

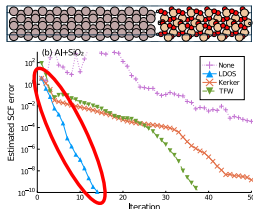
Accelerating mathematical developments in materials modelling by composable software

Michael F. Herbst

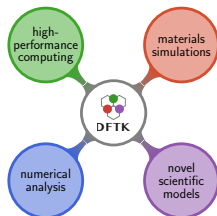
Mathematics for Materials Modelling (matmat.org), EPFL

1 June 2023

Slides: https://michael-herbst.com/talks/2023.06.01_gamm_composable.pdf



Developing novel DFT algorithms

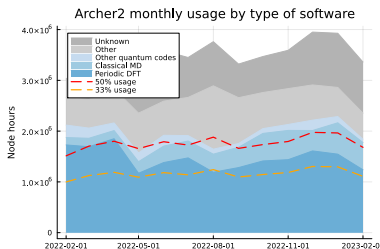
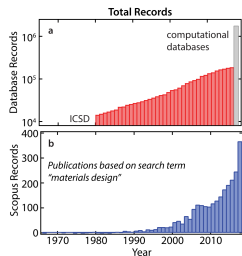


Julia

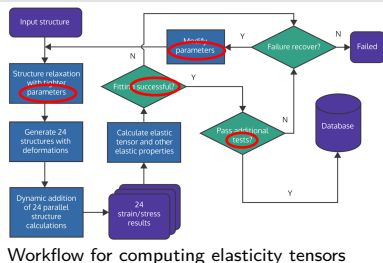
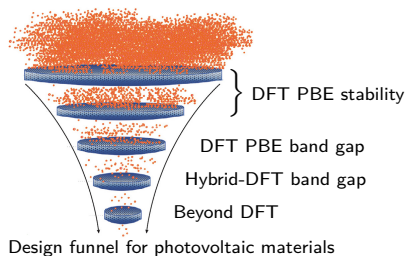
The Julia logo consists of the word 'julia' in a bold, black, sans-serif font. Above the letters 'i', 'l', and 'a' are three small colored circles: a blue circle above the 'i', a green circle above the 'l', and a purple circle above the 'a'.

Tackling to 21st century challenges

- 21st century challenges:
 - Renewable energy, green chemistry, health care ...
- Current solutions limited by properties of available materials
 - ⇒ Innovation driven by **discovering new materials**
- Crucial tool: **Computational materials discovery**
 - Systematic simulations on $\simeq 10^4 - 10^6$ compounds
 - Complemented by data-driven approaches
 - **Noteworthy share** of world's supercomputing resources



Sketch of high-throughput workflows



- Many parameters to choose (algorithms, tolerances, models)
 - Elaborate heuristics: **Failure rate** $\simeq 1\%$
 - Still: **Thousands** of failed calculations
 - ⇒ Wasted resources & increased human attention (limits throughput)
- **Goal:** **Self-adapting black-box algorithms**
 - Transform **empirical wisdom** to built-in **convergence guarantees**
 - Requires: Uncertainty quantification & error estimation
 - ⇒ Understand **where and how** to spend efforts best

Opportunities for mathematical research

- **Gap:** Mathematical understanding & simulation practice
- Broad range of concerned mathematical fields:
 - Optimisation, numerical linear algebra, analysis of PDEs, uncertainty quantification, model order reduction, ...
- Application domain: **Source for research problems**
 - Large-scale eigenvalue problems
(L. Lin, Y. Saad, C. Yang, ...)
 - Acceleration, fixed-point methods
(T. Kelly, A. Miedlar, Y. Saad, R. Schneider, H. vd. Vorst, H. Walker, ...)
 - Non-linear PDEs
(Z. Bai, E. Cancès, G. Friesecke, M. Lewin, I. Sigal, ...)
- Application domain: **Source for new methods**
 - Davidson diagonalisation (H. vd. Vorst, ...)
 - Thorough exploration of Anderson-type acceleration (see above)
- **17 minisymposia** at SIAM in 2021/22 (-CSE, -LA, -MS, -PP, -UQ) with contributions related to **electronic-structure theory**

(Exaggerative) state of codes in this field

Mathematical research

- **Goal:** Numerical experiments
- **Scope:** Reduced models
- High-level **language:**
Matlab, python, ...
- **Lifetime:** 1 paper
- **Size:** < 1k lines
- Does not care about performance

Application research

- **Goal:** Modelling physics
- **Scope:** All relevant systems
- Mix of **languages:**
C, FORTRAN, python, ...
- **Lifetime:** 100 manyears
- **Size:** 100k – 1M lines
- Obligated to write performant code

