DFTK: A Julian approach for simulating electrons in solids

Michael F. Herbst, Antoine Levitt, Eric Cancès

CERMICS, Inria Paris and École des Ponts ParisTech

29th July 2020

MATHerials

Inria

École des Ponts ParisTech

→ `https://michael-herbst.com/talks/2020.07.29_juliacon_dftk.pdf`

Contents

Contents

Why care about electrons?

- Electrons glue the world together
- Electrons keep the world apart

Why care about electrons?

- Electrons glue the world together
- Electrons keep the world apart

Materials and semiconductors

CC-by-3.0 https://en.wikipedia.org/wiki/File:Silicon.jpg

Chemical and pharmaceutical industry

CC-by-1.0 https://en.wikipedia.org/wiki/File:Lab_bench.jpg

Aren't experiments good enough?

- Experiments are expensive (money, people, time)
- 1 droplet water[1]: $1.7 \cdot 10^{21}$ particles
- Experiments only measure averages
- Sometimes hard to link to physical laws

$\Rightarrow$ Cooperative research of experiment and theory

$\Rightarrow$ Standard practice in industry and research

[1] Assume $0.05\,\mathrm{ml}$.

Electronic structure theory

- Goal: Electronic properties (conductivity, photoactivity, . . .)
  - Construct physically sound models
  - Validate / help to interpret experiment
- Sketching the approach:
  - Regime of quantum mechanics
  - System: Hamiltonian $\hat{\mathcal{H}}$, differential operator
  - Minimisation problem: Ground state Ψ with energy

$$E = \min_{\Psi} \int_{\mathbb{R}^{3N}} \Psi\left(\underline{\boldsymbol{r}}_1, \underline{\boldsymbol{r}}_2, \ldots, \underline{\boldsymbol{r}}_N\right) \hat{\mathcal{H}} \Psi\left(\underline{\boldsymbol{r}}_1, \underline{\boldsymbol{r}}_2, \ldots, \underline{\boldsymbol{r}}_N\right) \mathrm{d}\underline{\boldsymbol{r}}_1 \cdots \mathrm{d}\underline{\boldsymbol{r}}_N$$

  - Properties: Derivatives of the energy

Density-functional theory

$$E = \min_{\Psi} \int_{\mathbb{R}^{3N}} \Psi\left(\underline{\boldsymbol{r}}_1, \underline{\boldsymbol{r}}_2, \ldots, \underline{\boldsymbol{r}}_N\right) \hat{\mathcal{H}} \Psi\left(\underline{\boldsymbol{r}}_1, \underline{\boldsymbol{r}}_2, \ldots, \underline{\boldsymbol{r}}_N\right) \mathrm{d}\underline{\boldsymbol{r}}_1 \cdots \mathrm{d}\underline{\boldsymbol{r}}_N$$

- Challenge: Size of N
- 2 Silicon atoms: $N = 28 \Rightarrow \mathbf{2}^{84} \approx 2 \cdot 10^{25}$ quadrature points

$\Rightarrow$ Finished in 1 year:

$\Rightarrow$ Density-functional theory (DFT) approximation

- Effective one-particle model ($N = 1$)
- May construct DFT model for specific context
- Discretisation basis: Build known physics into model
- But: Non-convex, non-linear minimisation

Density-functional theory

$$E = \min_{\Psi} \int_{\mathbb{R}^{3N}} \Psi\left(\underline{\boldsymbol{r}}_1, \underline{\boldsymbol{r}}_2, \dots, \underline{\boldsymbol{r}}_N\right) \hat{\mathcal{H}} \Psi\left(\underline{\boldsymbol{r}}_1, \underline{\boldsymbol{r}}_2, \dots, \underline{\boldsymbol{r}}_N\right) \mathrm{d}\underline{\boldsymbol{r}}_1 \cdots \mathrm{d}\underline{\boldsymbol{r}}_N$$

- Challenge: Size of N
- 2 Silicon atoms: $N = 28 \Rightarrow \mathbf{2}^{84} \approx 2 \cdot 10^{25}$ quadrature points

$\Rightarrow$ Finished in 1 year: ≈ 1.5 attoseconds per quadrature point

$\Rightarrow$ Density-functional theory (DFT) approximation

- Effective one-particle model ($N = 1$)
- May construct DFT model for specific context
- Discretisation basis: Build known physics into model
- But: Non-convex, non-linear minimisation

Density-functional theory

$$E = \min_{\Psi} \int_{\mathbb{R}^{3N}} \Psi\left(\underline{\boldsymbol{r}}_1, \underline{\boldsymbol{r}}_2, \ldots, \underline{\boldsymbol{r}}_N\right) \hat{\mathcal{H}} \Psi\left(\underline{\boldsymbol{r}}_1, \underline{\boldsymbol{r}}_2, \ldots, \underline{\boldsymbol{r}}_N\right) \mathrm{d}\underline{\boldsymbol{r}}_1 \cdots \mathrm{d}\underline{\boldsymbol{r}}_N$$

- Challenge: Size of N
- 2 Silicon atoms: $N = 28 \Rightarrow \mathbf{2}^{84} \approx 2 \cdot 10^{25}$ quadrature points

