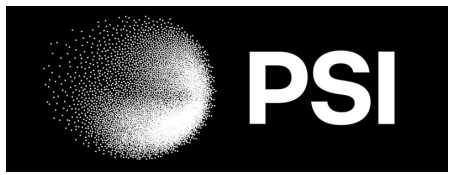
DFT in atom, pseudopotential generation and Julia implementation

Jusong Yu

October 15, 2024



Motivation

- The reference of optimization procedure is AE calculation and only need to run one time. It can take long time (> 1 min) for meta-GGA functionals.
- Native interfacing with Julia, where ML libraries in the community shine.
- An implementation of pseudopotential generation that easy to understand and easy to extend.
- Consolidate what I learned about pseudopotential theory.
- Consolidate what I learned about DFT.
- Consolidate what I learned about Julia.

Motivation

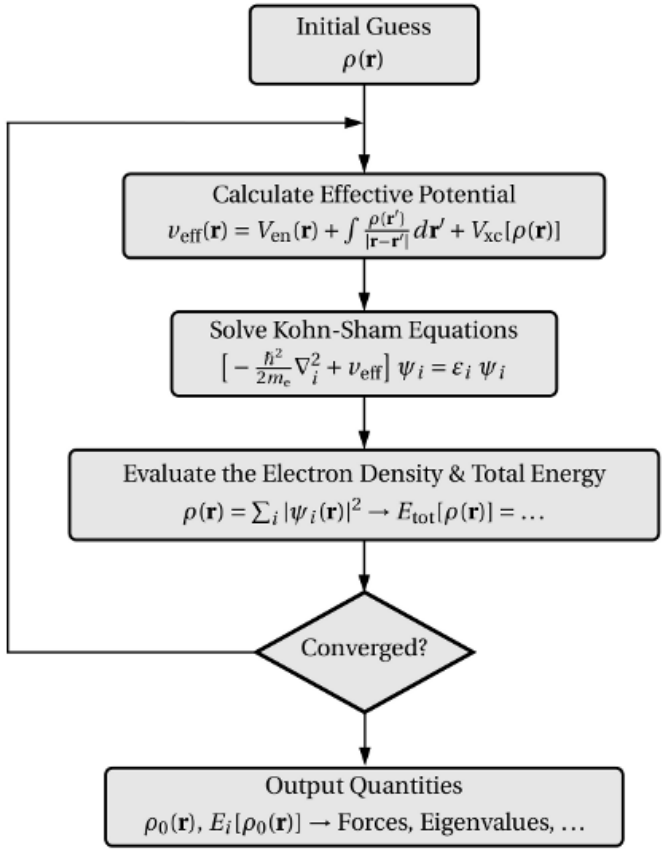
- The reference of optimization procedure is AE calculation and only need to run one time. It can take long time (> 1 min) for meta-GGA functionals.
- Native interfacing with Julia, where ML libraries in the community shine.
- An implementation of pseudopotential generation that easy to understand and easy to extend.
- Consolidate what I learned about pseudopotential theory.
- Consolidate what I learned about DFT.
- Consolidate what I learned about Julia.

Outline

Theory and equations

Implementation in Julia

DFT SCF is DFT SCF



* A flow chart of the DFT SCF iteration scheme. The initial guess in implementation is usually Thomas-Fermi potential (electrons behave like a uniform, non-interacting gas in a potential field).

Schrödinger equations in atom

The 3D Schrödinger equation is given by (in Hartree atomic units, through all following slides):

$$\left(-\frac{1}{2}\nabla^2 + V(\mathbf{x})\right)\psi(\mathbf{x}) = E\psi(\mathbf{x}), \tag{1}$$

For a spherically symmetric potential:

$$V(\mathbf{x}) = V(r), \tag{2}$$

Radial part can be separated and dependent on n, l, we then re-write to radial Schrödinger equation:

$$-\frac{1}{2}\left(r^2 R'_{nl}(r)\right)' + \left(r^2 V + \frac{1}{2}l(l+1)\right) = Er^2 R_{nl}(r). \tag{3}$$

From 2nd-order to 2 x 1st-order

2st-order radial Schrödinger equation

$$-\frac{1}{2}\left(r^2 R'_{nl}(r)\right)' + \left(r^2 V + \frac{1}{2}l(l+1)\right) R_{nl}(r) = E r^2 R_{nl}(r).$$

Substitute $P_{nl}(r) = rR_{nl}(r)$ and $Q_{nl}(r)=P'_{nl}(r)$ to convert it to two 1st-order equations:

$$P'_{nl}(r) = Q_{nl}(r), \tag{4}$$

$$Q'_{nl}(r) = 2\left(V(r) - E + \frac{l(l+1)}{2r^2}\right) P_{nl}(r). \tag{5}$$

Radial Dirac equation

$$P'_{n\kappa}(r) = -\frac{\kappa}{r} P_{nk}(r) + \left(\frac{E - V(r)}{c} + 2c \right) Q_{n\kappa}(r), \tag{6}$$

$$Q'_{n\kappa}(r) = -\left(\frac{E - V(r)}{c} \right) P_{n\kappa}(r) + \frac{\kappa}{r} Q_{n\kappa}(r), \tag{7}$$

where κ is to distinguish l and s by,

$$\kappa = \begin{cases} -l - 1 & \text{for } j = l + \frac{1}{2}, \text{ i.e. } s = +1, \\ l & \text{for } j = l - \frac{1}{2}, \text{ i.e. } s = -1. \end{cases} \tag{8}$$

Representation in 1D

- In solid, it is fair to apply the approximation that the potential around atom has spherical symmetry. Therefore, even for a open-shell configuration, a uniform distribution of electrons among the available orbitals is implicitly assumed.
- Most atomic structure codes use “logarithmic” mesh as it gives an efficient concentration near the core where densities and potentials vary most rapidly.

