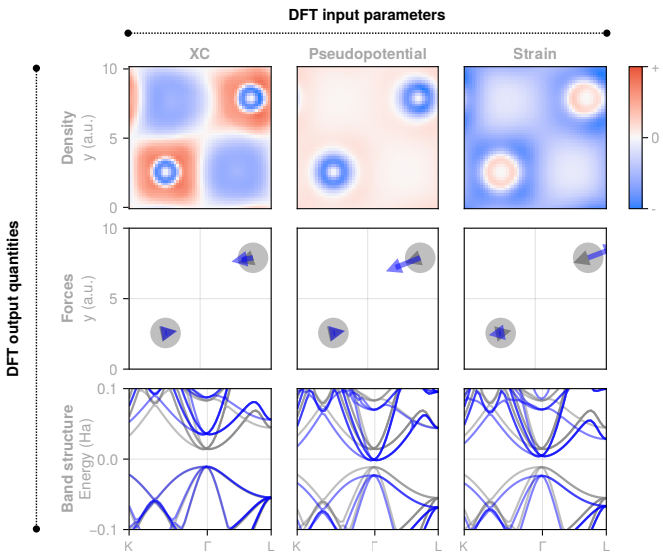
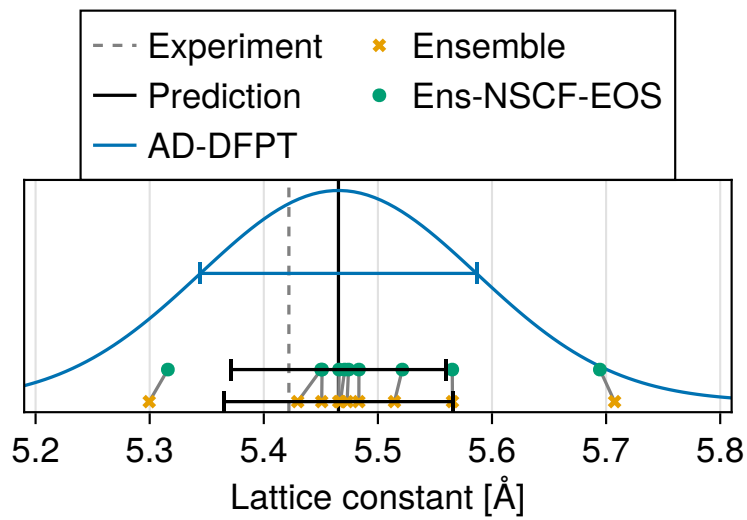
Algorithmic differentiation and error control with DFTK[1]

Bruno Ploumhans
Joint work with: Niklas F. Schmitz & Michael F. Herbst

Mathematics for Materials Modelling (matmat.org), EPFL

JuliaCon 2026, August 13, 2026

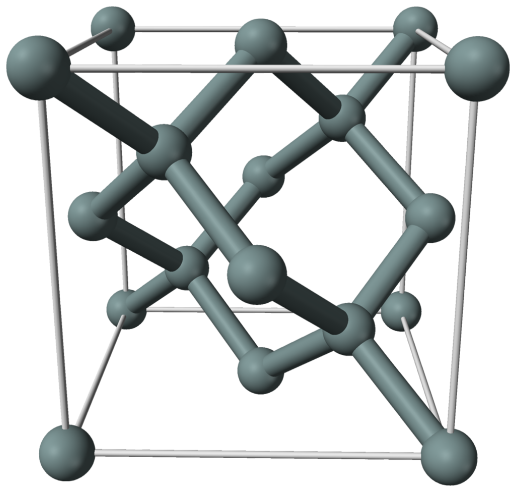




[1] N. F. Schmitz, B. Ploumhans, M. F. Herbst. *npj Comput. Mater.* (2026)

Context: Ab-initio modeling & materials discovery

Structure at atomic scale $\Longrightarrow$ properties
Applications: Novel batteries, solar cells, superconductors, ...



Workhorse method: Density-Functional Theory (DFT):
$\Longrightarrow$ Inputs: atomic species, positions, lattice
$\Longrightarrow$ Predicts electron density, total energy, forces, etc

Kohn-Sham Density Functional Theory (DFT)

Parameters θ (positions, lattice, model parameters,...)
Ground state: Find density matrix $P(\theta)$ minimizing energy $\mathcal{E}(\theta, P)$.
Self-consistent field (SCF) equations:

$$\begin{cases} H(\theta, P)\psi_n = \varepsilon_n \psi_n, & n = 1, \dots, N \\ P = \sum_{n=1}^{N} f(\varepsilon_n)\psi_n \psi_n^\dagger, \end{cases}$$

or compactly, writing f as a matrix function:

$$P = f(H(\theta, P)).$$

Plane-wave DFT: Large basis size N_b (avoid $\mathcal{O}(N_b^2)$ at all cost)
$\implies$ Matrix-free Hamiltonian (diagonal + low-rank + FFT convolutions)
$\implies$ Iterative solvers working only on N orbitals are standard here
inner eigensolver (eg LOBPCG), outer fixed point (eg Anderson)

Kohn-Sham Density Functional Theory (DFT)

Parameters θ (positions, lattice, model parameters,...)
Ground state: Find density matrix $P(\theta)$ minimizing energy $\mathcal{E}(\theta, P)$.
Self-consistent field (SCF) equations:

$$\begin{cases} H(\theta, P)\psi_n = \varepsilon_n \psi_n, & n = 1, \dots, N \\ P = \sum_{n=1}^{N} f(\varepsilon_n)\psi_n \psi_n^\dagger, \end{cases}$$

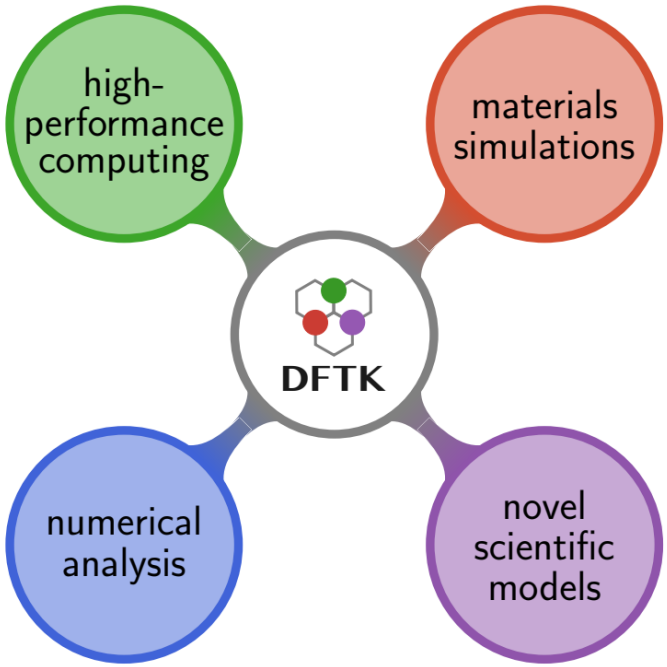
or compactly, writing f as a matrix function:

$$P = f(H(\theta, P)).$$

Plane-wave DFT: Large basis size N_b (avoid $\mathcal{O}(N_b^2)$ at all cost)
$\implies$ Matrix-free Hamiltonian (diagonal + low-rank + FFT convolutions)
$\implies$ Iterative solvers working only on N orbitals are standard here
inner eigensolver (eg LOBPCG), outer fixed point (eg Anderson)

Density-Functional ToolKit – `https://dftk.org`

- Plane-wave DFT code for cross-disciplinary research
- Implemented in Julia
- 10 000 lines, hackable & easy to extend
- HPC support: MPI, Nvidia & AMD GPUs
- Unique features: custom models, AD



Example: Silicon forces and stresses

```
using DFTK
using PseudoPotentialData

a = 10.26  # Silicon lattice constant in Bohr
lattice = a / 2 * [[0 1 1.];
                   [1 0 1.];
                   [1 1 0.]]
Si = ElementPsp(:Si, PseudoFamily("dojo.nc.sr.lda.v0_4_1.standard.upf"))
atoms     = [Si, Si]
positions = [ones(3)/8, -ones(3)/8]

model = model_DFT(lattice, atoms, positions; functionals=LDA())
basis = PlaneWaveBasis(model; Ecut=15, kgrid=[4, 4, 4])

scfres = self_consistent_field(basis, tol=1e-8)

forces = compute_forces_cart(scfres)
stresses = compute_stresses_cart(scfres)
```

Example: Silicon forces and stresses

```
using DFTK
using PseudoPotentialData
```