$\Rightarrow$ Finished in 1 year: ≈ 1.5 attoseconds per quadrature point

$\Rightarrow$ Density-functional theory (DFT) approximation

- Effective one-particle model ($N = 1$)
- May construct DFT model for specific context
- Discretisation basis: Build known physics into model
- But: Non-convex, non-linear minimisation

Self-consistent field procedure

- Euler-Lagrange equations (DFT):

$$\begin{cases} \hat{\mathcal{F}}_\rho = -\frac{1}{2}\Delta + V_\rho \\ \rho(\underline{\boldsymbol{r}}) = \sum_i f_{\varepsilon_F}(\varepsilon_i)\,|\psi_i(\underline{\boldsymbol{r}})|^2 \quad \text{with } \hat{\mathcal{F}}_\rho \psi_i = \varepsilon_i \psi_i, \\ \qquad \varepsilon_F \text{ chosen such that } \int \rho \, \mathrm{d}\underline{\boldsymbol{r}} = N, \\ \qquad \text{and } f_{\varepsilon_F}(x) = \left[1 + \exp\left(\frac{x-\varepsilon_F}{T}\right)\right]^{-1} \end{cases}$$

- Self-consistent field procedure (SCF):
  (1) Guess initial density ρ
  (2) Build Kohn-Sham operator $\hat{\mathcal{F}}_\rho$
  (3) Diagonalise it to get new $\{\psi_i\}_i$
  (4) Build new ρ go to (2).

Contents

Landscape of DFT codes

- `https://www.vasp.at`
- `https://www.abinit.org`
- `http://www.castep.org`
- `https://wiki.fysik.dtu.dk/gpaw`
- `https://www.quantum-espresso.org`
- `https://crd-legacy.lbl.gov/~chao/KSSOLV`
- ...

- Represents hundreds of man-years of coding!

$\Rightarrow$ Why bother with **DFTK**?

Interdisciplinary research field

- **Mathematicians:** Toy models and unphysical edge cases
- **Scientist:** Wants to focus on science, not numerics
- **High-performance person:** Exploit hardware specialities
- **Practitioner:** Reliable, black-box, high-level interface

- Typical obstacles:
  - Difficult problem ⇒ Often complex codes
  - Hard-coded: Workflow / algorithms / hardware optimisations
  - Huge code bases (1M lines and beyond)
  - Non-standard input syntax and API
  - Unusual systems frequently require hand-tuning
  - Two-language problem: Where to cut?

DFTK — https://dftk.org

- 14 months of development, ≈ 5000 lines
- julia code (modulo required Python and C libraries)
- Sizeable feature list:
  - Multi-level threading
  - 1D / 2D / 3D systems
  - Compose your own model
  - Integration with materials-related Python modules
  - > 500 electrons
- Performance: Within factor 2ish of established codes
- Platform for multidisciplinary collaboration
- Documentation and examples: `https://docs.dftk.org`

14 months and 5000 lines

- How did we manage . . .
- . . . well, there's the awesome julia community & ecosystem

14 months and 5000 lines

- How did we manage . . .
- . . . most we needed was there:
  - Fourier transforms (`FFTW`)
  - Linear solvers (`IterativeSolvers`)
  - Non-linear solvers (`Optim`, `NLsolve`)
  - High-level data (`PeriodicTable`, `Primes`)
  - Building blocks (`Roots`, `LineSearches`, `LinearMaps`, . . .)
  - Interfacing (`PyCall`)

14 months and 5000 lines

- How did we manage …
- … most we needed was there:
  - Fourier transforms (`FFTW`)
  - Linear solvers (`IterativeSolvers`)
  - Non-linear solvers (`Optim`, `NLsolve`)
  - High-level data (`PeriodicTable`, `Primes`)
  - Building blocks (`Roots`, `LineSearches`, `LinearMaps`, …)
  - Interfacing (`PyCall`)
- Thank you!

Contents

Lowering the entrance barrier for researchers

- Money: Always tight
- Time: 3-ish years for a PhD, master even less
- State of the art:
  - *Some* codes require software licences $\mathcal{O}$(5k€)
  - Usage: Input format and interface
  - Development: Scarce tests, comments, documentation
  - 1M lines of hardly uniform code conventions
  - Original developers have left (PhD is over)

Attempts to lower the barriers in DFTK

- julia: Zero cost and great learning resources
- Design goal: Code follows mathematical structure of DFT
- Aim for best agreement between code and equations (Unicode)
- Comments: Hint derivation or point to original articles
- `https://docs.dftk.org` with plenty of usage example
- Example projects:
  - Publication following master project
  - 8-week student project to toy with GPUs in DFT

  $\Rightarrow$ Both cases: No familiarity with DFT or julia
  - Error estimates: 10 weeks to publication

High-throughput screening

- *In silico* design of novel materials
- Challenging classifiers: Band gap, excitation energies, ...
- Narrow down 10k candidates to $\mathcal{O}(10)$
- Requirements:
  - Tunable parameters: Accuracy *versus* speed
  - Ideally: Target accuracy drives black-box workflow
  - Reliability: Breakdown of SCF not acceptable
  - Scriptability and interfacing to data science tools
  - Exploit whatever hardware exists (GPU, accelerators) ...

