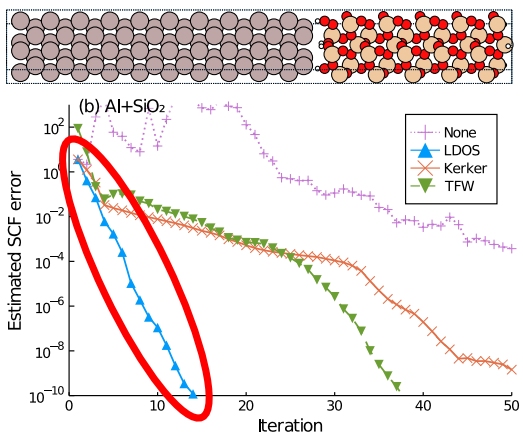
An interdisciplinary perspective on robust materials simulations

Michael F. Herbst

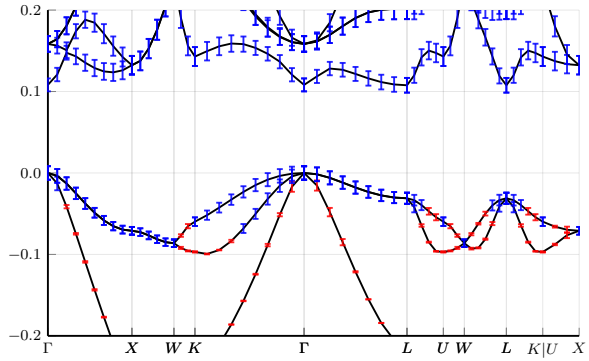
Mathematics for Materials Modelling (`matmat.org`), EPFL

8 May 2023

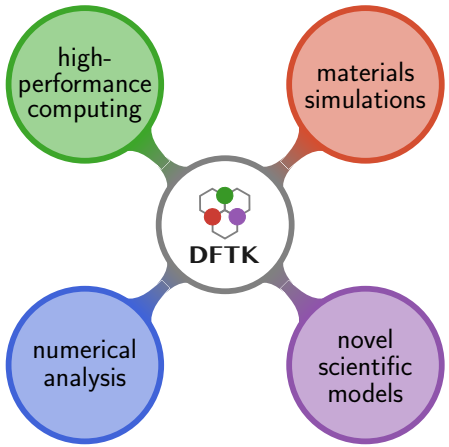
Slides: `https://michael-herbst.com/talks/2023.05.08_cis_gtn.pdf`



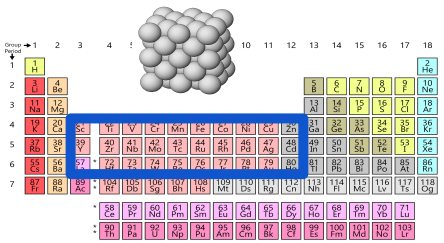
Self-adapting black-box DFT algorithms



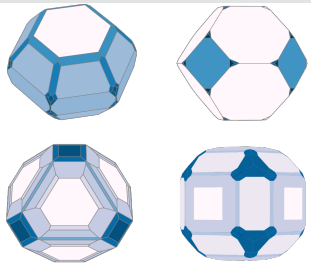
Model error in DFT simulations



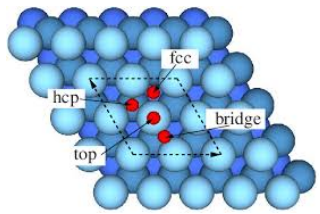
Task: Develop a new metallic catalyst for a surface reaction



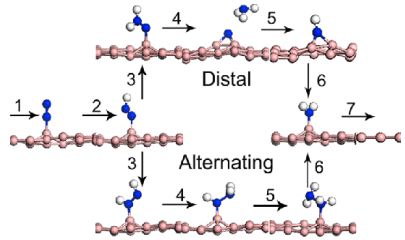
Host metal + dopant
$\simeq 30 \times 30 = 900$



Host surface
$\simeq 3 - 5$



Dopant adsorpotion site
$\simeq 30$



Reaction intermediates
$\simeq 10$

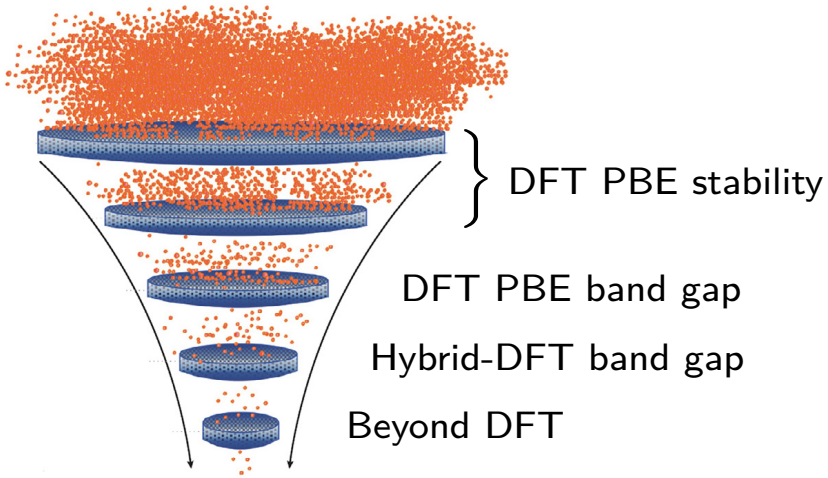
- Combinatorial design space: $\simeq 10^5 - 10^6$ possibilities
- Systematic experiments: Time and cost intensive

$\Rightarrow$ Computational screening to complement and accelerate
  - Harvest curated data bases
  - Data-driven methods and statistical learning

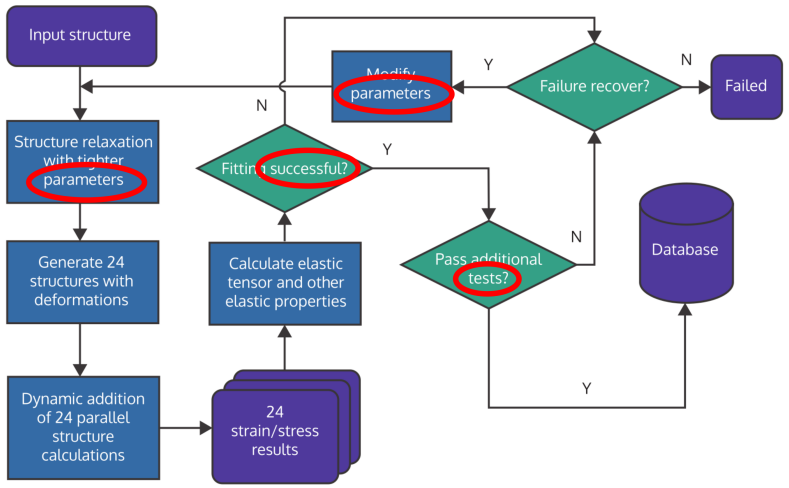
$\Rightarrow$ Regular need of millions of first-principle calculations
  - Noteworthy share of world's supercomputing resources
  - Growing list of data / workflow management tools