- Working with these codes requires different skillsets
 - ⇒ **Orthogonal** developer & user **communities**
- Obstacle for knowledge transfer:
 - Mathematical methods **never tried in practical setting**
(and may well not work well in the real world)
 - **Some issues cannot be studied** with mathematical codes
(and mathematicians may never get to know of them)
- What about emerging hardware, accelerators, performance?
 - Should be the regime of Computer Science (yet another community)

Difficulties of interdisciplinary research



high-performance computing

materials simulations


DFTK

numerical analysis

novel scientific models



$$H\Psi = E\Psi$$

- A social problem ...
 - Community conventions (e.g. publication culture)
 - Language barriers and context-sensitive terms
 - Speed of research (development of model vs. its analysis)
- ... **cemented in software:**
 - **Priorities differ** \Rightarrow What is considered a “good code” differs
 - Insurmountable obstacles for code integration
 - Collaborations can stop before they begin ...
- **Hypothesis: People compose if software composes**
-  **DFTK**, the Density-Functional ToolKit
 - Allows restriction to **relevant model problems**,
 - **and scale-up** to application regime (1000 electrons)
 - **Sizeable feature set** in **7500 lines** of code
 - MPI, self-adapting methods, algorithmic differentiation
 - Integrated in multi-scale pipeline (potential fitting, molecular dynamics)

Density-functional theory (insulators)

- Energy minimisation problem:

$$\min_{D \in \mathcal{P}} \mathcal{E}(D) = \min_{D \in \mathcal{P}} [\text{tr}(H_0 D) + E_{\text{Hxc}}(\text{diag } D)]$$

with $\mathcal{P} = \{D \in \mathfrak{S}_1(L^2) \mid 0 \leq D \leq 1, \text{tr}(D) = N, \text{tr}(-\Delta D) < \infty\}$, $[\text{diag } D](\underline{r}) = D(\underline{r}, \underline{r})$

- **DFT approximation:** Effective single-particle model

$$\left\{ \begin{array}{l} \forall i \in 1 \dots N : \left(-\frac{1}{2}\Delta + V(\rho_\Phi) \right) \psi_i = \varepsilon_i \psi_i, \\ V(\rho) = V_{\text{nuc}} + v_C \rho + V_{\text{XC}}(\rho), \\ \rho_\Phi = \sum_{i=1}^N |\psi_i|^2, \\ \Phi = (\psi_1, \dots, \psi_N) \in \left(L^2(\mathbb{R}^3, \mathbb{C}) \right)^N \text{ orthogonal} \end{array} \right.$$

nuclear attraction V_{nuc} , exchange-correlation V_{XC} , Hartree potential $-\Delta(v_C \rho) = 4\pi\rho$

\Rightarrow **Self-consistent field (SCF) problem:** $\rho(V(\rho)) = \rho$ with

$$\rho(V) = \text{diag} \left[\mathbb{1}_{(-\infty, \varepsilon_F]} \left(-\frac{1}{2}\Delta + V \right) \right] \quad \text{and } \varepsilon_F \text{ s. t. } \int \rho(V) = N$$

Self-consistent field problem

- Density-mixing **SCF procedure** (preconditioner P , damping α)

$$\rho_{n+1} = \rho_n + \alpha P^{-1} [\rho(V(\rho_n)) - \rho_n]$$

- In practice: Combined with **acceleration** (e.g. Anderson)
 - Dropped to simplify analysis
 - Re-introduced for numerical experiments

- Near a fixed-point the error goes as

$$e_{n+1} \simeq [1 - \alpha P^{-1} \varepsilon^\dagger] e_n$$

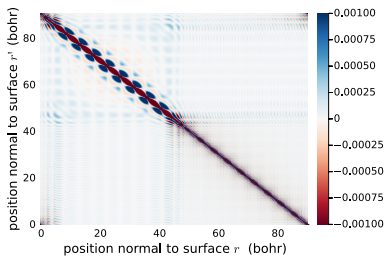
with dielectric matrix $\varepsilon^\dagger = (1 - \chi_0 K)$, $K(\rho) = V'(\rho)$, $\chi_0(V) = \rho'(V)$

- Convergence iff $-1 < [1 - \alpha P^{-1} \varepsilon^\dagger] < 1$
 - Dielectric matrix ε : **Depends on physics** (conduction, screening)
 - By second-order conditions: $\varepsilon \geq 0$ (near fixed point)

\Rightarrow Need $P^{-1} \simeq (\varepsilon^\dagger)^{-1}$ (**matching preconditioner**) or **small α**

