Reliable and efficient methods for computing DFT properties and derivatives

Eric Cancès, Michael F. Herbst*, Gaspard Kemlin,
Antoine Levitt, Benjamin Stamm, Bonan Sun

*Mathematics for Materials Modelling (`matmat.org`), EPFL

18 September 2024

Slides: `https://michael-herbst.com/talks/2024.09.18_MANUEL_Stuttgart.pdf`

Energy consumption of materials discovery

- Current solutions limited by properties of available materials

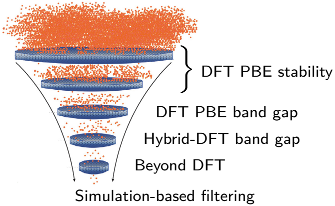
  ⇒ Innovation driven by discovering new materials

- Experimental research extremely energy intensive
  - 1 fume hood $\simeq$ 2-3 average households[1]

⇒ Complement experiment by computational materials discovery

[1] D. Wesolowski *et. al.* Int. J. Sustain. High. Edu. **11**, 217 (2010).

Computational materials discovery



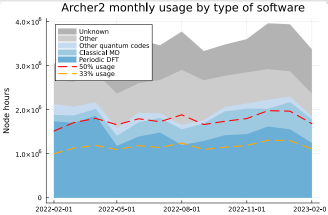
DFT PBE stability

DFT PBE band gap

Hybrid-DFT band gap

Beyond DFT

Simulation-based filtering



Archer2 monthly usage by type of software

- Goal: Only promising candidates made in the lab
- Systematic simulations on $\simeq 10^4 - 10^6$ compounds
  - Noteworthy share of world's supercomputing resources

- Complexity of multiscale materials modelling
  - Many parameters to choose (algorithms, tolerances, models)
  - Despite elaborate heuristics: Thousands of failed calculations

- Need for robust numerical methods
  - Mathematical insight and analysis crucial

  ⇒ Here: Property simulations in density-functional theory (DFT)

Density-functional theory

- DFT approximation: Effective single-particle model

$$\begin{cases} \forall i \in 1 \dots N : \left(-\frac{1}{2}\Delta + V\left(\rho_\Phi\right)\right) \psi_i = \varepsilon_i \psi_i, \\ V(\rho) = V_{\text{ext}} + V_{\text{Hxc}}(\rho), \quad \text{where } V_{\text{Hxc}}(\rho) = v_C\,\rho + V_{\text{XC}}(\rho) \\ \rho_\Phi = \sum_{i=1}^{N} f\left(\frac{\varepsilon_i - \varepsilon_F}{T}\right) \left|\psi_i\right|^2, \end{cases}$$

- Self-consistent field procedure: Fixed-point problem

$$F\big(V_{\text{ext}} + V_{\text{Hxc}}(\rho_{\text{SCF}})\big) = \rho_{\text{SCF}}$$

- $F(V)$ is the potential-to-density map (i.e. diagonalisation)

$$F(V) = \sum_{i=1}^{\infty} f\left(\frac{\varepsilon_i - \varepsilon_F}{T}\right) \left|\psi_i\right|^2 \quad \text{where} \quad \left(-\frac{1}{2}\Delta + V\right) \psi_i = \varepsilon_i \psi_i$$

- ε_F chosen such that $\int F(V) = N$ (number of electrons)

- nuclear attraction V_{nuc}, exchange-correlation V_{XC}, Hartree potential $-\Delta\left(v_C\,\rho\right) = 4\pi\rho$, ψ_i orthogonal, f: Occupation function between 0 and 2

Materials properties: Simulation ↔ experiment

- DFT properties: Response of system to external changes:
  - Connection Theory ⇔ Experiment
  - Modelling: Potential $V(\theta, \rho)$ depends on parameters θ (e.g. atomic positions, el. field)

- SCF procedure yields fixed-point density ρ_{SCF}

$$0 = F\Big(V(\theta, \rho_{\mathsf{SCF}})\Big) - \rho_{\mathsf{SCF}}$$

⇒ Defines implicit function $\rho_{\mathsf{SCF}}(\theta)$

- Properties are derivatives:
  - **Forces** (energy wrt. position), **dipole moment** (energy wrt. el. field), **elasticity** (energy cross-response to lattice deformation), phonons, electronic **spectra**, ...

- There are further interesting derivatives ...
  - θ is parameter of DFT model: ... **uncertainty quantification**
  - θ is parameter of discretisation: ... *a posteriori* **error estimates**

DFT derivatives: Density-functional perturbation theory

$$F(V_{\text{ext}} + V_{\text{Hxc}}(\rho_{\text{SCF}})) = \rho_{\text{SCF}}$$

- δV: Perturbation to V_{ext}, by chain rule

$$\begin{aligned} \delta\rho &= F'(V_{\text{ext}} + V_{\text{Hxc}}(\rho_{\text{SCF}})) \cdot (\delta V + K_* \delta\rho) \\ \Leftrightarrow \quad \delta\rho &= (1 - \chi_0 K)^{-1} \chi_0 \delta V \end{aligned}$$

  where $K_* = V'_{\text{Hxc}}(\rho_{\text{SCF}})$, $\chi_0 = F'(V_{\text{ext}} + V_{\text{Hxc}}(\rho_{\text{SCF}}))$

- Dyson equation: Solved by iterative methods (more on this later)
- Adler-Wiser formula (using $f_n = f(\varepsilon_n)$):

$$\delta\rho(r) = \sum_{n=1}^{\infty} \sum_{m=1}^{\infty} \frac{f_n - f_m}{\varepsilon_n - \varepsilon_m} \psi_n^*(r) \psi_m(r) \left(\delta V_{mn} - \delta\varepsilon_F \delta_{nm}\right)$$

  under the convention

$$\frac{f_n - f_n}{\varepsilon_n - \varepsilon_n} = \frac{1}{T} f'\left(\frac{\varepsilon_n - \varepsilon_F}{T}\right) = f'_n$$

  and where $\delta V_{mn} = \langle \psi_m | \delta V \psi_n \rangle$, $\delta\varepsilon_F$ has an explicit formula