FireWorks AFLOW Automatic-FLOW for Materials Discovery MATERIALSCLOUD AiiDA NOMAD NOVEL MATERIALS DISCOVERY pyiron

Sketch of high-throughput workflows



Design funnel for photovoltaic materials



Workflow for computing elasticity tensors

- Many parameters to choose (algorithms, tolerances, models)
  - Elaborate heuristics: Failure rate $\simeq 1\%$
  - Still: Thousands of failed calculations

  $\Rightarrow$ Wasted resources & increased human attention (limits througput)

- **Goal:** Self-adapting black-box algorithms
  - Transform empirical wisdom to built-in convergence guarantees
  - Requires: Uncertainty quantification & error estimation

  $\Rightarrow$ Understand where and how to spend efforts best

G. Hautier Comput. Mater. Sci. **164**, 108 (2019); L. Himanen *et. al.* Adv. Science **6**, 1900808 (2019).

Broader vision: Robust & error-controlled simulations

- Error control = Track simulation uncertainties:
  - Self-adapting simulations with mathematical guarantees
  - Integrate with error propagation efforts for surrogates[1]

  $\Rightarrow$ Byproducts: Data quality control, accelerated design
- Error control = Learn missing physics:
  - Data-enhanced models, active learning
  - Integration with experiment (autonomous discovery)

  $\Rightarrow$ Exploit high-fidelity experimental, beyond-DFT data
- Error control = Leverage inexactness:
  - Error balancing: Optimal adaptive parameter selection
  - Randomised methods, selective precision (16-bit, FPGA)
  - Multi-fidelity approaches (reduced basis, surrogates)

$\Rightarrow$ Understand where and how to spend efforts best

$\Rightarrow$ Realm of mathematical research

[1] F. Musil, A. Grisafi *et. al.* J. Chem. Theo. Comput. **15**, 2 (2019).

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

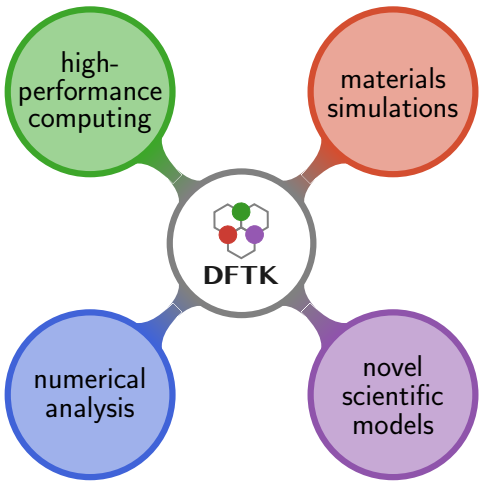
- **Goal:** Numerical experiments
- **Scope:** Reduced models
- High-level **language**:
  Matlab, python, ...
- **Lifetime:** 1 paper
- **Size:** < 1k lines
- Does not care about performance

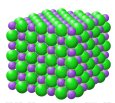
Application research

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

- Working with these codes requires different skillsets
  - $\Rightarrow$ 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

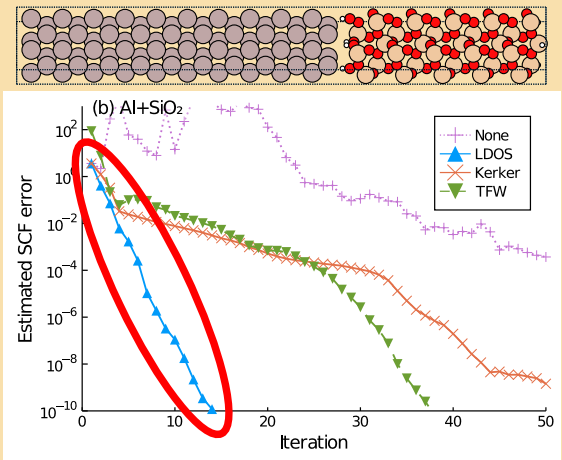




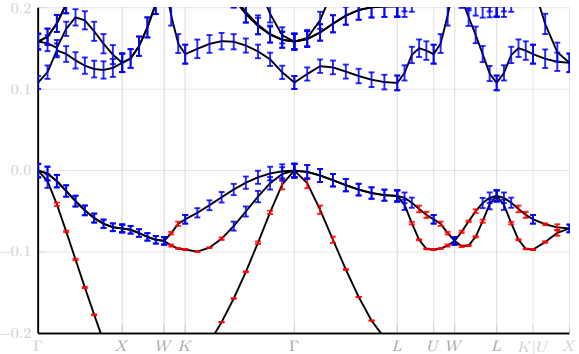
$H\Psi = E\Psi$

- 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 ⇒ 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)

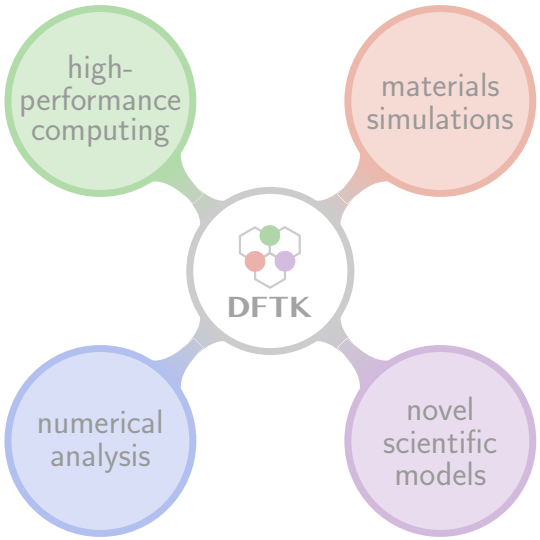
An interdisciplinary perspective on robust materials simulations



Self-adapting black-box DFT algorithms



Model error in DFT simulations



- Density-functional theory
- Local density of states preconditioner
- Adaptive damping

- Errors in DFT
- Fusing model predictions

- DFTK overview

Density-functional theory (insulators)

- Energy minimisation problem:

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

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

- DFT approximation: Effective single-particle model

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

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

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

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

Self-consistent field problem

- Potential-mixing SCF procedure (preconditioner P, damping α)

$$V_{n+1} = V_n + \alpha\, P^{-1}\left[V(\rho(V_n)) - V_n\right]$$

- 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

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

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

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

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

Drawback of established approaches

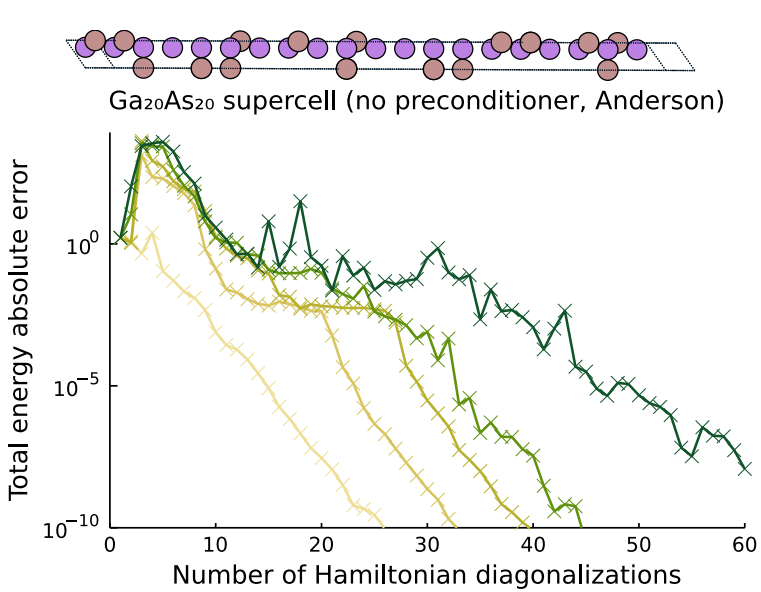
1. Preconditioner P is system-dependent and chosen *a priori*
   - Standard preconditioners: Derived from bulk materials
   - Misses important applications (e.g. inhomogeneous systems)
   - E.g. clusters, passivated surfaces, heterogeneous catalysis, ...

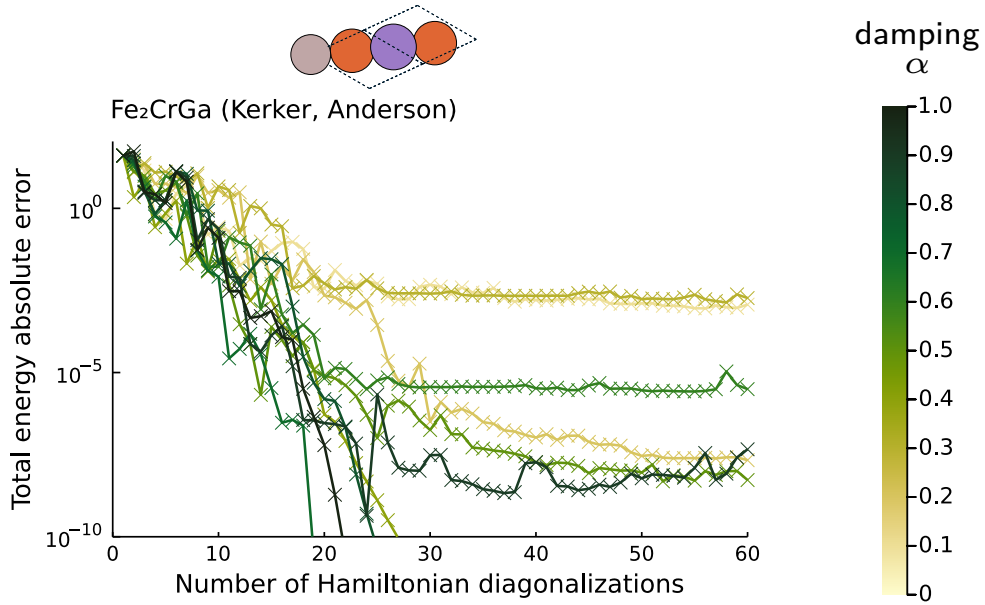
2. If no good preconditioner P known: Trial and error
   - Employ standard heuristics: E.g. **decrease damping** α
   - But: Can fail for interesting cases (the tough 1% ?)

$\Rightarrow$ Wasted computational resources

$\Rightarrow$ **Goal:** Black-box and self-adapting P and α

Illustration: Guessing a suitable damping α can be hard



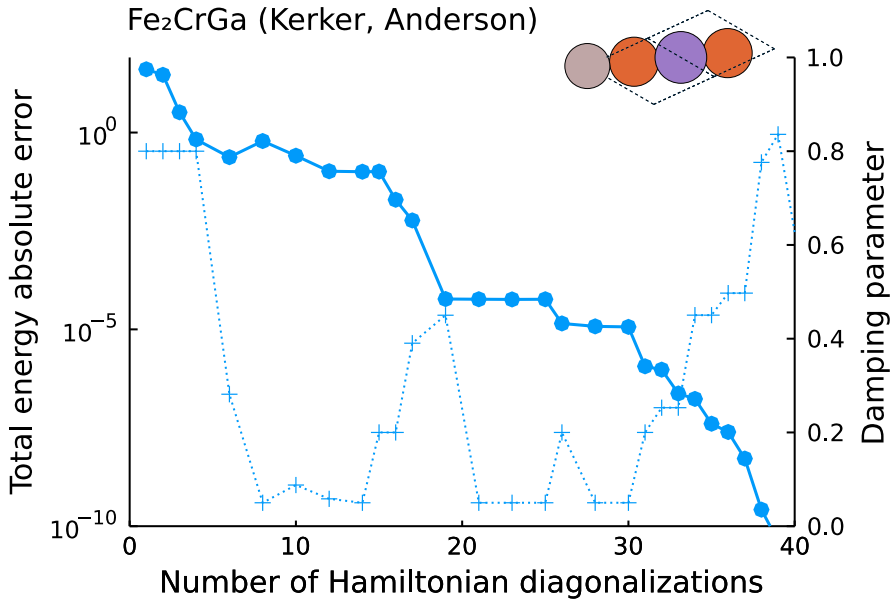


- Inefficient standard damping $(0.6 - 0.8)$
- Surprisingly small damping for smooth convergence

- Heusler alloy: Design space of interest
- Convergence difficulties found in high-throughput studies

- Irregular behaviour: α versus convergence
- Heuristics breaks: Larger damping is better

Black-box α: Adaptive damping[1]

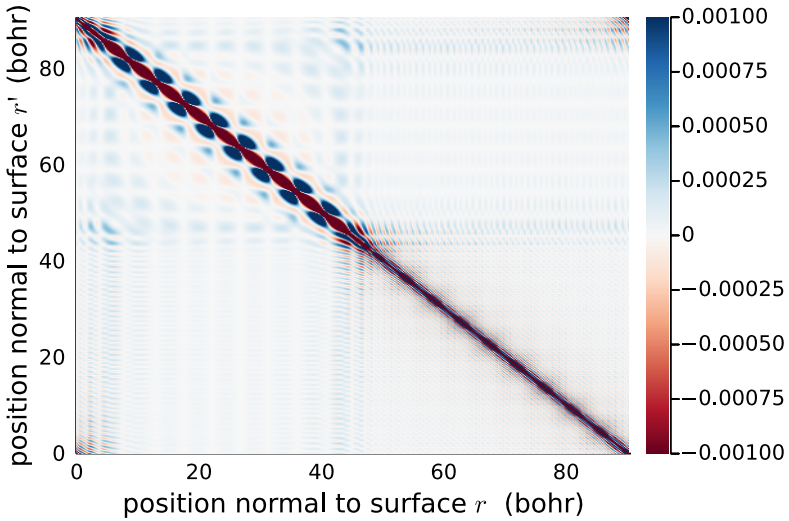


- **Theorem:** SCF convergence guaranteed if α small enough (see paper)
- α adapted *in each step* using line search & quadratic model
- Novelty: Reuse of expensive quantities in next SCF step
  $\Rightarrow$ No overhead if line search immediately successful
- For tricky systems: Adaptive damping has an overhead
  - But: Avoids trial and error
  - Mathematically motivated safeguard mechanism