Black-box P : Local density of states (LDOS) mixing¹

- Bulk preconditioning models approximate inverse $P^{-1} \simeq (\varepsilon^\dagger)^{-1}$
- Use $\varepsilon^\dagger = (1 - \chi_0 K)$ with $K(\rho) = V'(\rho)$, $\chi_0(V) = \rho'(V)$
- $\chi_0(r, r')$ unit-cell internal fluctuations, diagonal dominant:



- Tackle **charge sloshing**: Consider large-scale variations of χ_0 :

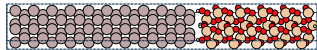
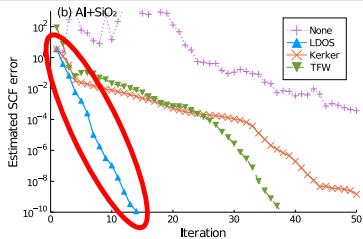
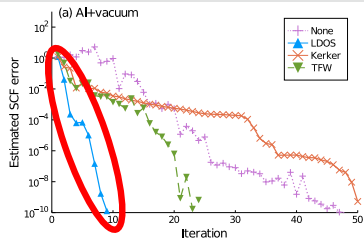
$$\chi_0(r, r') \simeq -\text{LDOS}(r)\delta(r, r') \quad (\text{homogenisation } \text{LDOS}(r) \approx \int \chi_0(r, r') dr')$$

- Apply preconditioner **iteratively**:

$$P^{-1} \rho_n = [1 - \widetilde{\chi}_0 K]^{-1} \rho_n, \quad \widetilde{\chi}_0(r, r') = -\text{LDOS}(r)\delta(r, r')$$

¹MFH, A. Levitt. J. Phys. Condens. Matter **33**, 085503 (2021).

LDOS preconditioning (examples)¹




- Inhomogeneous material: Aluminium metal + Insulator
- TFW: local Thomas-Fermi-von Weizsäcker mixing²
(Ad hoc modification of metallic screening model)
- LDOS automatically interpolates between Kerker mixing (suitable for metals) and no mixing (suitable for insulators)
 - ⇒ Based on mathematical understanding of screening
 - ⇒ Parameter-free and black-box

¹MFH, A. Levitt. J. Phys. Condens. Matter **33**, 085503 (2021).

²D. Raczkowski, A. Canning, L. W. Wang, Phys. Rev. B. **64**, 121101 (2001).

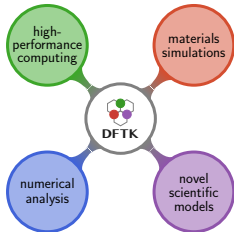
DEMO

How did  **DFTK** help us to get there?



→ https://michael-herbst.com/talks/2023.06.01_gamm_composable.html





How does DFTK achieve this?



- The magic of **Julia**:
 - Separating the **what** from the **how**
 - Clear design, inspired by mathematical structure
 - ⇒ Self-explaining code (a clear **what**)
 - Focus on **keeping code accessible** (7500 lines)
 - Started in 2018, already 30 contributors
 - Key features by undergrads & outsiders
- ⇒ High-productivity research framework
- ⇒ Supports joint **research across disciplines**

Separating the what from the how

- Why is this separation so important ...
 - ... for composable software?
 - ... **for multidisciplinary research?**
- Consider the **goal**: Implementing a new numerical scheme
- **Traditionally** users code in detail **how** the computation should proceed (Imperative programming)
 - How = architecture
 - How = linear algebra primitive (e.g. orthogonalisation)
 - How = memory layout
 - ...
- This has nothing to do with the mathematics we care about!
- Can the **how** be abstracted away?
- Let's see **julia**'s HPC developments ...

Accelerators	Shared Mem	Distributed
 CUDA.jl	 AMDGPU.jl	 OneAPI.jl
JuliaGPU/ GPUArrays.jl		
Reusable array functionality for Julia's various GPU backends. https://github.com • JuliaGPU • KernelAbstractions		
KernelAbstractions.jl - Heterogeneous programming in Julia Heterogeneous programming in Julia. Contribute to JuliaGPU/KernelAbstractions.jl development by creating an account on GitHub.		JuliaFolds/FLoops.jl
		Fast sequential, threaded, and distributed for-loops for Julia-fold for humans™ Announcing composable multi-threaded parallelism in Julia <small>23 July 2021 by Jeff Rossman (Julia Computing), Aron Ness (Julia Computing), Kien Nguyen (Julia)</small>
		Base / Multi-Threading
		Multi-Threading
		Base.Threads.@threads - Macro
		Julia Atomics Manifesto <small>This proposal aims to define the memory model of Julia and to provide certain guarantees in the presence of data races, both by default and through providing interfaces to allow the user to specify the level of guarantees required. This should allow native implementations in Julia of other system abstractions (like Boost.Atomic) to interoperate with native runtime code, and also to give generally maintainable behaviors without incurring significant performance cost. Additionally, it should be the general audience and not clear about the user to better-compatibility with respect to ensuring that an atomic-type field is accessed with proper care for synchronization.</small>
		JuliaParallel/MPI.jl MPI wrappers for Julia
		JuliaParallel/ Dagger.jl A framework for out-of-core and parallel execution
		Standard Library / Distributed Computing
		Distributed Computing