Getting rid of infinities (1)

- Represent $\delta\rho$ by variations $\delta\psi_n$ and δf_n[1] (our new unknowns)

$$\delta\rho(r) = \sum_{n=1}^{N} 2 f_n \operatorname{Re}\left(\psi_n^*(r)\delta\psi_n(r)\right) + \delta f_n \left|\psi_n(r)\right|^2$$

  where $\delta f_n = f_n'(\delta V_{nn} - \delta\varepsilon_F)$

- Define:
  - $P = \operatorname{span}\{\psi_n \,|\, n = 1, \ldots, N\}$: Space spanned by N lowest eigenpairs (ε_n, ψ_n) of H (occupied subspace)
  - $\Pi_Q = 1 - \Pi_P$ with Π_P projector onto P.

- Separate the contributions:

$$f_n \delta\psi_n = f_n \delta\psi_n^P + f_n \delta\psi_n^Q$$

- Note: We deal with the setting of *many* basis functions (Plane waves, wavelets, finite elements, real-space, . . .)

  ⇒ We cannot compute all eigenpairs of H

[1] E. Cancès, MFH, G. Kemlin, *et. al.* Lett. Math. Phys. **113**, 21 (2023).

Getting rid of infinities (2)

$$\sum_{n=1}^{N} 2 f_n \operatorname{Re}\left(\psi_n^*(r)\delta\psi_n^P(r)\right) = \sum_{n=1}^{N}\sum_{m=1}^{N} \frac{f_n - f_m}{\varepsilon_n - \varepsilon_m}\psi_n^*(r)\psi_m(r)\delta V_{mn}$$

- occupied-occupied $\delta\psi_n^P$: Use sum over states

$$f_n \delta\psi_n^P = \sum_{m=1, m\neq n}^{N} \Gamma_{mn}\psi_m$$

where we need $\Gamma_{nn} = 0$ and

$$\Gamma_{mn} + \Gamma_{nm}^* = \frac{f_n - f_m}{\varepsilon_n - \varepsilon_m}\delta V_{mn}$$

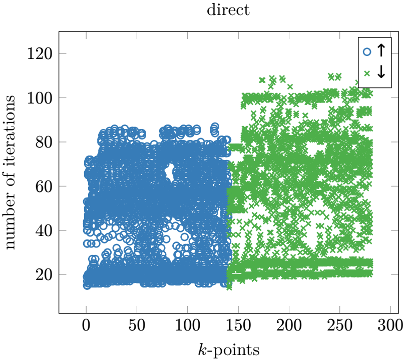
- Question 1: This is not unique. How to choose Γ_{nm}?

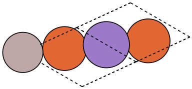
Getting rid of infinities (3)

- unocc-occ $\delta\psi_n^Q$: Use Sternheimer equation

$$\Pi_Q(H-\varepsilon_n)\Pi_Q\delta\psi_n = -\Pi_Q\delta V\psi_n \qquad \forall n = 1,\dots,N \quad (*)$$

- Question 2: $(*)$ is badly conditioned if gap $\varepsilon_{N+1}-\varepsilon_N$ small
  $\Rightarrow$ How can we make response cheaper for metals?



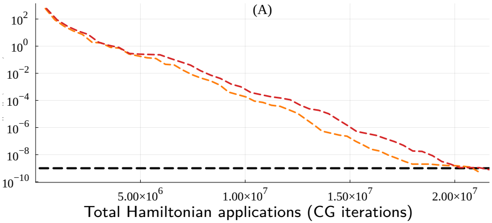


Fe$_2$MnAl Heusler alloy

Getting rid of infinities (4)

$$\frac{\partial \rho_{\text{SCF}}}{\partial \theta} = [1 - \chi_0 K]^{-1} \chi_0 \frac{\partial V}{\partial \theta} \qquad \text{(Dyson)}$$

- Dyson equation solved iteratively (e.g. GMRES)
- Each matvec $\chi_0 \delta V$ requires solving N Sternheimer equations
- Question 3: How to choose Sternheimer tolerance τ^{CG} adaptively (depending on GMRES tolerance τ)
- Naive strategies: $\tau^{\text{CG}} = \tau/100$ and $\tau^{\text{CG}} = \tau/10$ for $\tau = 10^{-9}$



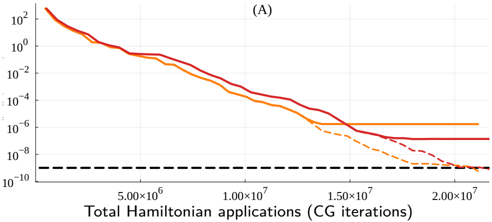
- Dashed: GMRES estimated residual norm
- Solid: Actual residual norm

- Fail by 3 orders (Al$_{40}$ supercell)

Getting rid of infinities (4)

$$\frac{\partial \rho_{\mathsf{SCF}}}{\partial \theta} = [1 - \chi_0 K]^{-1} \chi_0 \frac{\partial V}{\partial \theta} \qquad \text{(Dyson)}$$

- Dyson equation solved iteratively (e.g. GMRES)
- Each matvec $\chi_0 \delta V$ requires solving N Sternheimer equations
- Question 3: How to choose Sternheimer tolerance τ^{CG} adaptively (depending on GMRES tolerance τ)
- Naive strategies: $\tau^{\mathsf{CG}} = \tau/100$ and $\tau^{\mathsf{CG}} = \tau/10$ for $\tau = 10^{-9}$



- Dashed: GMRES estimated residual norm
- Solid: Actual residual norm

- Fail by 3 orders (Al$_{40}$ supercell)