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

Black-box P: Local density of states (LDOS) mixing[1]

- Bulk preconditioning models approximate inverse $P^{-1} \simeq \varepsilon^{-1}$
- Use $\varepsilon = (1 - K\chi_0)$ 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:

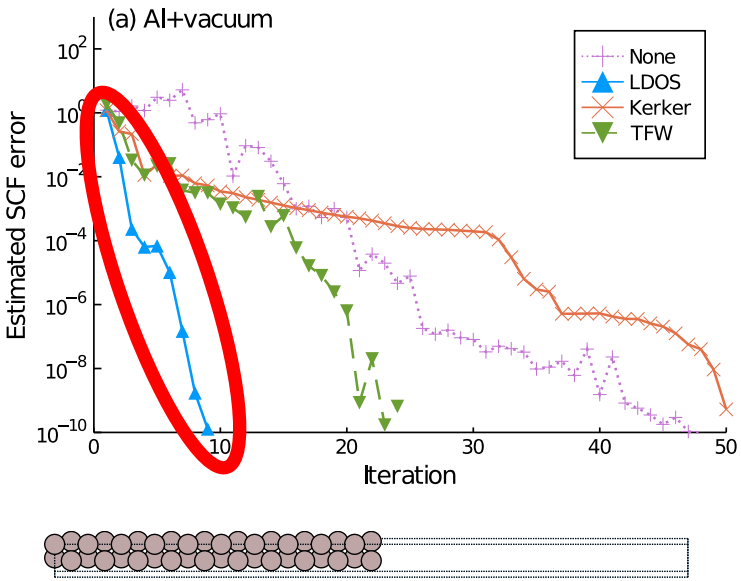
$$\chi_0(r, r') \simeq -\mathrm{LDOS}(r)\delta(r, r') \qquad (\text{homogenisation } \mathrm{LDOS}(r) \approx \int \chi_0(r, r')\,\mathrm{d}r')$$

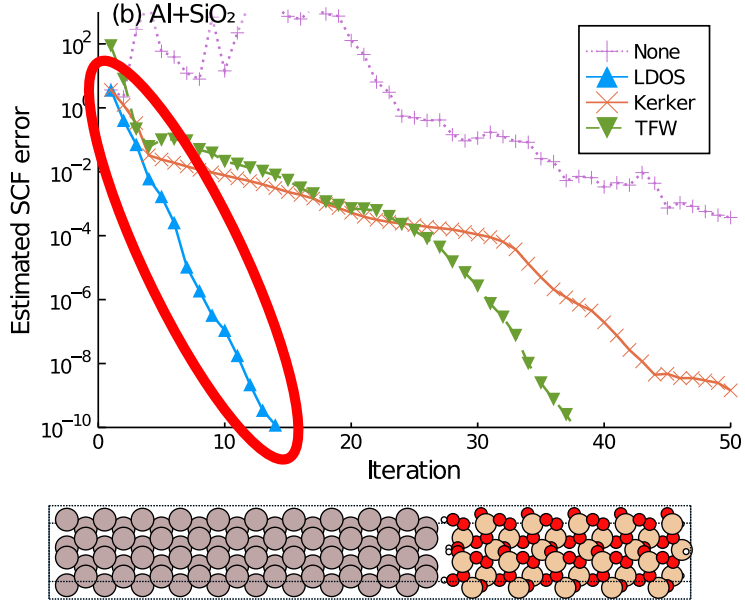
- Apply preconditioner iteratively:

$$P^{-1}V_n = [1 - K\widetilde{\chi_0})]^{-1}\,V_n, \qquad \widetilde{\chi_0}(r, r') = -\mathrm{LDOS}(r)\delta(r, r')$$

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

LDOS preconditioning (examples)[1]



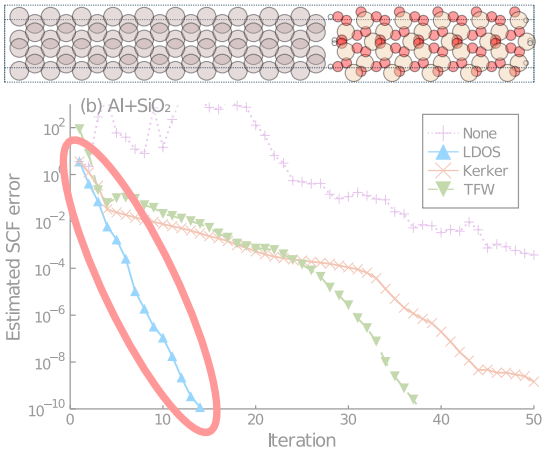


- Inhomogeneous material: Aluminium metal + Insulator
- TFW: local Thomas-Fermi-von Weizsäcker mixing[2]
  (Ad hoc modification of metallic screening model)
- LDOS automatically interpolates between Kerker mixing (suitable for metals) and no mixing (suitable for insulators)
  - ⇒ Based on mathematical understanding of screening
  - ⇒ Parameter-free and black-box

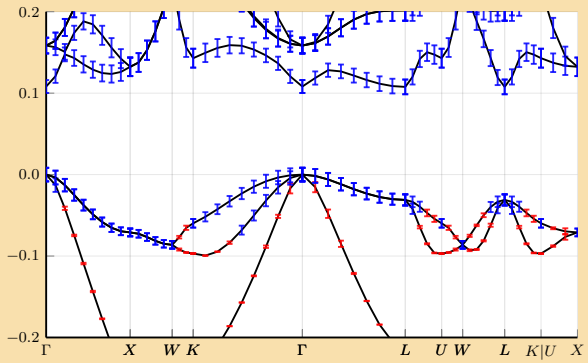
[1] MFH, A. Levitt. J. Phys. Condens. Matter **33**, 085503 (2021).

