{E}}}{\partial \mathcal{E}}\Big|_* = (x - L_x/2)$
- Dipole moment:

$$\mu(P) = \int_\Omega (x - L_x/2)\rho(r)\,\mathrm{d}r, \qquad \rho = \mathrm{diag}(P)$$

- Polarisability $\dfrac{d\mu}{d\mathcal{E}} = \dfrac{\partial \mu}{\partial P}\dfrac{\partial P}{\partial \mathcal{E}}$

1. Solve SCF $P_* = H^0(P_*)$ at zero field
2. Solve Sternheimer $\frac{\partial P}{\partial \mathcal{E}} = -[\boldsymbol{K} + \boldsymbol{\Omega}]^{-1} \frac{\partial H^{\mathcal{E}}}{\partial \mathcal{E}}$ *(implicit differentiation)*
3. Compute polarisability

Role of automatic differentiation

- Universal building blocks:
  - Primal pass: Solve SCF
  - f-rule: Solve Sternheimer

$\Rightarrow$ Code up once, use AD to take care of repetitive glue

- Adjoint-mode is goal:
  - Faster for larger number of parameters (neural net)
  - Support for higher derivatives
  - Sparsification techniques
- Adjoint-mode is feasible:
  - $\boldsymbol{K} + \boldsymbol{\Omega}$ is self-adjoint

  $\Rightarrow$ SCF r-rule: Adjoint-solve Sternheimer
- Let's look at things in practice ...

Contents

AD scenarios considered

- Forward-mode AD (ForwardDiff.jl)
  - Stresses via Hellmann-Feynman
  - Polarisability via implicit differentiation of SCF
- Adjoint-mode AD (Zygote.jl)
  - Stresses via Hellmann-Feynman
  - XC-functional gradients

Forward-mode AD with Hellman-Feynman

- For stresses $Q = E$, $\lambda = L$ (unit cell vectors)

$\Rightarrow$ Hellmann-Feynman applies

- Computing stresses:

$$\text{Stress} = \frac{1}{\det(\mathbf{L})} \left. \frac{\partial E\left[P_*, (I + \mathbf{M})\,\mathbf{L}\right]}{\partial \mathbf{M}} \right|_{\mathbf{M}=0}$$

- In julia code:

```
scfres = self_consistent_field(basis)  # Run SCF, get P*
     L = basis.model.lattice
stress = 1/det(L) * ForwardDiff.gradient(M -> recompute_energy(scfres, (I + M) * L),
                                         zero(L))
```

Stresses using ForwardDiff

ForwardDiff.jl workarounds & LIVE DEMO

Status of reverse-mode AD

ChainRules.jl workarounds & LIVE DEMO

AD scenarios considered

- Forward-mode AD (ForwardDiff.jl)
  - Stresses via Hellmann-Feynman **work**
  - Polarisability via implicit differentiation of SCF **works**
- Adjoint-mode AD (Zygote.jl)
  - Stresses via Hellmann-Feynman **work/WIP**
  - XC-functional gradients **WIP**

Strategies learned

- ForwardDiff.jl
  - ensure array-allocations can hold Dual numbers
  - custom overloads for non-Julia code (FFTW, spglib, ...)
- Zygote.jl with ChainRules.jl
  - avoid mutation
  - avoid indexing into large arrays
  - generating rrules from alternative primals
  - generating rrules from frules
  - general rrules for NLsolve.jl, IterativeSolvers.jl

Contents

Conclusion

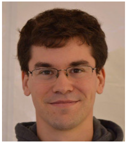
- We got the building blocks for 1st derivatives
- The challenge now is gluing it all together
- TODO:
    - Hide the details ...
    - Minimise code duplication
    - Optimise performance
    - Fix the details (symmetries, external libraries ...)
    - Higher derivatives?
- Happy for any input!

What next?

- Sensitivities:
  - Structural, alchemical, model parameters
  - Band gaps, atomisation energies, forces, geo-opt
- Data-driven design:
  - DFT models
  - Pseudopotentials
  - Tight-binding models

Acknowledgements

https://michael-herbst.com/talks/2021.10.07_dftk_ad_update.pdf



Antoine Levitt

Benjamin Stamm
Eric Cancès

all DFTK contributors

Inria
École des Ponts ParisTech
Summer of code
European Research Council
Established by the European Commission
julia
Applied and Computational Mathematics
RWTH AACHEN UNIVERSITY

Questions?

https://michael-herbst.com/talks/2021.10.07_dftk_ad_update.pdf

DFTK https://dftk.org

mfherbst
https://michael-herbst.com/blog
herbst@acom.rwth-aachen.de

niklasschmitz
niklasschmitz_
n.schmitz@tu-berlin.de