Contents

1. Gauge choices

2. Sternheimer with a Schur complement

3. Adaptive Sternheimer tolerance

4. Errors and sensitivities

The bad choice: Orthogonal gauge

- Recall, we need

$$\Gamma_{mn} + \Gamma^*_{nm} = \Delta_{mn} = \frac{f_n - f_m}{\varepsilon_n - \varepsilon_m} \delta V_{mn}$$

and additionally $\Gamma_{mn} = \langle \psi_m | f_n \delta\psi_n \rangle$ by construction

- Zero temperature (insulators): $\delta\psi^P = 0$

⇒ Orbitals can be kept orthogonal under response (for insulators)

- **Orthogonal gauge**: Enforce orthogonality in all cases, i.e.

$$0 = \delta \langle \psi_m | \psi_n \rangle = \langle \delta\psi_m | \psi_n \rangle + \langle \psi_m | \delta\psi_n \rangle$$

$$\Rightarrow \qquad 0 = \Gamma_{mn}/f_n + \Gamma^*_{nm}/f_m$$

$$\Rightarrow \qquad \Gamma^{\text{orth}}_{mn} = \frac{f_n}{\varepsilon_n - \varepsilon_m} \delta V_{mn}$$

- Problem: This can lead to a large contribution as $\varepsilon_n \to \varepsilon_m$ which is almost compensated by $\Gamma^{\text{orth},*}_{nm}$

⇒ Loss of numerical precision

The optimal choice: Minimal gauge

- Minimise the size of all contributions to $\delta\psi_n$, i.e.

$$\min \sum_{m,n} \frac{1}{f_n^2} \left|\Gamma_{mn}\right|^2$$

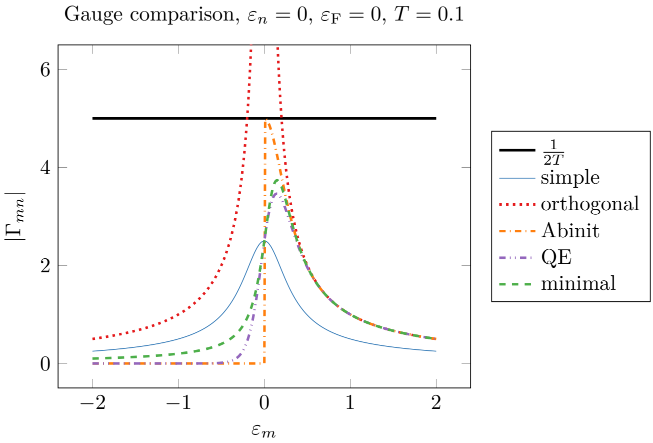
$$\text{s.t. } \Gamma_{mn} + \Gamma_{nm}^* = \Delta_{mn} = \frac{f_n - f_m}{\varepsilon_n - \varepsilon_m} \delta V_{mn}$$

- **Minimal gauge:** Solution to above problem

$$\Gamma_{mn} = \frac{f_n^2}{f_n^2 + f_m^2} \Delta_{mn}$$

- Other gauge choices:
  - Quantum Espresso: $\Gamma_{mn} = f_{\text{FD}}\left(\frac{\varepsilon_n - \varepsilon_m}{T}\right) \Delta_{mn}$
  - Abinit: $\Gamma_{mn} = \mathbb{1}_{f_n > f_m} \Delta_{mn}$

Comparison of gauges



- Graph investigates the growth of $\delta\rho$ wrt. δV
- $\frac{1}{2T}$ gives lower bound (from Δ_{mn}), we don't want to overshoot it

$\Rightarrow$ Orthogonal should be avoided, all others reasonable

Contents

1. Gauge choices
2. Sternheimer with a Schur complement
3. Adaptive Sternheimer tolerance
4. Errors and sensitivities

Extra SCF orbitals[1]

- Each *application* of χ_0 to a δV requires solving Sternheimer for all $n = 1, \ldots, N$

$$\left(\tilde{H} - \varepsilon_n\right) \delta\psi_n^Q = -\Pi_Q \, \delta V \psi_n \qquad \forall n = 1, \ldots, N$$

$H = -\frac{1}{2}\Delta + V$, $\tilde{H} = \Pi_Q H \Pi_Q$ $\quad (\varepsilon_n, \psi_n)$ eigenpairs of H

- If gap $\varepsilon_{N+1} - \varepsilon_N$ closes (metals), conditioning gets worse
- But we have not used all we know:
  - Standard iterative diagonalisations (and thus SCFs) yield N_ex additional orbitals $\Phi = (\psi_{N+1}, \ldots, \psi_{N+N_\text{ex}})$
  - Notable property: $\Phi^T H \Phi = \operatorname{diag}(\varepsilon_{N+1}, \ldots, \varepsilon_{N+N_\text{ex}})$

[1] E. Cancès, MFH, G. Kemlin, *et. al.* Lett. Math. Phys. **113**, 21 (2023).

Schur complement approach to response[1]

$$\left(\tilde{H} - \varepsilon_n\right) \delta\psi_n = -\Pi_Q \, \delta V \psi_n \; \forall i = 1, \dots, N$$

- Use N_{ex} extra orbitals to partition $\tilde{H}$:

$$\tilde{H} = \begin{pmatrix} \diagdown E_{\text{ex}} \diagdown & \mathbf{C} \\ \mathbf{C}^\dagger & \mathbf{R} \end{pmatrix} \quad \text{where} \quad \begin{aligned} E_{\text{ex}} &= \operatorname{diag}(\varepsilon_{N+1}, \dots, \varepsilon_{N+N_{\text{ex}}}) \\ \mathbf{C} &= \Phi\Phi^\dagger \tilde{H} \left(1 - \Phi\Phi^\dagger\right) \\ \mathbf{R} &= \left(1 - \Phi\Phi^\dagger\right) \tilde{H} \left(1 - \Phi\Phi^\dagger\right) \end{aligned}$$

⇒ Typical Schur complement setting:

- Solve for $\Phi\Phi^\dagger \delta\psi_n$ exactly
- $x = \left(1 - \Phi\Phi^\dagger\right) \delta\psi_n$ obtained by (b appropriate RHS)

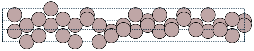
$$\left(\mathbf{R} - \mathbf{C}^\dagger E_{\text{ex}}{}^{-1} \mathbf{C} - \varepsilon_n\right) x = b$$

- Smallest eigenvalue about $\varepsilon_{N+N_{\text{ex}}} - \varepsilon_N$

⇒ Conditioning improved, savings on CG iterations

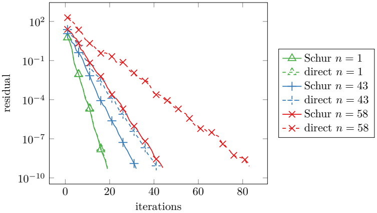
[1] E. Cancès, MFH, G. Kemlin, *et. al.* Lett. Math. Phys. **113**, 21 (2023).

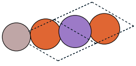
Schur-based response: Numerical examples[1]



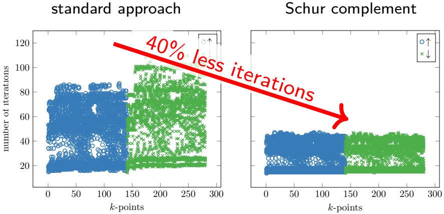
Al_{40} rattled supercell

k-point $[0.333, 0.0, 0.0]$





$\mathrm{Fe}_2\mathrm{MnAl}$ Heusler alloy



standard approach

Schur complement

40% less iterations

- Largest reduction in iterations near Fermi level ($n = 58$)
  (where gap is smallest)
- Overall 17% less iterations

⇒ Improvement comes for free
(extra bands needed during SCF)

- Relevant materials class with unusual magnetic properties
- Translates to challenging numerical behaviour
- Schur-based approach tames CG
- ca. 40% less iterations

[1] E. Cancès, MFH, G. Kemlin, *et. al.* Lett. Math. Phys. **113**, 21 (2023).

Contents

1. Gauge choices
2. Sternheimer with a Schur complement
3. Adaptive Sternheimer tolerance
4. Errors and sensitivities

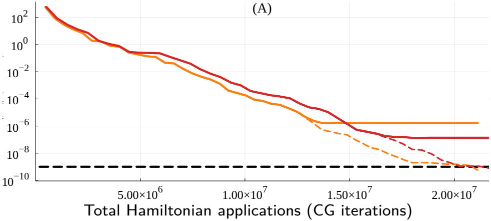
Choosing the Sternheimer tolerance

$$\frac{\partial \rho_{\text{SCF}}}{\partial \theta} = \left[1 - \chi_0 K\right]^{-1} \chi_0 \frac{\partial V}{\partial \theta} \qquad \left(\tilde{H} - \varepsilon_n\right) \delta\psi_n = -P\, \delta V \psi_n \ \forall i = 1, \dots, N$$

GMRES tolerance τ

CG tolerance $\tau_{i,n}^{\text{CG}}$

- Dyson + Sternheimer: Nested iteratively solved problems
- Tolerance for CGs when applying χ_0 ?
- Naive strategies: $\tau_{i,n}^{\text{CG}} = \tau/100$ and $\tau_{i,n}^{\text{CG}} = \tau/10$ for $\tau = 10^{-9}$



- Dashed: GMRES estimated residual norm
- Solid: Actual residual norm

- Fail by 3 orders (Al$_{40}$ supercell)

$\Rightarrow$ Need adaptive & guaranteed strategy for $\tau_{i,n}^{\text{CG}}$

Inexact GMRES[1]

- Inexact application: $(A + E_k)v_k$
- Inexact Arnoldi decomposition

$$AV_m + [E_1 v_1, E_2 v_2, \cdots, E_m v_m] = V_{m+1} H_m$$

- GMRES terminates with exact residual $\|r_m\| \leq \tau$ if

$$\|E_k v_k\| \leq \frac{\sigma_m(H_m)}{3m} \frac{\tau}{\|\tilde{r}_{k-1}\|}$$

where

- $\|\tilde{r}_{i-1}\|$: GMRES estimated residual norm
- $\sigma_m(H_m)$: m-th condition number of GMRES Hessenberg
- m: GMRES maximal subspace size

[1] V. Simonicini, D. Szyld. SIAM J. Sci. Comput., **25**, 454 (2003).

Dyson equation case

$$\frac{\partial \rho_{\mathsf{SCF}}}{\partial \theta} = \left[1 - \chi_0 K\right]^{-1} \chi_0 \frac{\partial V}{\partial \theta} \qquad \left(\tilde{H} - \varepsilon_n\right) \delta\psi_n = -P\, \delta V\, \psi_n \;\forall i = 1, \dots, N$$

GMRES tolerance τ CG tolerance $\tau_{i,n}^{\mathsf{CG}}$

$$\delta\rho(r) = \sum_{n=1}^{N} 2 f_n \operatorname{Re}\left(\psi_n^*(r)\delta\psi_n(r)\right) + \delta f_n \left|\psi_n(r)\right|^2$$

- Operator $A = 1 - \chi_0 K$, inexact operator $\tilde{A} = 1 - \tilde{\chi_0} K$ (using CG tolerance $\tau_{i,n}^{\mathsf{CG}}$)
- Then (without Schur complement trick):

$$\left\|(A - \widetilde{A})\, v_i\right\| \lesssim \sqrt{N}\, \|K v_i\| \cdot \max_{x\in\Omega} \max_{\substack{c\in\mathbb{R}^N \\ \|c\|=1}} \left|\sum_{n=1}^{N} 2 \operatorname{Re}\left(c_n\, \psi_n(x)\right)\right| \cdot \max_{n=1,\cdots,N} \frac{f_n}{\varepsilon_{N+1} - \varepsilon_n}\; \tau_{i,n}^{\mathsf{CG}}\, k \sqrt{\mathsf{Ecut}^{3/2}}$$

where k are system-size independent constants

⇒ Combine with inexact GMRES to adaptively determine $\tau_{i,n}^{\mathsf{CG}}$

Inexact Krylov methods for response[1]