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

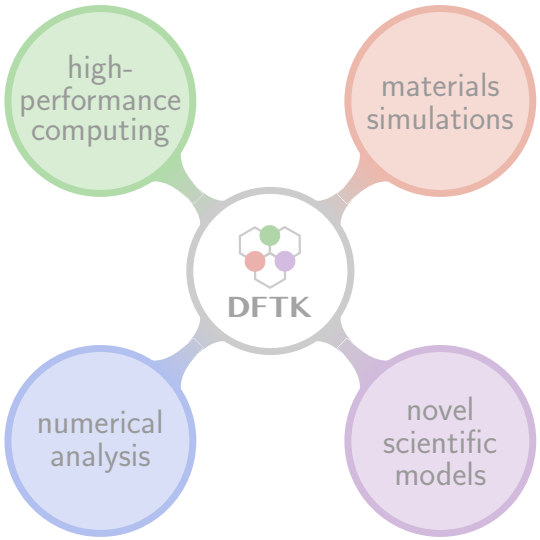
An interdisciplinary perspective on robust materials simulations



Self-adapting black-box DFT algorithms



Model error in DFT simulations



- Density-functional theory
- Local density of states preconditioner
- Adaptive damping

- Errors in DFT
- Fusing model predictions

- DFTK overview

Error sources in DFT simulations

- Model error: Selection of DFT model
- Computational approach:
  - Discretisation error: Basis size, k-point mesh
  - Algorithm error: Convergence thresholds (SCF, eigensolver)
  - Floating-point error: Floating-point arithmetic
- Additionally: Programming error, hardware error

- Discretisation error: Promising recent progress[1]
- Work on combined bounds with floating-point error[2]

- Recent work discussed here:
  - Dealing with the model error using multi-tasking surrogates

[1] E. Cancès, G. Dusson *et. al.* arxiv 2111.01470v1.

[2] MFH, A. Levitt, E. Cancès. Faraday Discus. **223**, 227 (2020).

DFT model classes

- DFT model hierarchy: Jacob's ladder
  - LDA, GGA, meta-GGA, Hybrid, RPA-like, Double Hybrid, ...
  - Each *rung* defines (parametrised) model class
  - Parameters found by fitting and/or from physics
  - Additional correction terms (+U, dispersion, counterpoise, ...)
- Higher rungs:
  - Generally more expensive
  - Generally more accurate
  - But: DFT is a non-variational approximation to exact physics

  $\Rightarrow$ No guaranteed accuracy order

- **Guiding idea:** Can we combine information from different functionals to balance accuracy / cost / deviating predictions?
- Important: We should not impose an order!

DFT model classes

- DFT model hierarchy: Jacob's ladder
  - LDA, GGA, meta-GGA, Hybrid, RPA-like, Double Hybrid, ...
  - Each *rung* defines (parametrised) model class
  - Parameters found by fitting and/or from physics
  - Additional correction terms (+U, dispersion, counterpoise, ...)
- Higher rungs:
  - Generally more expensive
  - Generally more accurate
  - But: DFT is a non-variational approximation to exact physics

  $\Rightarrow$ No guaranteed accuracy order
- **Guiding idea:** Can we combine information from different functionals to balance accuracy / cost / deviating predictions?
- Important: We should not impose an order!

Test problem: Ionisation potentials of organic molecules

- Dataset: $\simeq 3000$ small organic molecules[1]
  - ANI-1 subset (2–5 heavy atoms, a few with 6 heavies)
- Targeted quantity: Ionisation potential
  - Note: A challenging quantity for DFT
- Considered models:

	density-functional theory (DFT)			coupled cluster
model	PBE	PBE0	PBE0_DH	CCSD(T)
cost	1	1	10	1000
rung	2nd (GGA)	4th (Hybrid)	6th (double Hybrid)	*Reference*

- **Goal:** Surrogate for CCSD(T) but mostly use DFT data
- Work in progress report . . .

[1] C. Duan, F. Fang, A. Nandy, H. Kulik. J. Chem. Theo. Comput. **16**, 4373 (2020).

Delta learning: Learning to correct the error

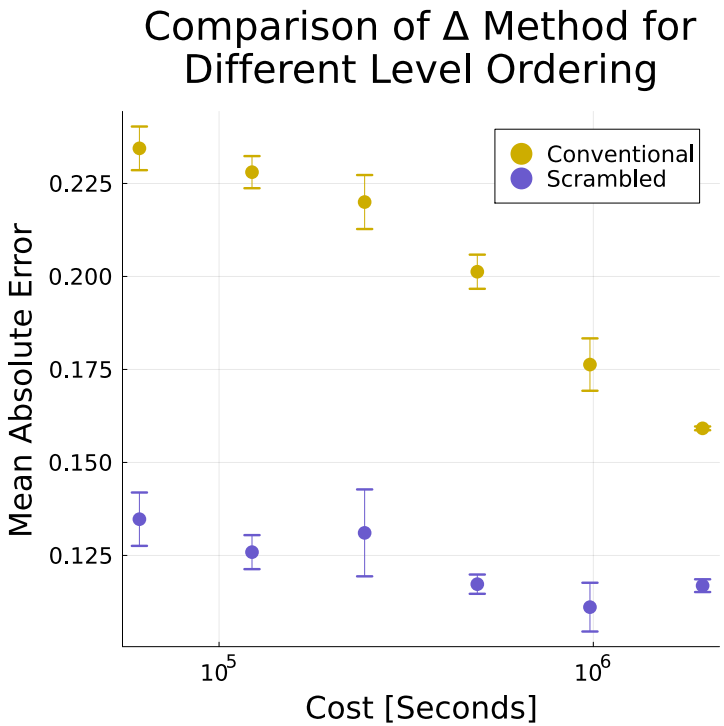
- **Idea:** Construct surrogate for *difference* between high-fidelity and low-fidelity
- Gaussian Process (GP) ansatz:

$$I^{\text{CCSD(T)}} - I^{\text{DFT}} = f(\xi) + \varepsilon$$
$$\varepsilon \sim \mathcal{N}(0, \sigma^2 I) \qquad \text{(Gaussian noise)}$$
$$f(\xi) \sim \mathcal{GP}(\mu, K_\theta) \qquad \text{(GP prior)}$$

ξ: vector of molecular descriptors, I^x: vector of simulated data,
K_θ: Kernel (e.g. polynomial, squared exponential),
σ, μ, θ hyperparameters

- **Training:** Need DFT & CCSD(T) data
- **Prediction:** Add DFT simulation to predicted mean of GP
- Apply recursively: Multiple levels

Delta learning: Discussion



- Conventional:
  PBE → PBE0 → PBE0_DH → CCSD(T)
- Scrambled:
  PBE0_DH → PBE0 → PBE → CCSD(T)
- Error bars from three draws (unit: eV)

- Data of *all* lower levels required
- Ordering matters, but the *physical one* does not always shine