Setup problem

```
a = 10.26  # Silicon lattice constant in Bohr
lattice = a / 2 * [[0 1 1.];
                   [1 0 1.];
                   [1 1 0.]]
Si = ElementPsp(:Si, PseudoFamily("dojo.nc.sr.lda.v0_4_1.standard.upf"))
atoms     = [Si, Si]
positions = [ones(3)/8, -ones(3)/8]

model = model_DFT(lattice, atoms, positions; functionals=LDA())
basis = PlaneWaveBasis(model; Ecut=15, kgrid=[4, 4, 4])
```

```
scfres = self_consistent_field(basis, tol=1e-8)
```

Solve for P

```
forces = compute_forces_cart(scfres)
stresses = compute_stresses_cart(scfres)
```

Postprocess using optimal P

Many physical properties are DFT derivatives

Quantity	Definition
Force-constant matrix (phonons)	$K_{s\alpha,t\beta} = -\frac{\partial F_{t\beta}}{\partial u_{s\alpha}}$
Dielectric susceptibility	$\bar{\chi}_{\alpha\beta} = \frac{\partial P^{\mathrm{el}}_{\beta}}{\partial \mathcal{E}_{\alpha}}$
Elastic tensor	$\bar{C}_{jk} = \frac{\partial \sigma_k}{\partial \eta_j}$
Born effective charges	$Z^*_{s\alpha\beta} = \frac{\partial F_{s\alpha}}{\partial \mathcal{E}_{\beta}}$
Force-response internal strain	$\Lambda_{s\alpha j} = \frac{\partial F_{s\alpha}}{\partial \eta_j}$
Piezoelectric tensor	$\bar{e}_{\alpha j} = -\frac{\partial \sigma_j}{\partial \mathcal{E}_{\alpha}}$

Quantities: forces F, electronic polarization P^{el}, stress σ
Differentiated wrt.: atomic displacement $u_{s\alpha}$, electric field $\mathcal{E}_{\alpha}$, strain η_j

How to compute them?

Density Functional Perturbation Theory (DFPT)[4]

- Implicit differentiation of SCF equations at solution P:

$$P = f(H(\theta, P)) \implies \frac{\partial P}{\partial \theta} = (1 - \chi_0 K)^{-1} \chi_0 \frac{\partial H}{\partial \theta},$$

$\chi_0 := \frac{\partial f}{\partial H}$ the independent susceptibility of the density matrix, $K := \frac{\partial H}{\partial P}$ due to nonlinear Hartree and XC terms.

- Solve for $\frac{\partial P}{\partial \theta}$ (efficient matrix-free solvers on N orbitals)
  (again nested: every χ_0 solves eigenvector PT by CG, outer solver is GMRES)
  improved: Schur-complement preconditioning[2], inexact GMRES[3], ...
- Hand-write $\frac{\partial H}{\partial \theta}$ + feed to core DFPT solver + hand-write assembly

$$A(\theta) = \mathcal{A}(\theta, P(\theta)) \longrightarrow \frac{\partial A}{\partial \theta} = \frac{\partial \mathcal{A}}{\partial \theta} + \frac{\partial \mathcal{A}}{\partial P} \frac{\partial P}{\partial \theta}$$

[2] Cancès, Herbst, Kemlin, Levitt, Stamm. *Lett. in Math. Phys.* (2023)
[3] Herbst, Sun. *SIAM J. Sci. Comput.* (2026)
[4] Baroni, de Gironcoli, Dal Corso, Giannozzi. *Rev. Mod. Phys.* (2001)

Density Functional Perturbation Theory (DFPT)[4]

- Implicit differentiation of SCF equations at solution P:

$$P = f(H(\theta, P)) \implies \frac{\partial P}{\partial \theta} = (1 - \chi_0 K)^{-1} \chi_0 \frac{\partial H}{\partial \theta},$$

$\chi_0 := \frac{\partial f}{\partial H}$ the independent susceptibility of the density matrix, $K := \frac{\partial H}{\partial P}$ due to nonlinear Hartree and XC terms.

- Solve for $\frac{\partial P}{\partial \theta}$ (efficient matrix-free solvers on N orbitals)
  (again nested: every χ_0 solves eigenvector PT by CG, outer solver is GMRES)
  improved: Schur-complement preconditioning[2], inexact GMRES[3], ...
- Hand-write $\frac{\partial H}{\partial \theta}$ + feed to core DFPT solver + hand-write assembly

$$A(\theta) = \mathcal{A}(\theta, P(\theta)) \quad \longrightarrow \quad \frac{\partial A}{\partial \theta} = \frac{\partial \mathcal{A}}{\partial \theta} + \frac{\partial \mathcal{A}}{\partial P} \frac{\partial P}{\partial \theta}$$

---

[2] Cancès, Herbst, Kemlin, Levitt, Stamm. *Lett. in Math. Phys.* (2023)
[3] Herbst, Sun. *SIAM J. Sci. Comput.* (2026)
[4] Baroni, de Gironcoli, Dal Corso, Giannozzi. *Rev. Mod. Phys.* (2001)

Density Functional Perturbation Theory (DFPT)[4]

- Implicit differentiation of SCF equations at solution P:

$$P = f(H(\theta, P)) \implies \frac{\partial P}{\partial \theta} = \boxed{(1 - \chi_0 K)^{-1} \chi_0} \frac{\partial H}{\partial \theta},$$

Needs careful hand writing

$\chi_0 := \frac{\partial f}{\partial H}$ the independent susceptibility of the density matrix, $K := \frac{\partial H}{\partial P}$ due to nonlinear Hartree and XC terms.

- Solve for $\frac{\partial P}{\partial \theta}$ (efficient matrix-free solvers on N orbitals)
  (again nested: every χ_0 solves eigenvector PT by CG, outer solver is GMRES)
  improved: Schur-complement preconditioning[2], inexact GMRES[3], ...
- Hand-write $\frac{\partial H}{\partial \theta}$ + feed to core DFPT solver + hand-write assembly

$$A(\theta) = \mathcal{A}(\theta, P(\theta)) \quad \longrightarrow \quad \frac{\partial A}{\partial \theta} = \frac{\partial \mathcal{A}}{\partial \theta} + \frac{\partial \mathcal{A}}{\partial P} \frac{\partial P}{\partial \theta}$$

---

[2] Cancès, Herbst, Kemlin, Levitt, Stamm. *Lett. in Math. Phys.* (2023)
[3] Herbst, Sun. *SIAM J. Sci. Comput.* (2026)
[4] Baroni, de Gironcoli, Dal Corso, Giannozzi. *Rev. Mod. Phys.* (2001)

Density Functional Perturbation Theory (DFPT)[4]

- Implicit differentiation of SCF equations at solution P:

Needs careful hand writing

$$P = f(H(\theta, P)) \implies \frac{\partial P}{\partial \theta} = \boxed{(1 - \chi_0 K)^{-1} \chi_0} \frac{\partial H}{\partial \theta},$$

$\chi_0 := \frac{\partial f}{\partial H}$ the independent susceptibility of the density matrix, $K := \frac{\partial H}{\partial P}$ due to nonlinear Hartree and XC terms.

- Solve for $\frac{\partial P}{\partial \theta}$ (efficient matrix-free solvers on N orbitals)
  (again nested: every χ_0 solves eigenvector PT by CG, outer solver is GMRES)
  improved: Schur-complement preconditioning[2], inexact GMRES[3], ...
- Hand-write $\frac{\partial H}{\partial \theta}$ + feed to core DFPT solver + hand-write assembly

$$A(\theta) = \mathcal{A}(\theta, P(\theta)) \quad \longrightarrow \quad \frac{\partial A}{\partial \theta} = \boxed{\frac{\partial \mathcal{A}}{\partial \theta} + \frac{\partial \mathcal{A}}{\partial P} \frac{\partial P}{\partial \theta}}$$

Mechanical and tedious

---

[2] Cancès, Herbst, Kemlin, Levitt, Stamm. *Lett. in Math. Phys.* (2023)
[3] Herbst, Sun. *SIAM J. Sci. Comput.* (2026)
[4] Baroni, de Gironcoli, Dal Corso, Giannozzi. *Rev. Mod. Phys.* (2001)

Differentiate setup&postprocess with AD, solve with DFPT

```
using DFTK
using PseudoPotentialData
```