- **Theorem:** Guaranteed convergence of GMRES to τ when

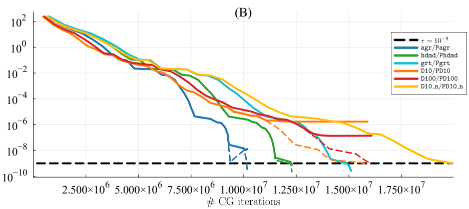
$$\tau_{i,n}^{\mathsf{CG}} \lesssim \frac{1}{k\,C} \, \frac{\sqrt{|\Omega|}}{N\,\mathsf{Ecut}^{3/4}} \, \frac{1}{f_n} \, \frac{s}{3m\ \|\tilde{r}_{i-1}\|} \, \tau$$

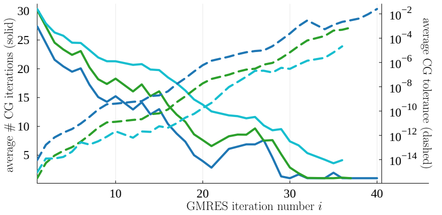
  - $\|\tilde{r}_{i-1}\|$: GMRES estimated residual norm
  - s: Estimate for cond. num. of GMRES Hessenberg matrix (updated on the fly)
  - m: GMRES maximal subspace size
  - k: Constants of order 1
  - C: System size-indep. const. (includes blue from prev. slide)

- Main features:
  - Looser tolerance closer to convergence (as $\tilde{r}_{i-1} \to 0$)
  - Looser tolerance for small f_n (when Sternheimer worst conditioned)
  - Tighter tolerance for larger systems (as $\frac{\sqrt{|\Omega|}}{N} \searrow$)

[1] MFH, B. Sun, *in preparation*.

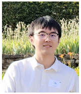
Inexact Krylov methods for response[1]





- Guaranteed (grt) computes C exactly
- Balanced (hdmd) sets $C = 1$
  - Requires a good preconditioner for metals (since $\|Kv_i\|$ dropped)
  - ⇒ We employ standard Kerker preconditioner also in GMRES
- Aggressive (agr) drops even more constants
  - Even faster than hdmd, but can be a factor 10 off
- From about 20M to 12M Hamiltonian applications (the expensive step)

⇒ Superlinear convergence



Bonan Sun

[1] MFH, B. Sun, *in preparation*.

Contents

1. Gauge choices
2. Sternheimer with a Schur complement
3. Adaptive Sternheimer tolerance
4. Errors and sensitivities

DFT error: Computing model sensitivities

- DFT models usually contain parameters θ
  - Natural question: How sensitive are results ?
- Consider model sensitivity of force $\mathcal{F}(\rho, \theta)$:

$$\frac{d\mathcal{F}}{d\theta} = \frac{\partial \mathcal{F}}{\partial \rho_{\text{SCF}}} \frac{\partial \rho_{\text{SCF}}}{\partial \theta} + \frac{\partial \mathcal{F}}{\partial \theta} \tag{1}$$

- Computed by response theory (we've seen this before !):

$$\frac{\partial \rho_{\text{SCF}}}{\partial \theta} = [1 - \chi_0 K]^{-1} \chi_0 \frac{\partial V}{\partial \theta}$$

- We know how to solve this (previous section)

⇒ Should be easy, right ?

DFT error: Computing model sensitivities

- DFT models usually contain parameters θ
  - Natural question: How sensitive are results ?
- Consider model sensitivity of force $\mathcal{F}(\rho, \theta)$:

$$\frac{d\mathcal{F}}{d\theta} = \frac{\partial \mathcal{F}}{\partial \rho_{\text{SCF}}} \frac{\partial \rho_{\text{SCF}}}{\partial \theta} + \frac{\partial \mathcal{F}}{\partial \theta} \tag{1}$$

- Computed by response theory (we've seen this before !):

$$\frac{\partial \rho_{\text{SCF}}}{\partial \theta} = [1 - \chi_0 K]^{-1} \chi_0 \frac{\partial V}{\partial \theta}$$

- We know how to solve this (previous section)

⇒ Should be easy, right ?

Computing sensitivities

$$\frac{d\mathcal{F}}{d\theta} = \frac{\partial \mathcal{F}}{\partial \rho_{\mathsf{SCF}}} \frac{\partial \rho_{\mathsf{SCF}}}{\partial \theta} + \frac{\partial \mathcal{F}}{\partial \theta}; \qquad \frac{\partial \rho_{\mathsf{SCF}}}{\partial \theta} = [1 - \chi_0 K]^{-1} \chi_0 \frac{\partial V}{\partial \theta} \tag{1}$$

- **Obstacle:** Parameters are in innermost layer (model definition)
  - Each DFT model: Different derivatives $\frac{\partial V}{\partial \theta}$ (can be horrible)
  - Each quantity of interest: Different sensitivity expression (1)

  ⇒ Combinatorial explosion

- Use algorithmic differentiation (AD) (≈ automatic derivatives)
  - Generic framework for DFT derivatives / response properties

  ⇒ Breaks "one PhD student per derivative" paradigm

  ⇒ New properties/derivatives by non-DFT experts!