Multitasking: All DFT models are equal

- Asymmetric multitasking[a] with DFT models $\alpha \in \{\text{PBE}, \text{PBE0}, \ldots\}$

$$
\begin{aligned}
I^{\text{CCSD(T)}} &= f^{\text{CCSD(T)}}(\xi) + \varepsilon^{\text{CCSD(T)}} \\
I^{\alpha} &= f^{\alpha}(\xi) + \varepsilon^{\alpha} \\
f^{\alpha}(\xi) &= \rho^{\alpha} f^{\text{CCSD(T)}}(\xi) + \delta^{\alpha}(\xi) && \text{(assumed shared structure)} \\
\varepsilon^{x} &\sim \mathcal{N}(0, \sigma_x^2 I) && \text{(Gaussian iid noise)} \\
f^{\text{CCSD(T)}}(\xi) &\sim \mathcal{GP}\left(\mu^{\text{CCSD(T)}}, K^{\text{CCSD(T)}}\right) \\
\delta^{\alpha}(\xi) &\sim \mathcal{GP}\left(\mu^{\alpha}, K^{\alpha}\right)
\end{aligned}
$$

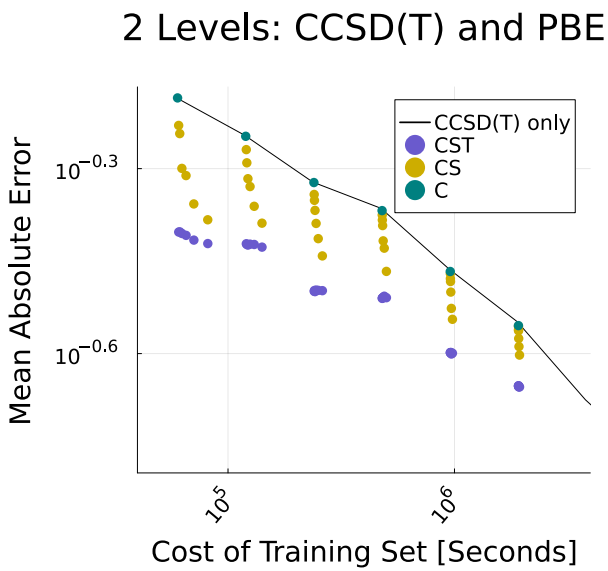
Hyperparameters: μ^x, σ_x, ρ^x & kernel params

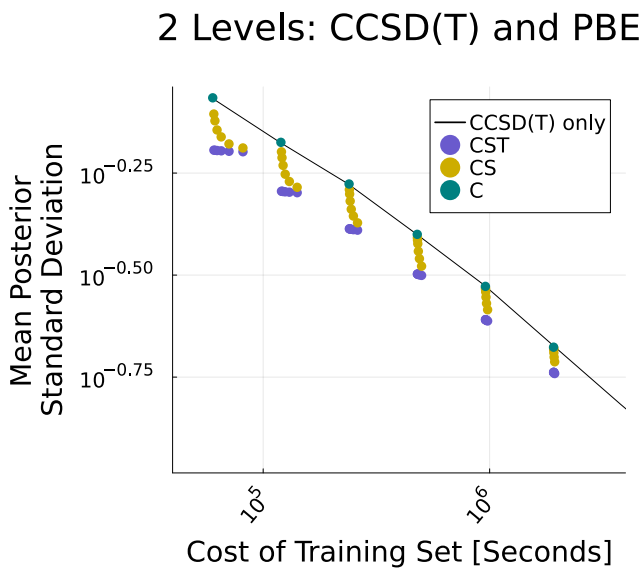
- Rationale of α-specific correlation ρ^{α} & δ^{α}:
  - Avoid negative transfer learning on $f^{\text{CCSD(T)}}$
- Assume δ^{α} independent, independent from $f^{\text{CCSD(T)}}$
  - Simplified observation covariance matrix
  - Analytical inference feasible

[a] G. Leen, J. Peltonen, S. Kaski. Mach. Learn. **89**, 157 (2012)

Multitasking: IP results (1)

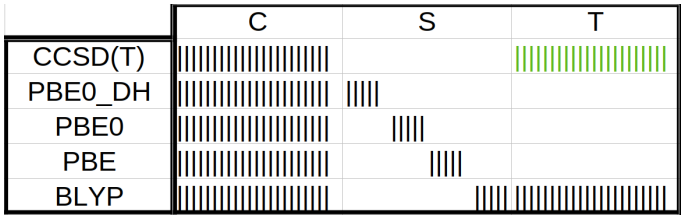
- **C**ore, **S**upplementary and **T**arget data sets
- **T** *never* available at CCSD(T) level
- Cost model: CCSD(T) is 1000-fold PBE
  (CCSD(T) scales $\mathcal{O}(N^7)$ and DFT $\mathcal{O}(N^3)$ with N number of electrons)

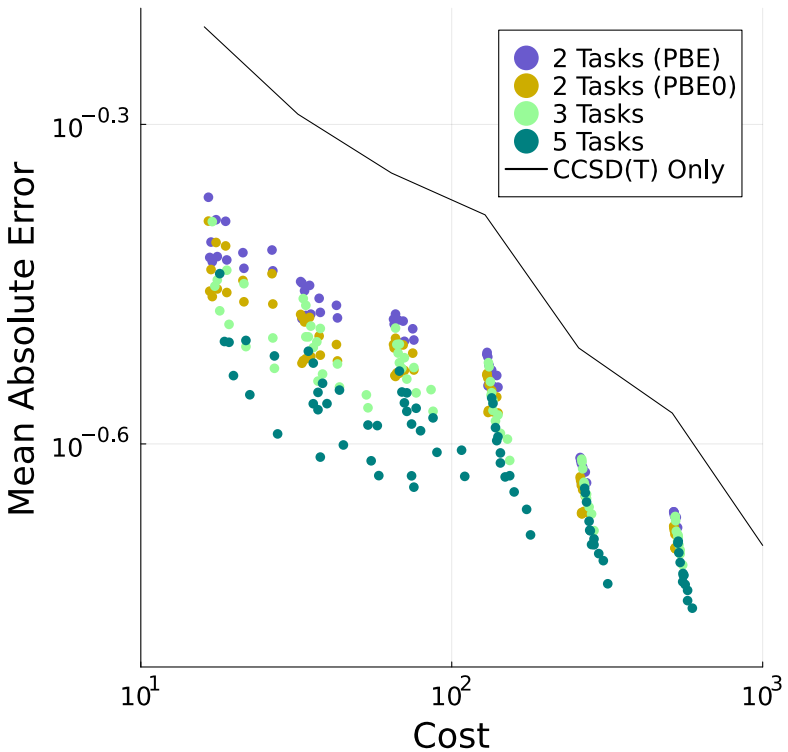




- Accuracy improved for PBE0 $\rightarrow$ CCSD(T); error in eV

Multitasking: IP results (2)