Setup problem

```
a = 10.26  # Silicon lattice constant in Bohr
lattice = a / 2 * [[0 1 1.];
                   [1 0 1.];
                   [1 1 0.]]
Si = ElementPsp(:Si, PseudoFamily("dojo.nc.sr.lda.v0_4_1.standard.upf"))
atoms     = [Si, Si]
positions = [ones(3)/8, -ones(3)/8]

model = model_DFT(lattice, atoms, positions; functionals=LDA())
basis = PlaneWaveBasis(model; Ecut=15, kgrid=[4, 4, 4])
```

```
scfres = self_consistent_field(basis, tol=1e-8)
```

Solve for P

```
forces = compute_forces_cart(scfres)
stresses = compute_stresses_cart(scfres)
```

Postprocess

Differentiate setup&postprocess with AD, solve with DFPT

```
using DFTK
using PseudoPotentialData
```

Setup problem → AD

```
a = 10.26  # Silicon lattice constant in Bohr
lattice = a / 2 * [[0 1 1.];
                   [1 0 1.];
                   [1 1 0.]]
Si = ElementPsp(:Si, PseudoFamily("dojo.nc.sr.lda.v0_4_1.standard.upf"))
atoms     = [Si, Si]
positions = [ones(3)/8, -ones(3)/8]

model = model_DFT(lattice, atoms, positions; functionals=LDA())
basis = PlaneWaveBasis(model; Ecut=15, kgrid=[4, 4, 4])
```

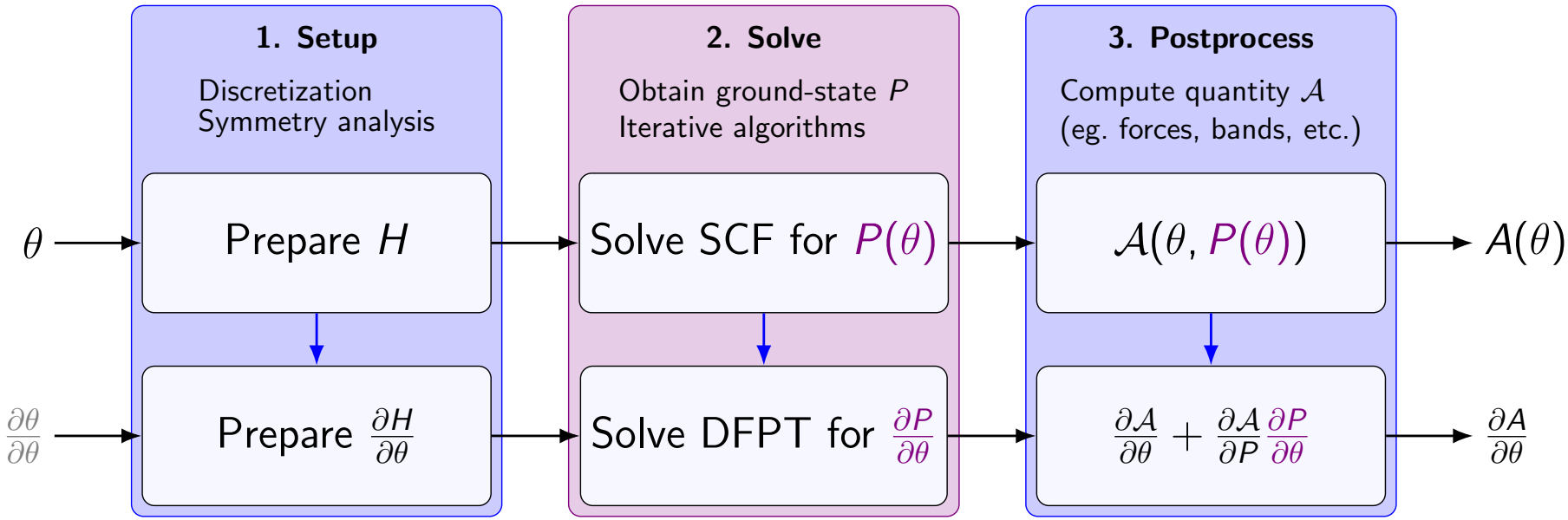
```
scfres = self_consistent_field(basis, tol=1e-8)
```

Solve for P → DFPT

```
forces = compute_forces_cart(scfres)
stresses = compute_stresses_cart(scfres)
```

Postprocess → AD

End-to-end DFT derivatives with AD-DFPT[5]



- Setup and postprocessing handled by general-purpose AD system (ForwardDiff.jl)
- We **impose** custom AD rule: Converge SCF, then DFPT solver.

[5] N. F. Schmitz, B. Ploumhans, M. F. Herbst. *npj Comput. Mater.* (2026)