GitHub - eth-crcr/typed3@beta0ad3...
github.com



```
function power_method(A, x; niter=100)
    for i = 1:niter
        x = A * x
        x ./= norm(x)
    end
    x
end
```

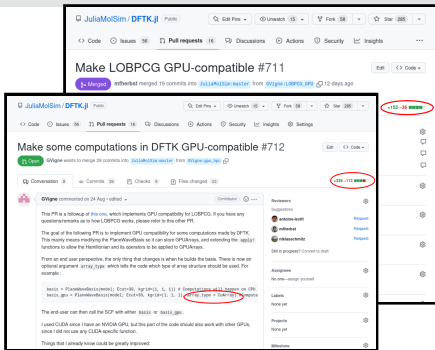
```
A = rand(10, 10); A = A + A' + 10I; x = rand(10)

using LinearMaps, IterativeSolvers
itinv(A) = LinearMap(x -> cg(A, x), size(A)...)

using CUDA
power_method(itinv(CuArray(A)), CuArray(x))

using AMDGPU
power_method(itinv(ROCArray(A)), ROCArray(x))
```

GPU support in DFTK



The screenshot shows two GitHub pull requests for the JuliaMolSim/DFTK.jl repository. The top PR, #711, is titled "Make LOBPCG GPU-compatible" and has 10 commits. The bottom PR, #712, is titled "Make some computations in DFTK GPU-compatible" and has 1 commit. Both PRs show a green "Merged" status. The bottom PR includes a comment from @Wgjn explaining the goal of implementing GPU compatibility for LOBPCG and mentioning the use of the GPUArray.jl package. A code snippet in the comment shows a Julia function call: `basis = PlaneWaveBasis(model; Ecut=30, kgrid=(1, 1, 1), architecture=DFTK.GPU(CuArray))`. The PR #712 also shows a commit message: "Add GPUArray.jl dependency".

- Use **julia**'s HPC abstractions to target all of CUDA, ROCm, oneAPI
- < 500 lines changed
- Collaboration with **julia** lab: CS, physics & maths
- 10-week GSoC project

```
basis = PlaneWaveBasis(model; Ecut=30, kgrid=(1, 1, 1),  
architecture=DFTK.GPU(CuArray))
```



- Note: **julia** allows seamless composition of
 - Floating-point agnostic code for computing arbitrary derivatives (algorithmic differentiation), guaranteed error control (intervals), etc.
 - Fast code integrating with MPI, CUDA, ...



Stress =

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

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

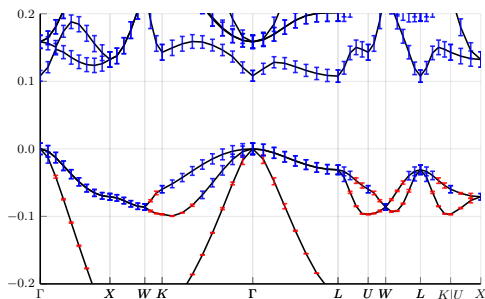
- Stress computation (Definition vs.  code)¹
- Post-processing step \Rightarrow Not performance critical
- Comparison of implementation complexity:
 -  **DFTK**: 20 lines¹ (forward-mode algorithmic differentiation)
 - Quantum-Espresso: 1700 lines²
 - \simeq 10-week GSoC project


\Rightarrow No performance impact & accessible code

¹<https://github.com/JuliaMolSim/DFTK.jl/blob/master/src/postprocess/stresses.jl>

²<https://github.com/QEF/q-e/blob/develop/PW/src>

Support of a *a posteriori* error analysis

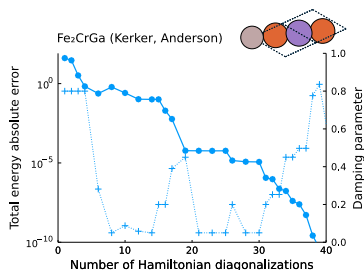


- Albeit the HPC capabilities: **Numerical experiments** are feasible
- E.g. fully guaranteed error bounds for band structures¹
- Deals with a **reduced Kohn-Sham model** and requires **interval arithmetic**
- Captures basis set error, floating-point error, convergence error
- Recent using  **DFTK** considers also density and force errors²

¹MFH, A. Levitt, E. Cancès. Faraday Discuss. **223**, 227 (2020).