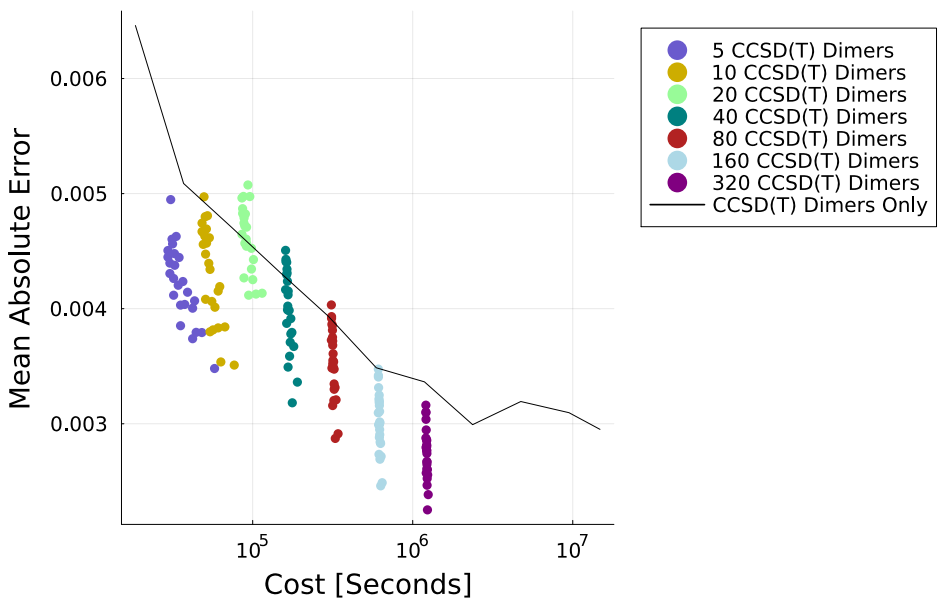




- Cost model: CCSD(T) is 50-fold DFT
  (extremely conservative)
- 500 target data points, average over 3 runs

- Lessons learned:
  - Given a CCSD(T) budget: Add DFT helps
  - To target accuracy:
    Save CCSD(T) by using (many) DFT

Multitasking outlook: Transfer learning between datasets

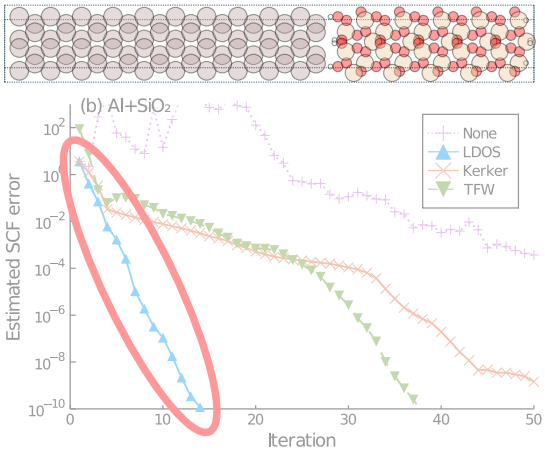


- Water monomer and water dimer
- Target: Dimer interaction energy
- $\simeq$ 6000 monomer, max. 1000 DFT dimer datapoints
- 200 target points, average over 3 runs

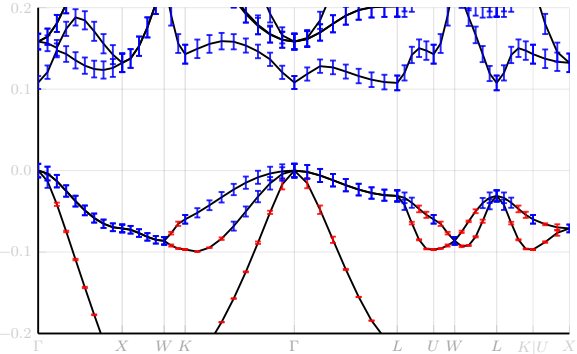
- Cost model:

	DFT	CCSD(T)
monomer	1	10
dimer	10	1000

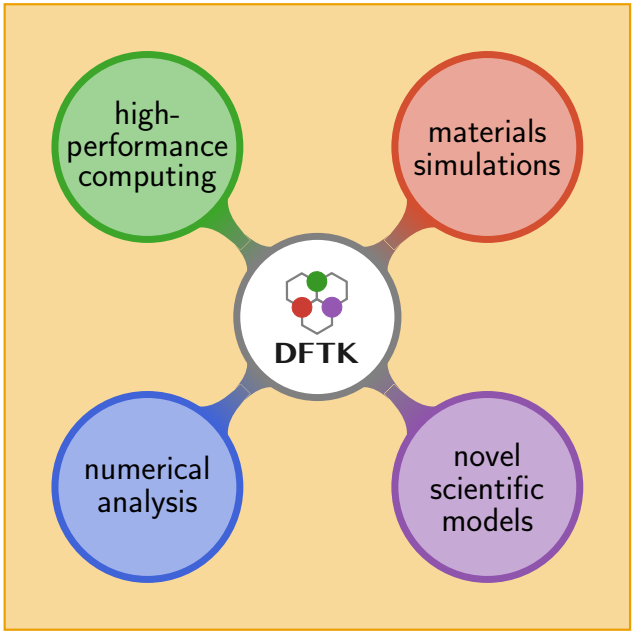
An interdisciplinary perspective on robust materials simulations



Self-adapting black-box DFT algorithms

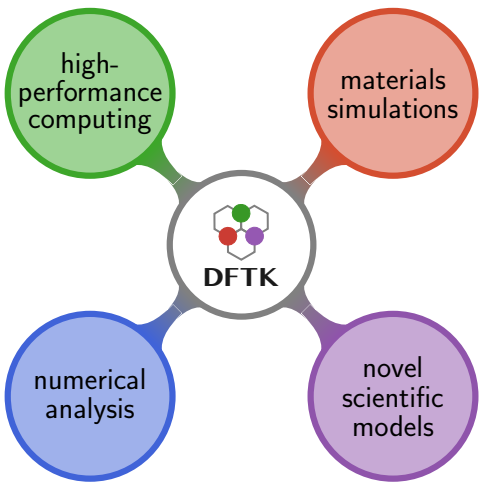


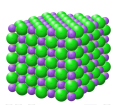
Model error in DFT simulations



- Density-functional theory
- Local density of states preconditioner
- Adaptive damping
- Errors in DFT
- Fusing model predictions
- **DFTK** overview

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





$H\Psi = E\Psi$

- julia code for plane-wave DFT, started in 2019
- Fully composable with julia ecosystem:
  - Arbitrary precision (32bit, >64bit, . . .)
  - Algorithmic differentiation (AD)
  - HPC tools: GPU acceleration, MPI parallelisation
- Key tool in all presented 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)

⇒ Build to enable multidisciplinary synergies

- Accessible high-productivity framework across domains:
  - Key code contributions by undegrads / PhD students
  - Initial AD support in 10 weeks (CS Bachelor)
  - Initial GPU support in 10 weeks (Physics Bachelor)
  - Relevant contributions from outside collab. circle

New features from generic code: Sensitivity analysis

```
function dft_energy(a, θ)
    model = model_DFT(make_structure(a), PbeExchange(θ))
    basis = PlaneWaveBasis(model; Ecut=..., kgrid=... )
    self_consistent_field(basis).energies.total
end
optimise_lattice(θ) = optimise(a -> dft_energy(a, θ))

sensitivities =
    ForwardDiff.gradient(optimise_lattice, [κ, β])
```

$$a_* = \arg\min_a \mathcal{E}(a, \theta)$$

$$\text{sensitivities} = \frac{da_*}{d\theta}$$

- Arbitrary, user-desired derivatives in one line of code
  - Three nested layers of iterative solvers
  - Almost a byproduct of our generic julia implementation
    (main addendum: Stable response solver)
  - Breaks "one PhD student per derivative" paradigm

  $\Rightarrow$ New properties/derivatives by non-DFT experts!