Combinatorial derivatives: many outputs wrt. many inputs

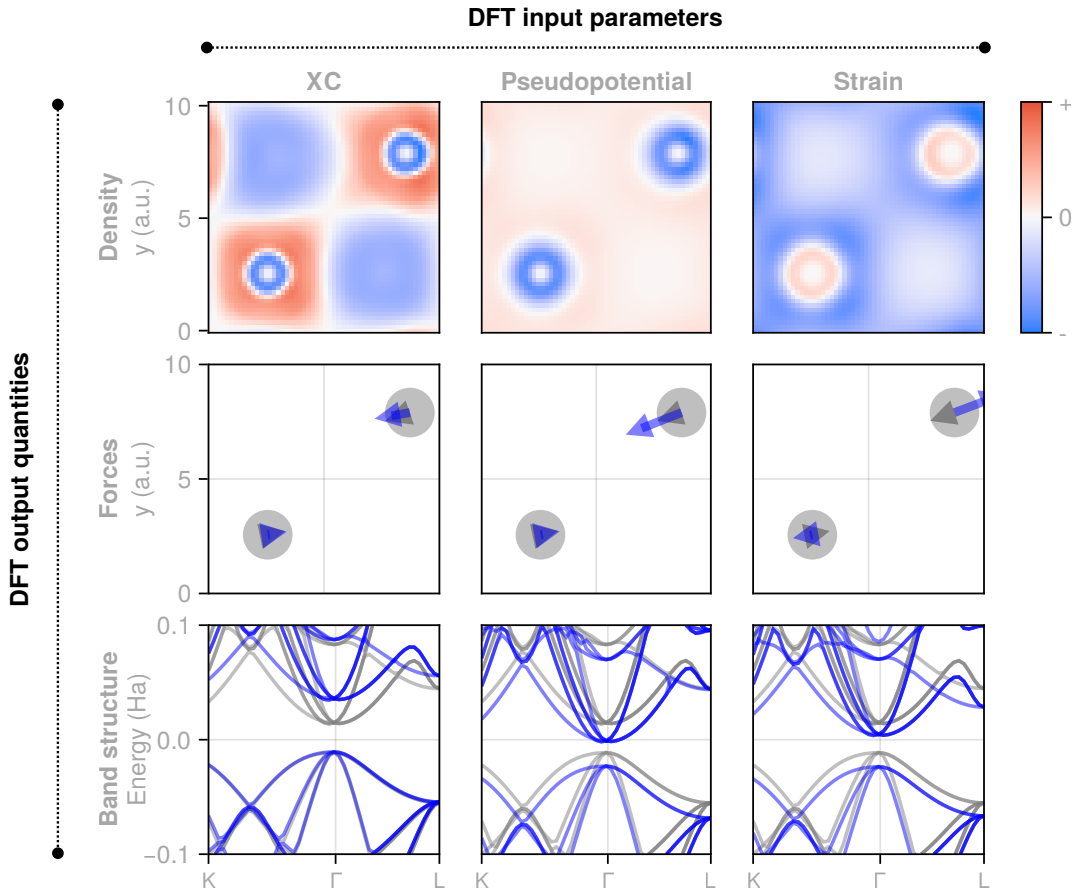


Figure: Systematic DFT derivatives from AD-DFPT[6] in DFTK.jl

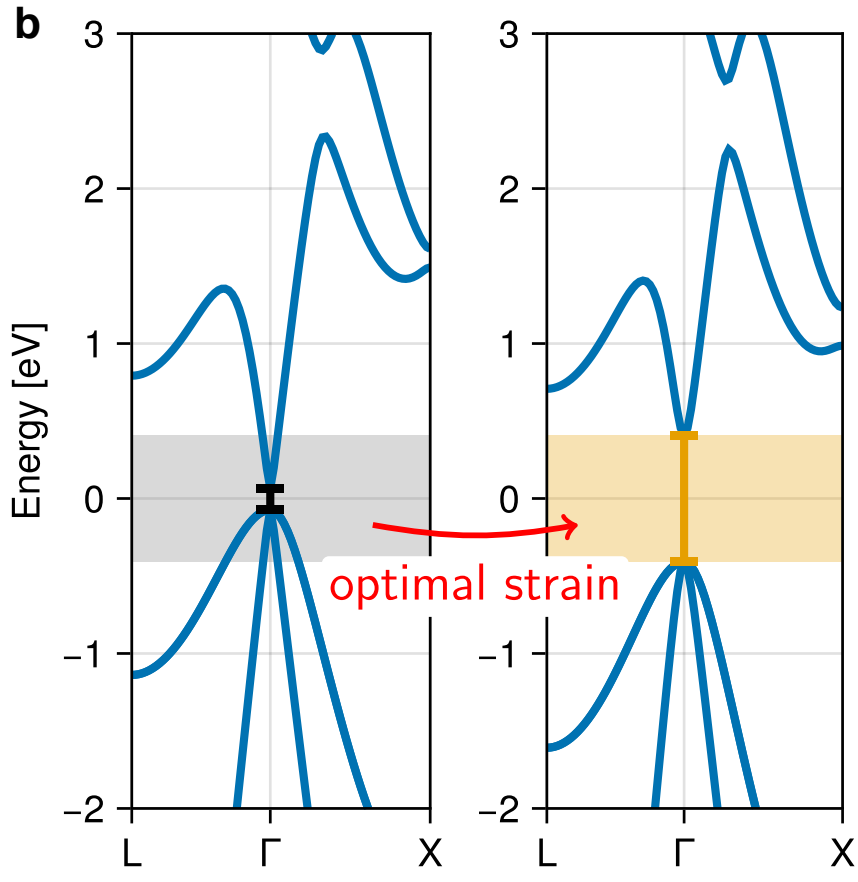
[6] N. F. Schmitz, B. Ploumhans, M. F. Herbst. *npj Comput. Mater.* (2026)

Novel derivatives: Band gap inverse design by strain

a

```
using DFTK, PseudoPotentialData, AtomsIO
using ForwardDiff, DifferentiationInterface, Optim

system = load_system("mp-2534-GaAs.cif")
pseudopotentials = PseudoFamily("dojo.nc.sr.pbe.v0_4_1.standard.upf")
model0 = model_DFT(system; functionals=PBE(), pseudopotentials,
                   smearing=Smearing.Gaussian(), temperature=1e-3)

function strain_bandgap(η)
    model = Model(model0; lattice=(1 + η) * model0.lattice)
    basis = PlaneWaveBasis(model; Ecut=42, kgrid=(8, 8, 8))
    scfres = self_consistent_field(basis; tol=1e-6)
    eigenvalues_Γ = scfres.eigenvalues[1]
    ε_vbm = maximum(eigenvalues_Γ[eigenvalues_Γ .≤ scfres.εF])
    ε_cbm = minimum(eigenvalues_Γ[eigenvalues_Γ .> scfres.εF])
    ε_cbm - ε_vbm
end

η0 = [0.0]              # Initialize with zero strain (at equilibrium)
bandgap_target = 0.03  # Target band gap in [Ha]
bandgap_loss(η) = (bandgap_target - strain_bandgap(η[1]))^2
res = Optim.optimize(bandgap_loss, η0, BFGS(),
                     Optim.Options(; iterations=5, x_abstol=1e-3);
                     autodiff=AutoForwardDiff())
println("Optimized design strain η = ", res.minimizer)
```

b



- Easy: This is a full code example, just write a loss function with DFT
- `Optim.optimize` requests AD gradients, which triggers AD-DFPT

Note: Full code for materials design

```julia
using DFTK, PseudoPotentialData, AtomsIO
using ForwardDiff, DifferentiationInterface, Optim

system = load_system("mp-2534-GaAs.cif")
pseudopotentials = PseudoFamily("dojo.nc.sr.pbe.v0_4_1.standard.upf")
model0 = model_DFT(system; functionals=PBE(), pseudopotentials,
                   smearing=Smearing.Gaussian(), temperature=1e-3)

function strain_bandgap(η)
    model = Model(model0; lattice=(1 + η) * model0.lattice)
    basis = PlaneWaveBasis(model; Ecut=42, kgrid=(8, 8, 8))
    scfres = self_consistent_field(basis; tol=1e-6)
    eigenvalues_Γ = scfres.eigenvalues[1]
    ε_vbm = maximum(eigenvalues_Γ[eigenvalues_Γ .≤ scfres.εF])
    ε_cbm = minimum(eigenvalues_Γ[eigenvalues_Γ .> scfres.εF])
    ε_cbm - ε_vbm
end

η0 = [0.0]              # Initialize with zero strain (at equilibrium)
bandgap_target = 0.03  # Target band gap in [Ha]
bandgap_loss(η) = (bandgap_target - strain_bandgap(η[1]))^2
res = Optim.optimize(bandgap_loss, η0, BFGS(),
                     Optim.Options(; iterations=5, x_abstol=1e-3);
                     autodiff=AutoForwardDiff())
println("Optimized design strain η = ", res.minimizer)
```

Nov

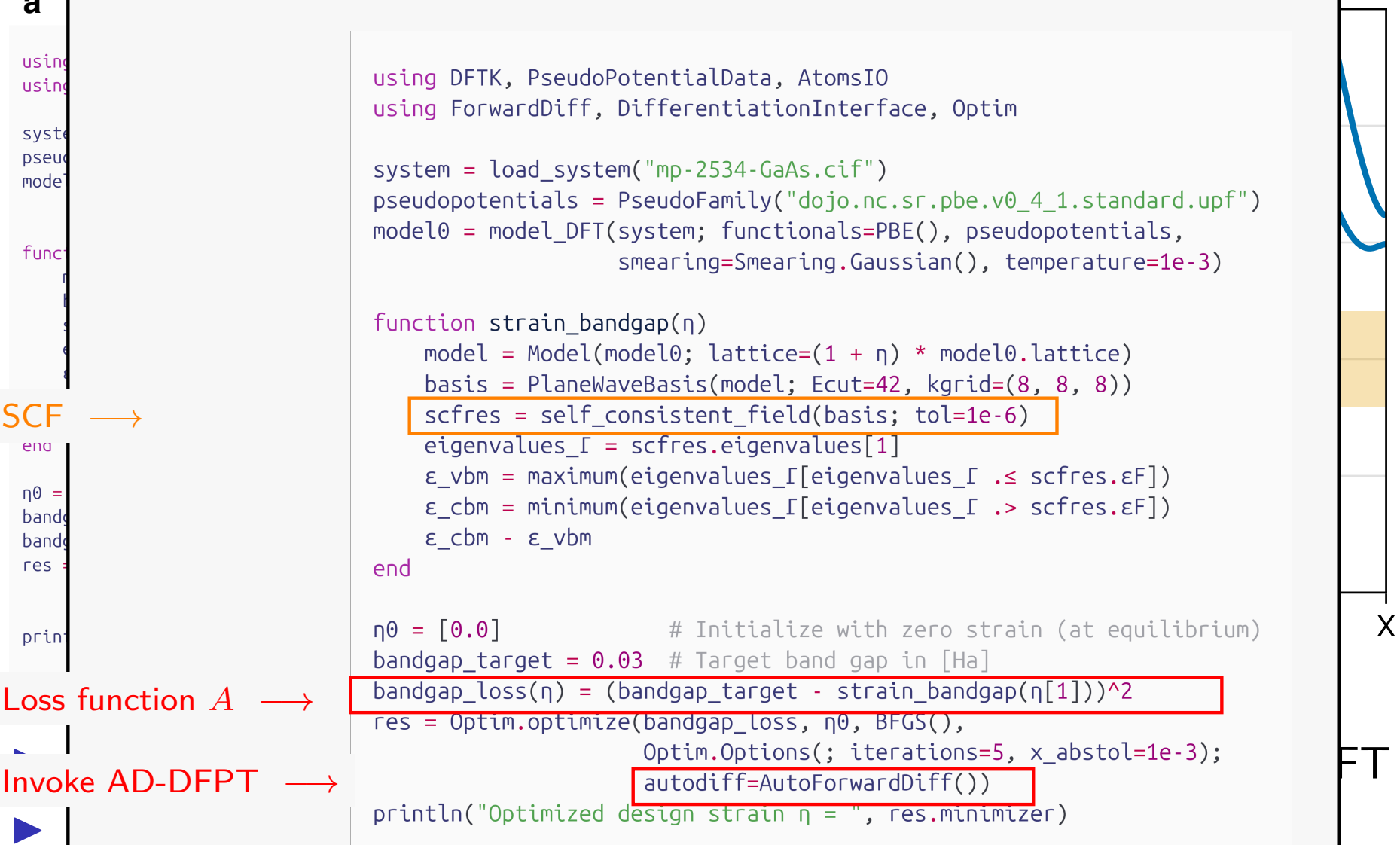
Note: Full code for materials design

```julia
using DFTK, PseudoPotentialData, AtomsIO
using ForwardDiff, DifferentiationInterface, Optim

system = load_system("mp-2534-GaAs.cif")
pseudopotentials = PseudoFamily("dojo.nc.sr.pbe.v0_4_1.standard.upf")
model0 = model_DFT(system; functionals=PBE(), pseudopotentials,
                   smearing=Smearing.Gaussian(), temperature=1e-3)

function strain_bandgap(η)
    model = Model(model0; lattice=(1 + η) * model0.lattice)
    basis = PlaneWaveBasis(model; Ecut=42, kgrid=(8, 8, 8))
    scfres = self_consistent_field(basis; tol=1e-6)
    eigenvalues_Γ = scfres.eigenvalues[1]
    ε_vbm = maximum(eigenvalues_Γ[eigenvalues_Γ .≤ scfres.εF])
    ε_cbm = minimum(eigenvalues_Γ[eigenvalues_Γ .> scfres.εF])
    ε_cbm - ε_vbm
end

η0 = [0.0]              # Initialize with zero strain (at equilibrium)
bandgap_target = 0.03  # Target band gap in [Ha]
bandgap_loss(η) = (bandgap_target - strain_bandgap(η[1]))^2
res = Optim.optimize(bandgap_loss, η0, BFGS(),
                     Optim.Options(; iterations=5, x_abstol=1e-3);
                     autodiff=AutoForwardDiff())
println("Optimized design strain η = ", res.minimizer)
```