²E. Cancès, G. Dusson, G. Kemplin et. al. SIAM J. Sci. Comp., **44**, B1312 (2022).

Robust & efficient algorithms



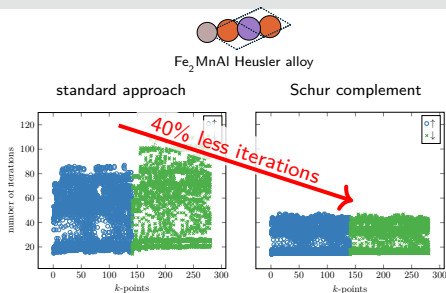
- Black-box SCF damping α^1
- α adapted *in each step* using line search & quadratic model
- Novelty: Reuse of expensive quantities in next SCF step
- Reduces trial and error

⇒ Maths / physics collaboration:

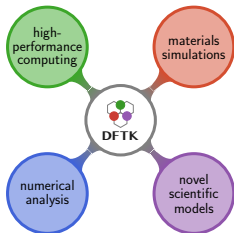
Exchange of ideas between simplified & practical settings crucial



¹MFH, A. Levitt. J. Comput. Phys. **459**, 111127 (2022).

²E. Cancès, MFH, G. Kemlin, *et. al.* Lett. Math. Phys. **113**, 21 (2023).



- First-principle properties of metals
- Schur-complement approach to perturbation theory²
(exploits partially converged states)
- ca. 40% less iterations



- Fully composable due to  abstractions:
 - Arbitrary precision (32bit, >64bit, ...)
 - Algorithmic differentiation (AD)
 - HPC tools: GPU acceleration, MPI parallelisation⇒ Accessible high-productivity research framework
- Low barriers for cross-disciplinary research:
 - Allows restriction to relevant model problems,
 - and scale-up to application regime (1000 electrons)⇒ Sizeable feature set of DFT methods in 7500 lines
 - Including some unique features (Self-adapting algorithms)
- Mathematical works with  DFTK
 - Self-adapting black-box DFT methods^{a,b}
 - Numerical analysis of DFT^{c,d}
 - Practical error bounds^{e,f}

^aMFH, A. Levitt. J. Phys. Condens. Matter **33**, 085503 (2021).

^bMFH, A. Levitt. J. Comput. Phys. **459**, 111127 (2022).

^cE. Cancès, G. Kemplin, A. Levitt. J. Matrix Anal. Appl., **42**, 243 (2021).

^dE. Cancès, MFH, G. Kemplin, et. al. Lett. Math. Phys. **113**, 21 (2023).

^eMFH, A. Levitt, E. Cancès. Faraday Discuss. **223**, 227 (2020).

^fE. Cancès, G. Dusson, G. Kemplin et. al. SIAM J. Sci. Comp., **44**, B1312 (2022)18 / 21

- Eric Cancès (École des Ponts)
- **Antoine Levitt** (Université Paris-Saclay)
- Niklas Schmitz (TU Berlin)
- Benjamin Stamm (Universität Stuttgart)
- Guillaume Vigne (Mines Paris)
- All  DFTK contributors

EPFL

XtMat



Summer of code



RWTHAACHEN
UNIVERSITY




Open PhD & PostDoc positions in the MatMat group



Possible topics include:

- **Uncertainty quantification for DFT:**
Error in data-driven DFT models, pseudopotentials, propagation to properties and MD potentials
- **Self-adapting numerical methods** for high-throughput DFT simulations
- See <https://matmat.org/jobs/>

- **Interdisciplinary research** linking maths and simulation:
 - Become part of maths & materials institutes @ EPFL


- Collaboration inside  MARVEL:
NATIONAL CENTRE OF COMPUTATIONAL RESEARCH


- Reproducible workflows & sustainable software
- Computational materials discovery
- Statistical learning methods




Questions?


https://michael-herbst.com/talks/2023.06.01_gamm_composable.pdf

 <https://matmat.org>

 mfherbst

 michael.herbst@epfl.ch

 [DFTK https://dftk.org](https://dftk.org)

 <https://michael-herbst.com/learn-julia>