- Avoids combinatorial explosion
  - Future models automatically supported
  - . . . so are unusual derivatives

$\Rightarrow$ Setting the scene for new approaches:

- Sensitivity analysis & UQ, data-enhanced models

DFTK design: Keeping code concise & accessible

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

```
# 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)
)
```

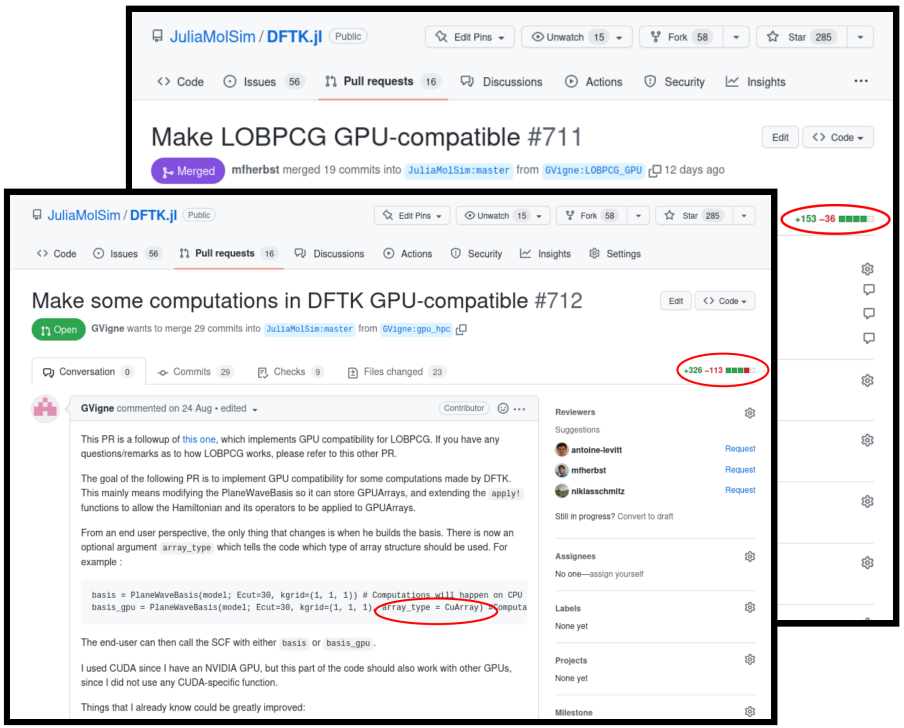
- Stress computation (Definition *vs.* julia code)[1]
- Post-processing step ⇒ Not performance critical
- Comparison of implementation complexity:
  - DFTK: 20 lines[1] (using forward-mode AD)
  - Quantum-Espresso: 1700 lines[2]
- Note: julia allows seamless composition of
  - Floating-point agnostic code for AD (slightly slower)
  - Fast code integrating with MPI, CUDA, . . .

⇒ 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

Preliminary GPU support in DFTK



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

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

DFTK: Bringing mathematical research to the applications

- Mathematical works with DFTK
  - Self-adapting black-box DFT methods[a,b]
  - Numerical analysis of DFT[c,d]
  - Practical error bounds[e,f]
- Exploring algorithmic differentiation:
  - "Automatic response": Phonons & higher-order properties
  - Data-enhanced DFT models
  - Full AD-able simulation pipeline: DFT, potentials, MD

  $\Rightarrow$ Uncertainty quantification all the way
- Part of growing julia materials modelling community
  - Common interfaces and data structures (e.g. AtomsBase)
- Outreach and teaching: DFTK summer school in 2022
  - United CS, maths, physics, chemistry, materials

Growing user base:

MIT, RWTH Aachen University, Carnegie Mellon University, École des Ponts ParisTech, Inria, UBFC Université Bourgogne Franche-Comté, TUM, VT, EPFL

[a] MFH, A. Levitt. J. Phys. Condens. Matter **33**, 085503 (2021).

[b] MFH, A. Levitt. J. Comput. Phys. **459**, 111127 (2022).

[c] E. Cancès, G. Kemlin, A. Levitt. J. Matrix Anal. Appl., **42**, 243 (2021).

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

[e] MFH, A. Levitt, E. Cancès. Faraday Discus. **223**, 227 (2020).

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

Summary

- High-throughput screening
  - Main obstacle: Large number of parameters
  - Chosen empirically ⇒ Reliability limited
- Black-box strategies for damping & preconditioning
  - Build on combining mathematical and physical insight
  - Safeguard mechanism: Increase robustness for hard cases
  - Readily available in DFTK
- Multi-tasking surrogate models
  - No need to impose model ordering ⇒ Well-suited for DFT setting
  - Can use cheap model data to compensate for expensive simulations
  - Promising to exploit existing data sets (highly heterogeneous!)
- DFTK: Multidisciplinary software development
  - julia-based framework for new DFT algorithms
  - In one code: Reduced problems and scale-up to realistic applications
  - High-productivity research framework
  - Overcome disciplinary barriers: People compose if software composes

Acknowledgements

https://michael-herbst.com/talks/2023.05.08_cis_gtn.pdf

- Eric Cancès (École des Ponts)
- **Katharine Fisher** (MIT)
- **Antoine Levitt** (Université Paris-Saclay)
- Youssef Marzouk (MIT)
- Niklas Schmitz (TU Berlin)
- Guillaume Vigne (Mines Paris)
- All  **DFTK** contributors

Summer of code

Opportunities to learn more

LauzHack event *(with G. Dalle, C. Dufour, J. Grainger, F. Wechsler)*:

"Introduction to Julia"

- **10 May 2023 18:15** BC410, EPFL & zoom
- Get a tour of the julia programming language . . .
- . . . and some free pizza

⇒ `https://memento.epfl.ch/event/introduction-to-julia/`

Upcoming seminar:

"Julia for Materials Modelling"

- **24 May 2023 15:00** MED 2 1124 & zoom
- Status of julia for materials science
- Overview of existing tools & ecosystems
- Hands-on showcases and perspectives

⇒ `https://memento.epfl.ch/event/julia-for-materials-modelling-2/`

Questions?

mfherbst

https://matmat.org

michael.herbst@epfl.ch

DFTK https://dftk.org

julia https://go.epfl.ch/julia