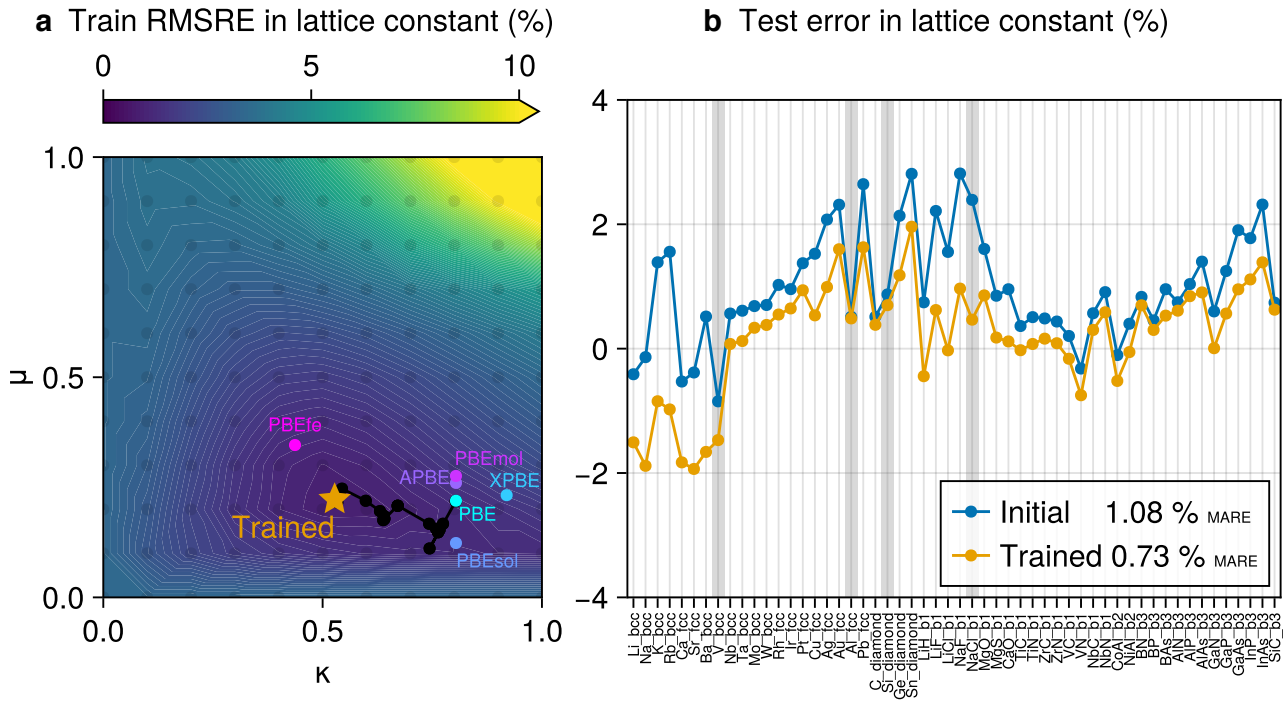
SCF ⟶

Loss function A ⟶

Invoke AD-DFPT ⟶



Novel derivatives: Learning exchange-correlation parameters[7]



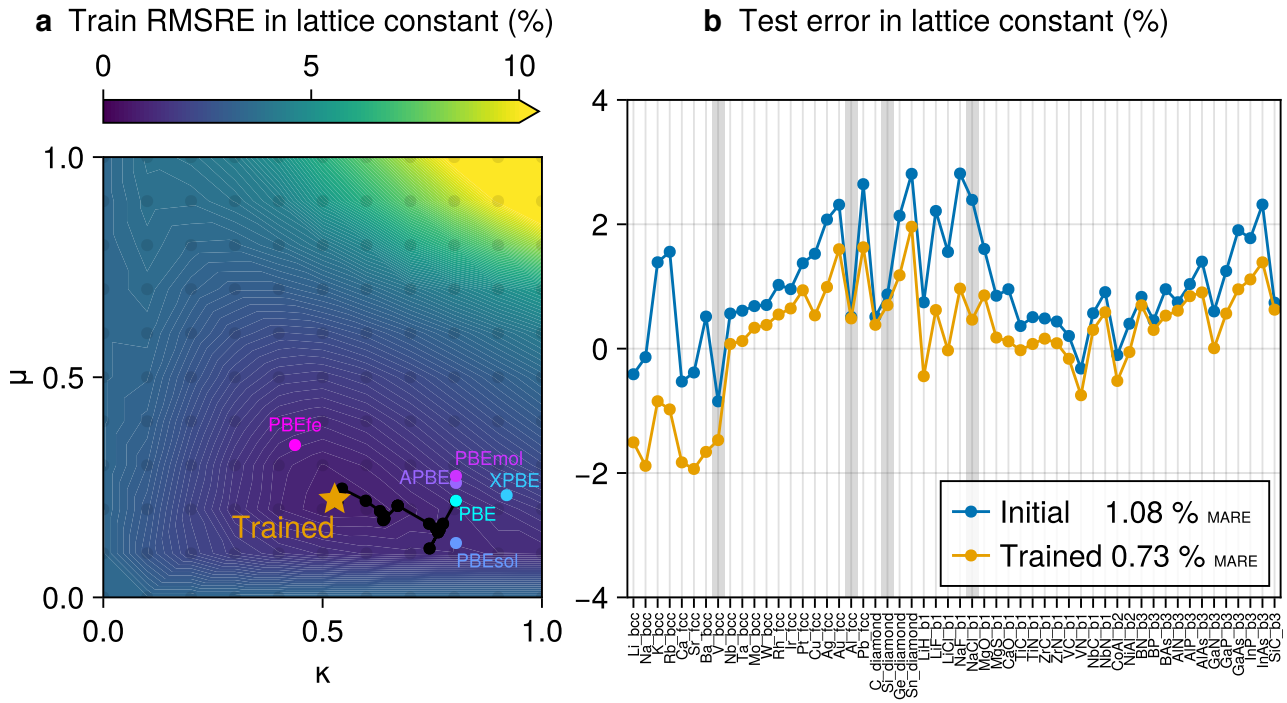
- Optimize a DFT functional (2 parameters θ) on C = Si, Al, V, NaCl

$$L_{\text{xc}}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{a^\star(C_i,\theta) - a_i^{\text{expt}}}{a_i^{\text{expt}}}\right)^2, \quad a^\star(C,\theta) = \arg\min_a \min_P \mathcal{E}(C,a,\theta,P)$$

  - **Nested iterative methods** (eigensolver, SCF, lattice optimization)
  - Unusual derivative $\dfrac{\partial a^\star}{\partial \theta} = -\left(\dfrac{\partial^2 E}{\partial a^2}\right)^{-1} \dfrac{\partial^2 E}{\partial \theta \partial a}$