Sensitivities in practice

```
function dft_forces(θ)
    system = ...
    model  = model_DFT(system, PbeExchange(θ))
    basis  = PlaneWaveBasis(model; Ecut=..., kgrid=... )
    scfres = self_consistent_field(basis).energies.total
    compute_forces_cart(scfres)
end
sensitivities = ForwardDiff.gradient(dft_forces, θ)
```

$$\frac{d\mathcal{F}}{d\theta} = \frac{\partial \mathcal{F}}{\partial \rho_{\text{SCF}}} \frac{\partial \rho_{\text{SCF}}}{\partial \theta} + \frac{\partial \mathcal{F}}{\partial \theta}$$

- AD saves manual coding: Request gradient (1), AD delivers
- AD *orchestrates* calculation, i.e. constructs RHS for

$$\frac{\partial \rho_{\text{SCF}}}{\partial \theta} = \left[1 - \chi_0 K\right]^{-1} \chi_0 \frac{\partial V}{\partial \theta}$$

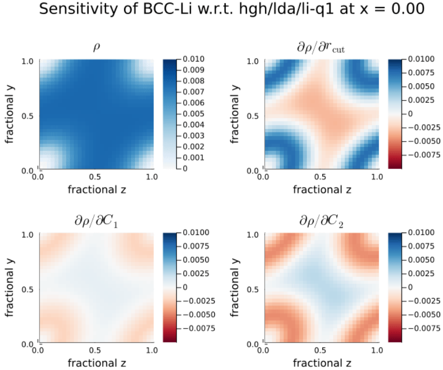
- . . . and expression to compute $\frac{d\mathcal{F}}{d\theta}$ from it

- Hard to achieve in traditional electronic structure codes
- Readily available in DFTK (`https://dftk.org`)

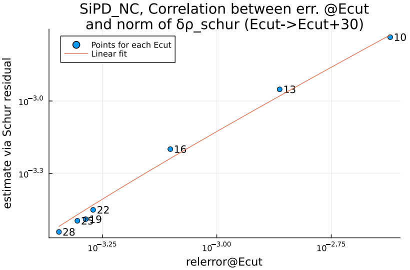
Niklas Schmitz

Outlook: Pseudopotential sensitivities

- Pseudopotential sensitivity of electronic density



Outlook: A posteriori error estimation



Estimation of basis set error in ρ

- Suppose an SCF is solved in a small basis to obtain ρ

$\Rightarrow$ Obtain estimate of error in ρ by solving a response problem !

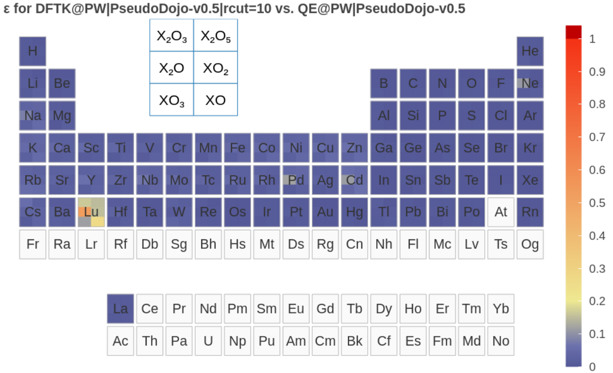
- Basis of recent practical error bound for forces[1]

  (Development & testing has been performed in DFTK)

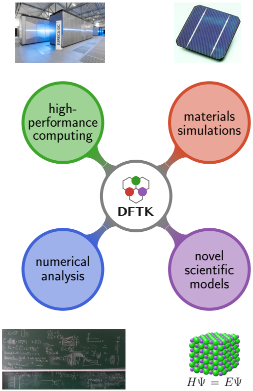
[1] E. Cancès, G. Dusson, G. Kemlin *et. al.* SIAM J. Sci. Comp., **44**, B1312 (2022).

Closing the gap between maths and high-throughput

- **DFTK** plugin for AiiDA workflow manager
- **Goal:** Simplify automated testing of novel algorithms
- Verification study Quantum-Espresso vs. **DFTK**



Density-functional toolkit[1] — https://dftk.org



- julia code for cross-disciplinary research:
  - Allows restriction to relevant model problems,
  - *and* scale-up to application regime (1000 electrons)
  - Sizeable feature set in 7500 lines of code
  - Including some unique features (Self-adapting algorithms)
  - Integrated with high-throughput: MARVEL AiiDA
- Fully composable due to julia abstractions:
  - Arbitrary precision (32bit, >64bit, . . .)
  - Algorithmic differentiation (AD)
  - HPC tools: GPU acceleration, MPI parallelisation
- Accessible high-productivity research framework:
  - Key contributions by undergrads (AD, GPU, Pseudos, . . .)
  - Over 30 contributors in 5 years (Maths, physics, CS, . . .)

Advertisement break

Open PostDoc in the EPFL MatMat group

Topic: Efficient inverse materials design

- Interdisciplinary environment
- Bayesian optimisation
- AD & gradient approaches
- See `https://matmat.org/jobs/`

Psi-k workshop *(with A. Levitt, J. Haegeman)*:

"Julia for numerical problems in quantum and solid-state physics"

- **26–28 November 2024** at EPFL, CECAM-HQ, Lausanne
- Targets people from linear algebra, physics and computer science
- Techniques & collaborations enabled by julia

⇒ `https://www.cecam.org/workshop-details/1355`
Deadline: 20th Sep

Summary

- Challenges of response calculations for metals
  - Closing gap worsens conditioning of linear system
  - Ambiguity in representing density response (gauge freedom)

- Mathematical analysis of DFPT
  - Schur-complement approach to response
  - Adaptive Krylov methods
  - Preconditioning strategies for Dyson equation in metals
  - $\sim 80\%$ faster, while no additional cost
  - Readily available in DFTK

- Enables fast & robust derivative computations (in combination with AD)
  - Fast properties
  - Fast error estimates
  - Fast sensitivities

Acknowledgements

MatMat group

- Anna Paulish (MXt Mat)
- Niklas Schmitz (MXt Mat)
- Cédric Travelletti (MXt Mat)

Schur complement

- Eric Cancès (École des Ponts)
- Gaspard Kemlin (Université de Picardie)
- Antoine Levitt (Université Paris-Saclay)
- Benjamin Stamm (Stuttgart)

Inexact Krylov

- Bonan Sun (EPFL, now MPI Magdeburg)