Screening and state of the art

- State of the art:
  - Plenty accuracy-related parameters
  - Chosen by experience: Too tight *versus* too optimistic
  - API decided *a priori* (two-language problem)

  $\Rightarrow$ Decisions hard-coded (e.g. floating-point type)

  $\Rightarrow$ Little freedom to tweak algorithms

  $\Rightarrow$ *Basically* hard fork for every accelerator / architecture

$\Rightarrow$ One long-term driving force behind DFTK

A posteriori error analysis in DFT

- *A posteriori* error: Upper bound how far solution is off
- Mathematical answer for: Has target accuracy been reached?
- Sources of error in DFT:
  - Model error
  - Discretisation error
  - Algorithm error
  - Arithmetic error
- *A posterior* bound on error $\Rightarrow$ Automatic error balancing
- For DFT: Full picture not yet understood

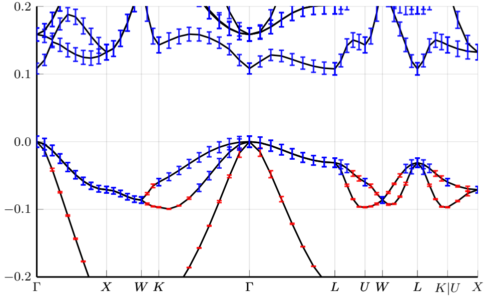
A posteriori error analysis with DFTK

- Requirements:
  - Mathematical theory can only treat reduced models
  - Step-by-step expansion
  - Ingredients not yet clear (e.g. form of integrals, derivatives)

  $\Rightarrow$ Need accessible toolbox for experimentation
  - Arithmetic error: Interval arithmetic, elevated precision
- DFTK offers:
  - Fully customisable model
  - Support for arbitrary floating-point types
  - Use julia ecosystem on DFTK datastructures:
    - Numerical quadrature, forward-mode AD, ...

    $\Rightarrow$ Rapid prototyping in numerical linear algebra

A posteriori error analysis: First results[1]



- Reduced model: Non-self-consistent Kohn-Sham
- Estimation of arithmetic error (`IntervalArithmetic.jl`)
- Used elevated floating-point type (`DoubleFloats.jl`)

- Time to submission: 10 weeks

[1]M. F. Herbst, A. Levitt and E. Cancès. Faraday Discuss. *In press.* (2020)

Reliable SCF algorithms

- Convergence of SCF depends on dielectric properties

$\Rightarrow$ Different SCF needed for metals, insulators, semiconductors

- Current schemes based on results for bulk materials

$\Rightarrow$ What to do e.g. for surfaces?

- Requirements:
  - 1D / 2D / 3D: Analyse spectral properties
  - Rapid prototyping to mix and match ideas

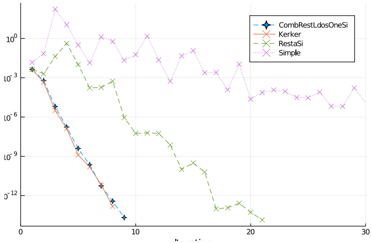
  $\Rightarrow$ High-level code inside key algorithms

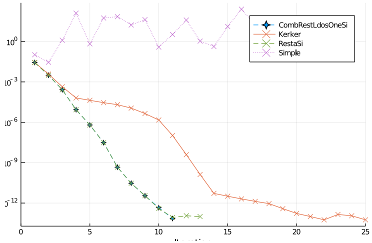
  - Testing requires realistic systems, but first version never works

  $\Rightarrow$ Scalup should not imply a rewrite of toy code

Proposing SCF algorithms with DFTK (WIP)

- Initial developments on 1D and 2D systems
- Compute spectral properties with `KrylovKit.jl`
- Tested identical implementation on systems > 500 electron
- Currently 4th refinement iteration:



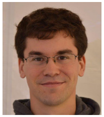


- Aluminium (10 repeats)
- Silicon (10 repeats)

Summary and outlook

- DFTK: Interdisciplinary software development
  - Closely integrated with julia ecosystem
  - Mix and match to build new algorithms
  - Rapid prototyping and toy problems
  - Scale-up for realistic testing and applications
- Near-future steps
  - SCF schemes for challenging systems (e.g. spin)
  - Error estimates and black-box workflows for DFT
  - Mixed precision and multigrid methods
  - Methods beyond DFT and SCF
  - Full GPU integration

Acknowledgements


Antoine Levitt

Eric Cancès

Other DFTK contributors:

- `@gkemlin`, `@ssirajdine`, `@louisponet`

julia & package developers

- `DoubleFloats.jl`
- `IntervalArithmetic.jl`
- `NLsolve.jl`
- `Optim.jl`
- `Plots.jl`
- `PyCall.jl`
- `Roots.jl`
- `StaticArrays.jl`
- `$THOSE_I_FORGOT`

Questions?

DFTK `https://dftk.org`

mfherbst

`https://michael-herbst.com/blog`

`michael.herbst@inria.fr`