[7] N. F. Schmitz, B. Ploumhans, M. F. Herbst. *npj Comput. Mater.* (2026)

Novel derivatives: Learning exchange-correlation parameters[7]



- Optimize a DFT functional (2 parameters θ) on C = Si, Al, V, NaCl

$$L_{\text{xc}}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{a^\star(C_i,\theta) - a_i^{\text{expt}}}{a_i^{\text{expt}}}\right)^2, \quad a^\star(C,\theta) = \arg\min_a \min_P \mathcal{E}(C,a,\theta,P)$$

  - **Nested iterative methods** (eigensolver, SCF, lattice optimization)
  - Unusual derivative $\dfrac{\partial a^\star}{\partial \theta} = -\left(\dfrac{\partial^2 E}{\partial a^2}\right)^{-1} \dfrac{\partial^2 E}{\partial \theta \partial a}$

[7] N. F. Schmitz, B. Ploumhans, M. F. Herbst. *npj Comput. Mater.* (2026)

Novel derivatives: Forward uncertainty quantification[8]

- Given parameter uncertainty $\theta \sim \mathcal{N}(\theta_0, \Sigma)$, propagate to lattice constant $a^\star(\theta)$
- Linearization around mean θ_0:

$$a^\star(\theta) \approx a^\star(\theta_0) + J(\theta - \theta_0)$$

$$\implies \mathcal{N}(a^\star(\theta_0), J\Sigma J^\top) \text{ with } J = \frac{\partial a^\star}{\partial \theta}|_{\theta_0} \text{ from AD-DFPT}$$

- only 1 relaxation + 1 derivative, no N-fold ensemble required
- general

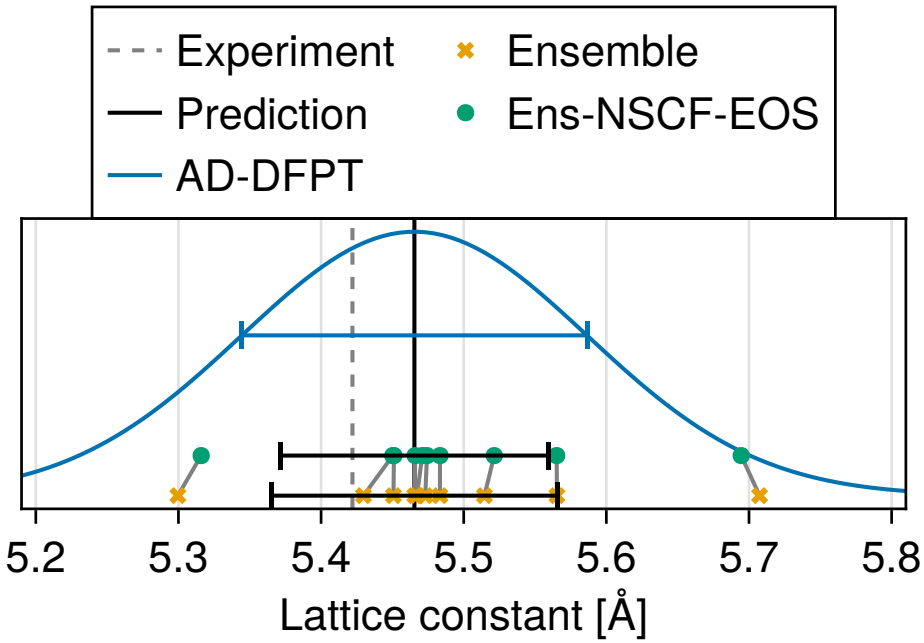


Figure: XC uncertainty (here BEEF [a]) propagated to lattice constant on Si (AD-DFPT and Ensemble baselines)

[a] Mortensen, Kaasbjerg, Frederiksen, Nørskov, Sethna, Jacobsen. (2005). Phys. Rev. Lett., 95, 216401.

[8] N. F. Schmitz, B. Ploumhans, M. F. Herbst. *npj Comput. Mater.* (2026)

Take-home messages

- Derivative of an iterative solver is another iterative solver.
  - Fixed-point iteration $\rightarrow$ iterative linear solve (GMRES)
  - Iterative diagonalization $\rightarrow$ **carefully tuned** iterative linear solve (CG)
  - SCF $\rightarrow$ DFPT (GMRES around CG)
  - **Might be better to write it by hand!**
- AD-DFPT: Embeds DFPT into general-purpose AD system (ForwardDiff.jl)
- This is **forward-mode AD**: scales with number of input parameters (just like DFPT)

Take-home messages

- Derivative of an iterative solver is another iterative solver.
- AD-DFPT: Embeds DFPT into general-purpose AD system (ForwardDiff.jl)
  - **End-to-end** derivatives of DFT quantities with respect to any parameters
  - Standard use case: properties
  - **Novel use cases**: optimization, uncertainty propagation, ...
- This is **forward-mode AD**: scales with number of input parameters (just like DFPT)

Take-home messages

- Derivative of an iterative solver is another iterative solver.
- AD-DFPT: Embeds DFPT into general-purpose AD system (ForwardDiff.jl)
- This is **forward-mode AD**: scales with number of input parameters (just like DFPT)
  - Can do a lot already
  - **Reverse mode** is in the works (high-dimensional optimization, ML-XC training, ...)

Thank you!

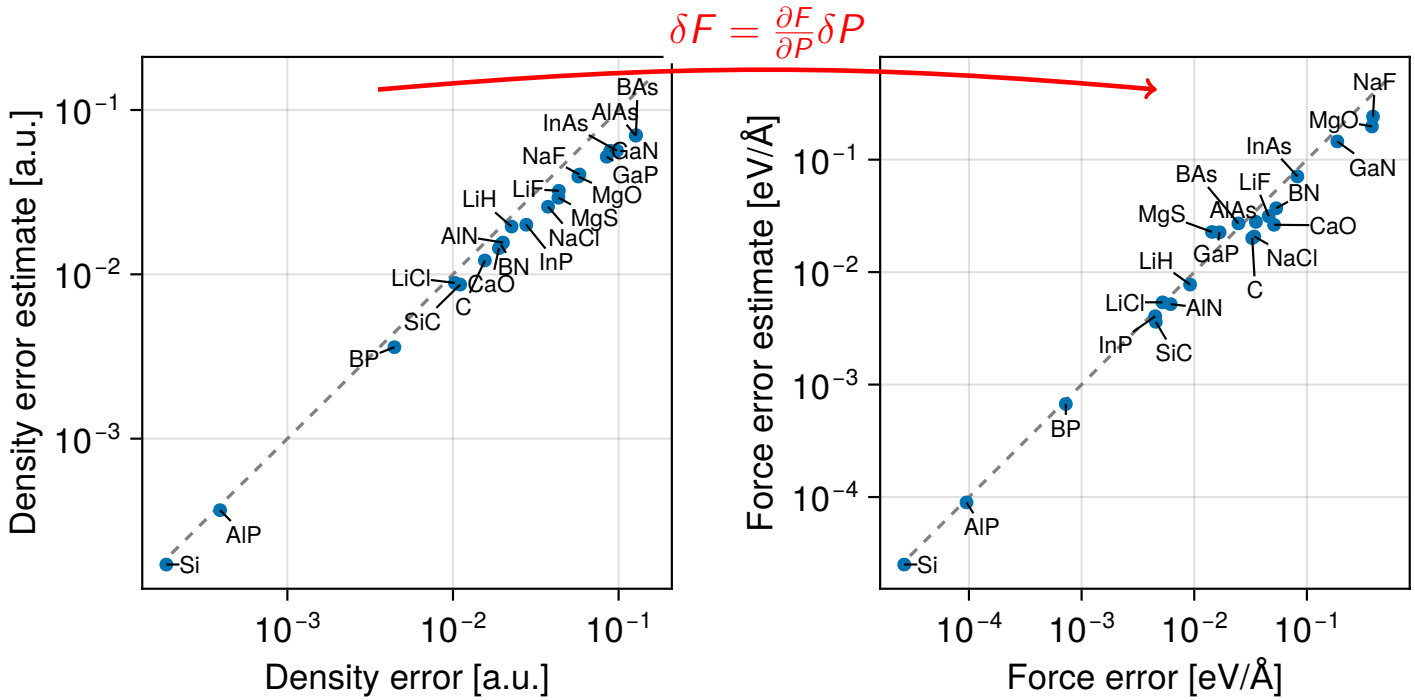
Thank you! Questions?