Fonds national suisse

MARVEL
NATIONAL CENTRE OF COMPETENCE IN RESEARCH

EPFL MXt Mat

Questions?

https://matmat.org

mfherbst

✉ michael.herbst@epfl.ch

🖥 https://michael-herbst.com/talks/2024.09.18_MANUEL_Stuttgart.pdf

DFTK https://dftk.org

Contents

How does algorithmic differentiation (AD) work?

```
function F(x)
    y1 = x[1] + x[2]  # F1 = sum
    y2 = 2 * p        # F2 = double
    return y2
end
```

- Goal: Compute derivative of this code
- Function $F : \mathbb{R}^2 \to \mathbb{R}$ with $F(x) = \text{double}(\text{sum}(x_1, x_2))$
- Derivative at $\tilde{x}$ is characterised by its Jacobian matrix

$$[J_F(\tilde{x})]_{ij} = \left(\left. \frac{\partial F}{\partial x} \right|_{x=\tilde{x}} \right)_{ij} = \left. \frac{\partial F_i}{\partial x_j} \right|_{x=\tilde{x}}$$

- Finite differences: Simple, one column at a time:

$$[J_F(\tilde{x})]_{:,j} = \frac{F(\tilde{x} + \alpha e_j) - F(\tilde{x})}{\alpha}$$

  (with e_i unit vectors)

$\Rightarrow$ Inaccurate and slow ($\mathcal{O}(N)$ times primal cost)

Chain rule to the rescue!

```
function F(x)
    y1 = x[1] + x[2]  # F1 = sum
    y2 = 2 * p        # F2 = double
    return y2
end
```

$$F(x) = \text{double}(\text{sum}(x_1, x_2))$$

- "double" and "sum" are simple and frequent primitives

⇒ Key idea of AD:

  - Compose the derivative of F from the Jacobians of primitives
  - Assumed to be known and already implemented

- Use chain rule as glue, e.g. for a Jacobian element at $\tilde{x}$:

$$\frac{\partial F_i}{\partial x_j} = \frac{\partial \text{double}(a)}{\partial a} \left(\frac{\partial \text{sum}(c,d)}{\partial c} \frac{\partial x_1}{\partial x_j} + \frac{\partial \text{sum}(c,d)}{\partial d} \frac{\partial x_2}{\partial x_j} \right)$$

- More compact: $\quad e_i^T J_F e_j = e_i^T J_{\text{double}} J_{\text{sum}} e_j$

- Note: J_{double} is needed at $\text{sum}(\tilde{x}_1, \tilde{x}_2)$

Forward-mode algorithmic differentiation

```
function F(x)
    y1 = x[1] + x[2]  # F1 = sum
    y2 = 2 * p        # F2 = double
    return y2
end
```

$$F(x) = \mathsf{double}(\mathsf{sum}(x_1, x_2))$$

$$e_i^T J_F e_j = e_i^T J_{\mathsf{double}} J_{\mathsf{sum}} e_j$$

- Forward-diff: Evaluate in order with *primal* F:
  1. Set $y_0 = (x_1, x_2)$, $\dot{y}_0 = e_j$
  2. Compute $y_1 = \mathsf{sum}(y_0)$ and $\dot{y}_1 = J_{\mathsf{sum}}(y_0)\dot{y}_0$
  3. Compute $y_2 = \mathsf{double}(y_1)$ and $\dot{y}_2 = J_{\mathsf{double}}(y_1)\dot{y}_1$
  4. Obtain $F(x_1, x_2)$ as y_2 and $[J_F]_{:,j} = \dot{y}_2$

⇒ Again one column of J_F at a time

- Implementation: Numbers → dual numbers
- Vectorisation & other tricks: Usually faster than finite diff.
- But: Still $\mathcal{O}(N)$ times primal cost

Optimal cost for differentiation (1)

```
function F(x)
    y1 = x[1] + x[2]   # F1 = sum
    y2 = 2 * p         # F2 = double
    return y2
end
```

$$F(x) = \text{double}(\text{sum}(x_1, x_2))$$

$$e_i^T J_F e_j = e_i^T J_{\text{double}} J_{\text{sum}} e_j$$

Proposition

If $f : \mathbb{R}^N \to \mathbb{R}$ is a differentiable function, computing $\nabla f = J_f$ is asymptotically not more expensive than f itself.

⇒ This is violated for finite diff and forward diff.

- Let's try to be more clever:
  - We could write $F(x) = b^T A x$ for appropriate (sparse) A, b
  - Equivalent formulation: $F(x) = (A^T b)^T x$
  - Differentiate that: $\nabla F = A^T b$ ⇒ costs the same as F.
- To generalise this idea note that (for scalar functions)

$$F(x) = b^T J_F x + O(x^2)$$

Optimal cost for differentiation (2)

```
function F(x)
    y1 = x[1] + x[2]   # F1 = sum
    y2 = 2 * p         # F2 = double
    return y2
end
```

$$F(x) = \text{double}(\text{sum}(x_1, x_2))$$

$$e_i^T J_F e_j = e_i^T J_{\text{double}} J_{\text{sum}} e_j$$

- Let's try to be more clever:
    - We could write $F(x) = b^T A x$ for appropriate (sparse) A, b
    - Equivalent formulation: $F(x) = (A^T b)^T x$
    - Differentiate that: $\nabla F = A^T b$ $\Rightarrow$ costs the same as F.