- N. F. Schmitz, B. Ploumhans, M. F. Herbst. *npj Comput. Mater.* (2026). `https://doi.org/10.1038/s41524-025-01880-3`
- Notebook: `https://showcases.matmat.org/2025/autodiff_dftk.html`
- All examples: `https://github.com/niklasschmitz/ad-dfpt`
- Framework: DFTK.jl (`https://dftk.org`)

Thanks to Niklas for the original version of the slides!

Appendix

Plane-wave error estimation[10]



Setup: Use a uniform low $E_{\text{cut}} = 20$ Ha across solids.
Perturbative density error estimates δP following method of[9].
$\implies$ propagate to quantities of interest by AD (here forces).

[9] Cancès, Dusson, Kemlin, Levitt. *SIAM J. Sci. Comp.* (2022)
[10] N. F. Schmitz, B. Ploumhans, M. F. Herbst. *npj Comput. Mater.* (2026)

Parameters in DFT

- Model (exchange-correlation, pseudopotentials, ...)
- Numerics (basis E_{cut}, $\mathbf{k}$-points, SCF tolerance, ...)

Tasks with gradients

- structure optimization, ...
- learning exchange-correlation, ...
- uncertainty propagation

Our new tool: Algorithmic Differentiation for plane-wave DFT (AD-DFPT)[a]

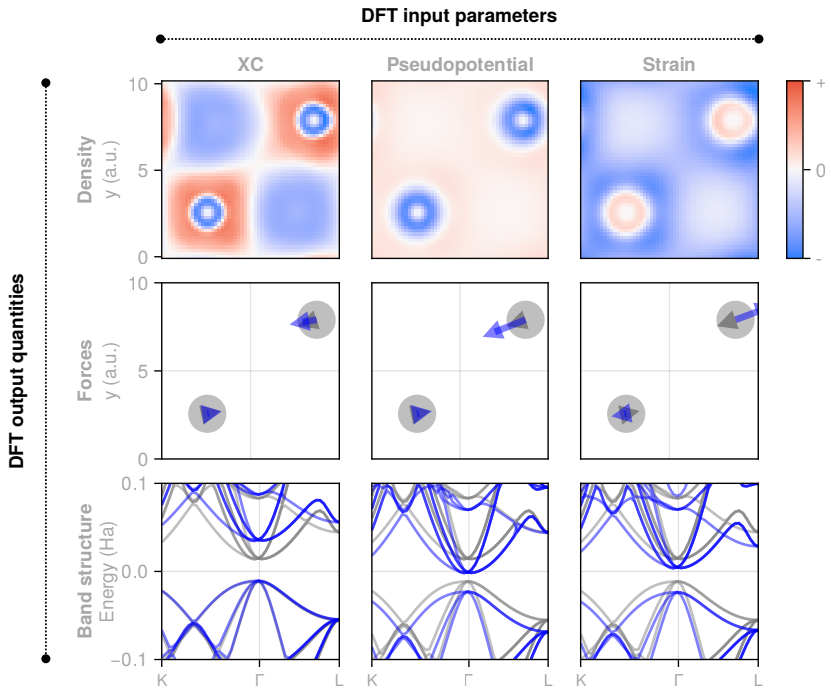
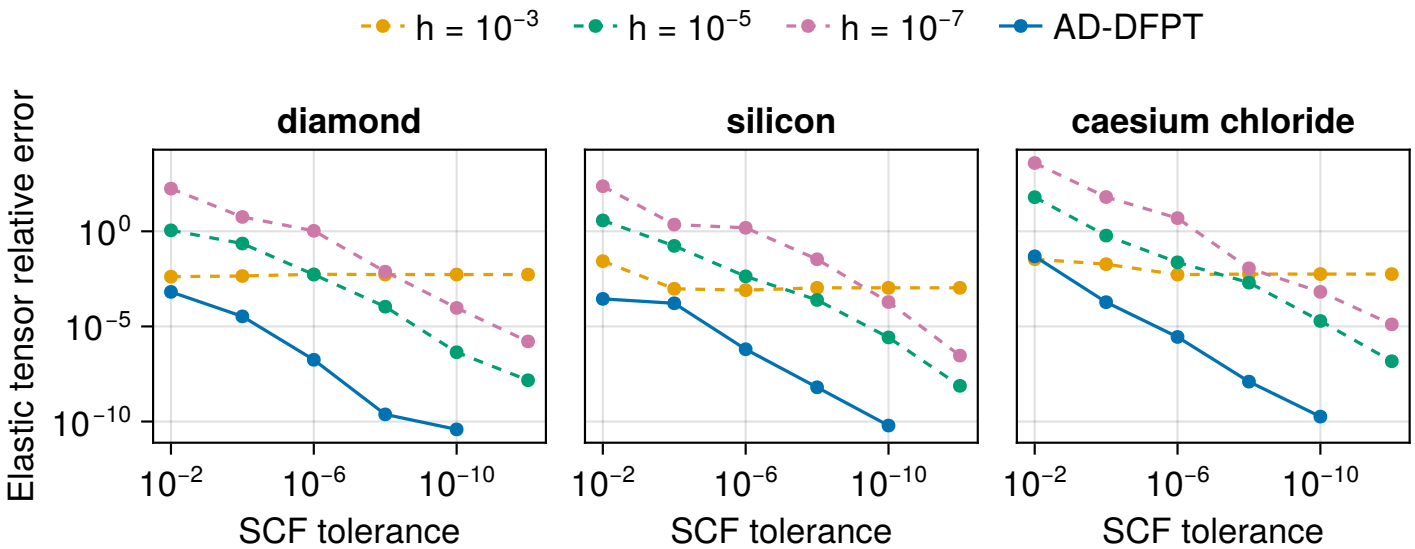


Figure: Derivatives from AD-DFPT in DFTK.jl (`dftk.org`)

[a] N. F. Schmitz, B. Ploumhans, M. F. Herbst. *npj Comput. Mater.* (2026).

Precision: Elasticity



- Accuracy better than finite differences, even at loose SCF tolerance
- Implementation simply by defining[11] (strain η)
  - Stress $\sigma(\eta) = \frac{1}{V(\eta)} \frac{\partial \mathcal{E}}{\partial \eta}$, "Hellmann-Feynman Thm."
  - Elastic tensor $C = \frac{\partial \sigma}{\partial \eta}|_{\eta^\star}$, AD-DFPT
- AD-DFPT combines precision and simplicity.

[11] `https://docs.dftk.org/stable/examples/elastic_constants/`

Plane-wave discretization

Plane-waves are the Fourier modes of the periodic domain:

$$e_{\mathbf{G}}(\mathbf{r}) = |\Omega|^{-\frac{1}{2}} \exp(-i\mathbf{G}\cdot\mathbf{r}), \quad \mathbf{G} \in \mathcal{R}^*.$$

Variational approximation space:

$$X_{E_{\text{cut}}} := \text{Span}\left\{ e_{\mathbf{G}}, \quad \mathbf{G} \in \mathcal{R}^*, \quad \frac{1}{2}|\mathbf{G}|^2 \leq E_{\text{cut}} \right\}.$$

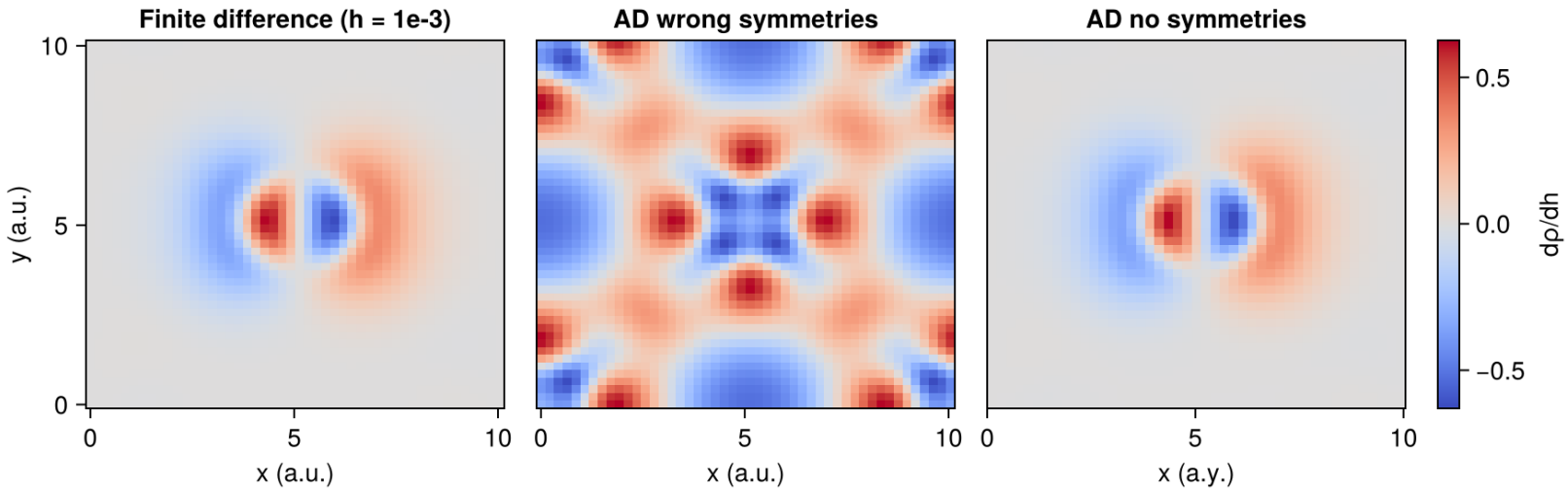
Strengths:

- systematic convergence using E_{cut}
- allows FFT-based methods

Weakness:

- lack of local refinement ($\Longrightarrow$ pseudopotentials)

Special: AD and symmetry-breaking perturbations



A symmetry-breaking density derivative $\frac{\partial \rho}{\partial \theta}$ computed in three ways.

- The problem: Our automated symmetry detection initially only considered the unperturbed crystal structure
- The fallback solution: Disable symmetries (expensive) or handle manually
- Better: determine stabilizer subgroup for perturbation automatically inside custom AD rule

Pseudopotentials

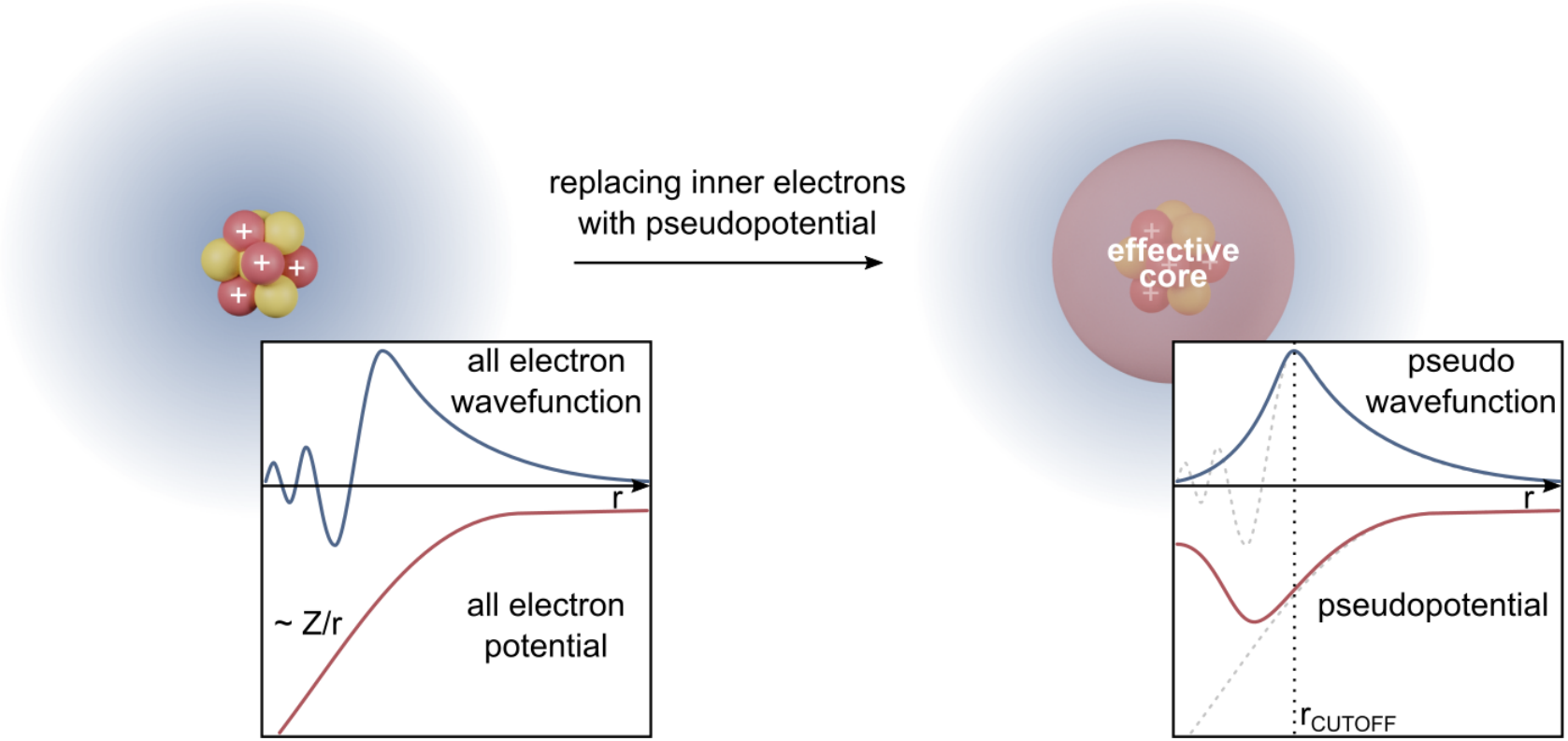
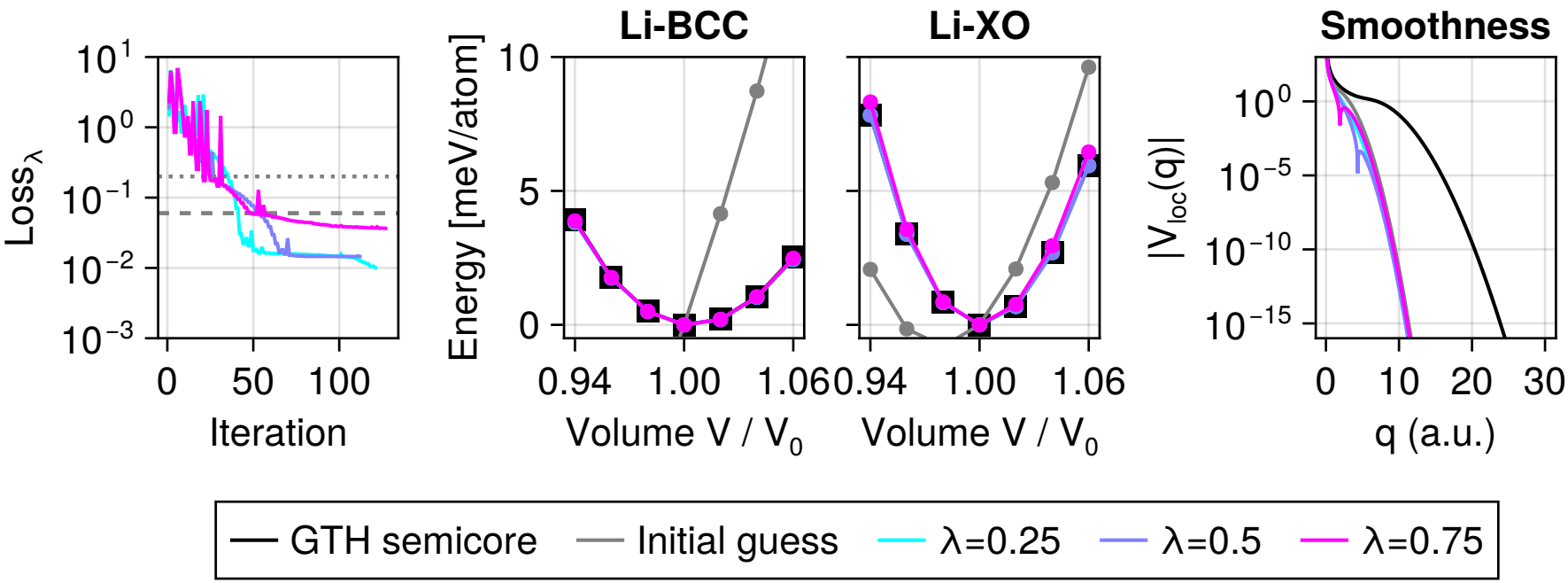


Figure: Edvin Fako, CC BY-SA 4.0, via Wikimedia Commons

Optimizing pseudopotentials[12]



[12] N. F. Schmitz, B. Ploumhans, M. F. Herbst. *npj Comput. Mater.* (2026)