- To generalise this idea note that (for scalar functions)

$$F(x) = b^T J_F x + \mathcal{O}(x^2) \qquad \text{with } b = e_1 = 1$$

$\Rightarrow$ Focus on computing adjoint of Jacobian:

$$e_i^T J_F e_j = \left(J_F^T e_i\right)^T e_j = \left(J_{\text{sum}}^T J_{\text{double}}^T e_i\right)^T e_j$$

Adjoint-mode algorithmic differentiation

```
function F(x)
    y1 = x[1] + x[2]  # F1 = sum
    y2 = 2 * p        # F2 = double
    return y2
end
```

$$F(x) = \mathsf{double}(\mathsf{sum}(x_1, x_2))$$

$$e_i^T J_F e_j = \left(J_{\mathsf{sum}}^T J_{\mathsf{double}}^T e_i \right)^T e_j$$

- Adjoint-mode AD: Derivative in reverse instruction order.
- *Forward pass*:
  1. Set $y_0 = (x_1, x_2)$
  2. Compute $y_1 = \mathsf{sum}(y_0)$ and store it
  3. Compute $y_2 = \mathsf{double}(y_1)$ and store it
- *Reverse pass*:
  1. Set $\bar{y}_2 = e_i$
  2. Compute $\bar{y}_1 = [J_{\mathsf{double}}(y_1)]^T \bar{y}_2$
  3. Compute $\bar{y}_0 = [J_{\mathsf{sum}}(y_0)]^T \bar{y}_1$
- Obtain $[J_F]_{i,:}$ as $\bar{y}_0^T \implies$ One row at a time

Adjoint-mode algorithmic differentiation (2)

- Given $f : \mathbb{R}^N \to \mathbb{R}$ there is only one $e_i = 1$

$\Rightarrow$ Only one reverse pass computes full gradient ∇f

$\Rightarrow$ $\mathcal{O}(1)$ times primal cost

- Many names:
  - Adjoint trick, back propagation, reverse-mode AD
- Some difficulties / challenges:
  - Reverse control flow required!
  - (Hurts your heads sometimes)
  - Storage / memory costs
  - All mutation is bad ...
- One has to be a bit more clever for iterative algorithms ...

Contents

5 Algorithmic differentiation

6 Details on Schur complement approach

Sternheimer equations

$$\frac{\partial \rho_{\text{SCF}}}{\partial \theta} = [1 - \chi_0 K]^{-1} \chi_0 \frac{\partial V}{\partial \theta} \tag{2}$$

- Solving (2) (Dyson equation) is not cheap
- Each *application* of χ_0 to a δV requires iteratively solving (3) (Sternheimer equation) for all $n = 1, \ldots, N$

$$\Pi_Q (H - \varepsilon_n) \Pi_Q \delta\psi_n = -\Pi_Q \delta V \psi_n \tag{3}$$

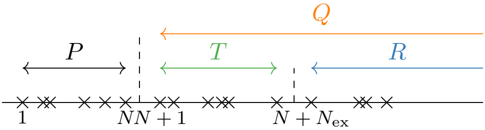
  where

  - $\delta\psi_n$: Orbital perturbation (to be determined)
  - $P = \operatorname{span}\{\psi_n \,|\, n = 1, \ldots, N\}$: Space spanned by N lowest eigenpairs (ε_n, ψ_n) of H (occupied subspace)
  - $\Pi_Q = 1 - \Pi_P$ with Π_P projector onto P.

- Caveat: (3) is badly conditioned if gap $\varepsilon_{N+1} - \varepsilon_N$ small

  $\Rightarrow$ Response can be expensive for metals

Schur-complement approach[1] (1)



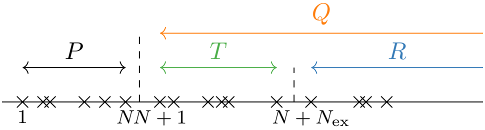
- SCF diagonalisations yield N_{ex} additional orbitals $\Phi = (\psi_{N+1}, \dots, \psi_{N+N_{\text{ex}}})$ spanning T.
  - Not fully converged, i.e. $H\psi_n \neq \varepsilon_n \psi_n$ for $n = N+1, \dots, N+N_{\text{ex}}$
  - But: $\Phi^\dagger H \Phi = \operatorname{diag}(\varepsilon_{N+1}, \dots, \varepsilon_{N+N_{\text{ex}}})$
- Split orbital perturbation $\Pi_Q \delta\psi_n = \Phi\alpha_n + \Pi_R \delta\psi_n^R$ to obtain:

$$\Pi_Q (H - \varepsilon_n) \Phi \alpha_n + \Pi_Q (H - \varepsilon_n) \Pi_R \delta\psi_n^R = \underbrace{-\Pi_Q \delta V \psi_n}_{:=b_n}$$

- Schur complement: Solve component in T (along Φ) explicitly:

$$\alpha_n = \underbrace{\left(\Phi^\dagger H \Phi\right)^{-1}}_{=D^{-1}} \left(\Phi^\dagger b_n - \underbrace{\Phi^\dagger \left(H - \varepsilon_n\right) \Pi_R}_{=h_{RT}^\dagger} \delta\psi_n^R\right)$$

Schur-complement approach[1] (2)



$$\Pi_Q(H-\varepsilon_n)\Phi\alpha_n + \Pi_Q(H-\varepsilon_n)\Pi_R\delta\psi_n^R = b_n$$

$$\alpha_n = D^{-1}\left(\Phi^\dagger b_n - h_{RT}^\dagger\delta\psi_n^R\right)$$

- It only remains to iteratively solve the component $\delta\psi_n^R$:

$$\left[\Pi_R(H-\varepsilon_n)\Pi_R - h_{RT}D^{-1}h_{RT}^\dagger\right]\Pi_R\delta\psi_n^R = \left[\Pi_R - h_{RT}D^{-1}\Phi^\dagger\right]b_n$$

- Π_R almost removes small eigenmodes of $H-\varepsilon_n$

⇒ Smallest eigenvalue of $\Pi_R(H-\varepsilon_N)\Pi_R$ is about $\varepsilon_{N+N_{\text{ex}}}-\varepsilon_N$

- For metals: Substantially larger than $\varepsilon_{N+1}-\varepsilon_N$

⇒ Improved conditioning

[1] E. Cancès, MFH, G. Kemlin, *et. al.* Lett. Math. Phys. **113**, 21 (2023).