Solving eigen problem by integration (I)

The wavefunction, potential, and density are directly sampled. Starting from the radial Schrödinger equation we are targeting to find the discrete spectrum E for which the wave functions obey the Dirichlet boundary conditions, from which the asymptotic behavior at origin is defined

$$\lim_{r \to 0} \to 0, \qquad \lim_{r \to \infty} \to 0, \tag{9}$$

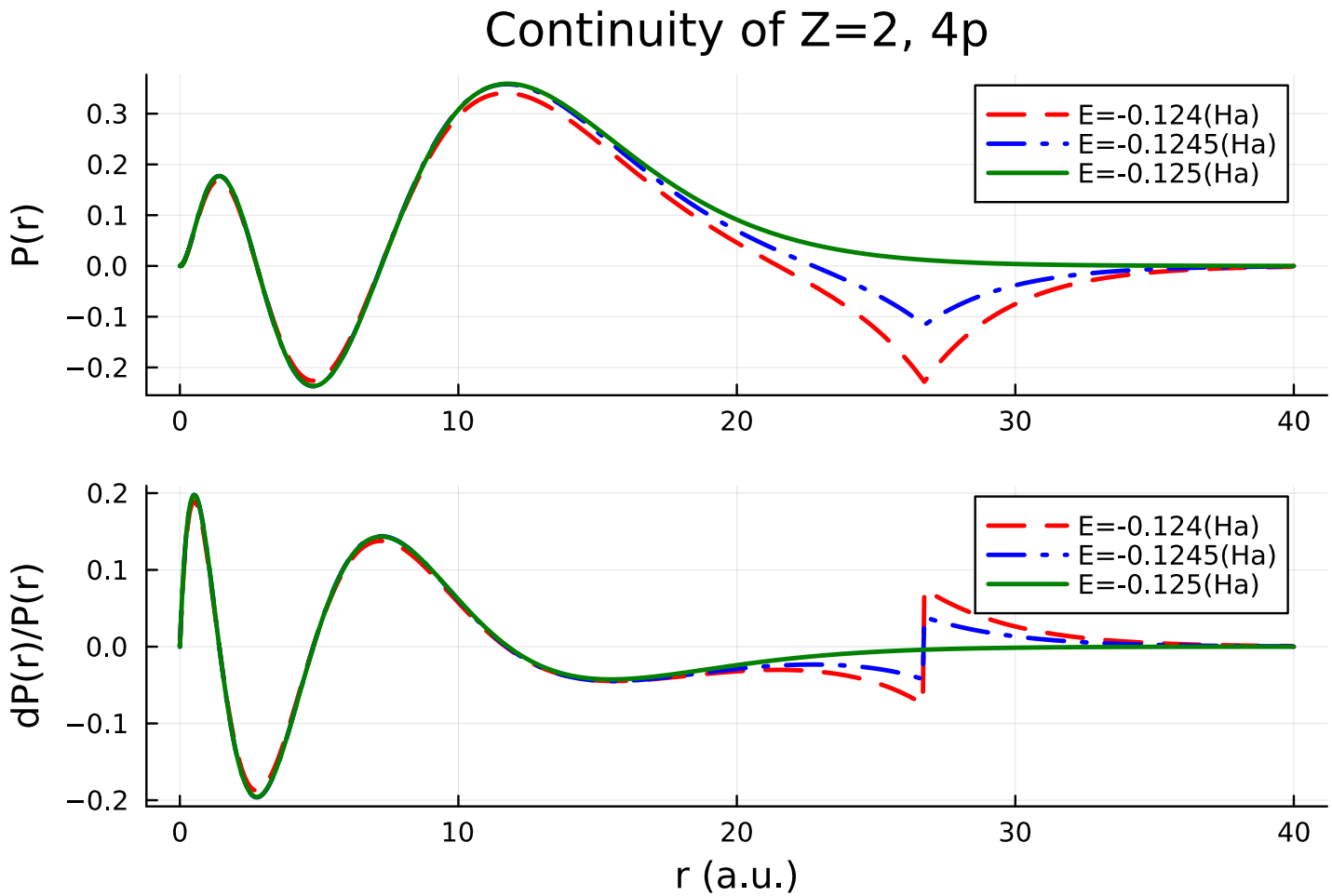
* There are also finite-difference solver and Numerov's method for solving 2nd-ODE. But for pseudopotential generation, those methods not generic on further evaluating the pseudo-wave properties e.g. logarithmic-derivative.

Solving eigen problem by integration (II)

Impose the boundary conditions

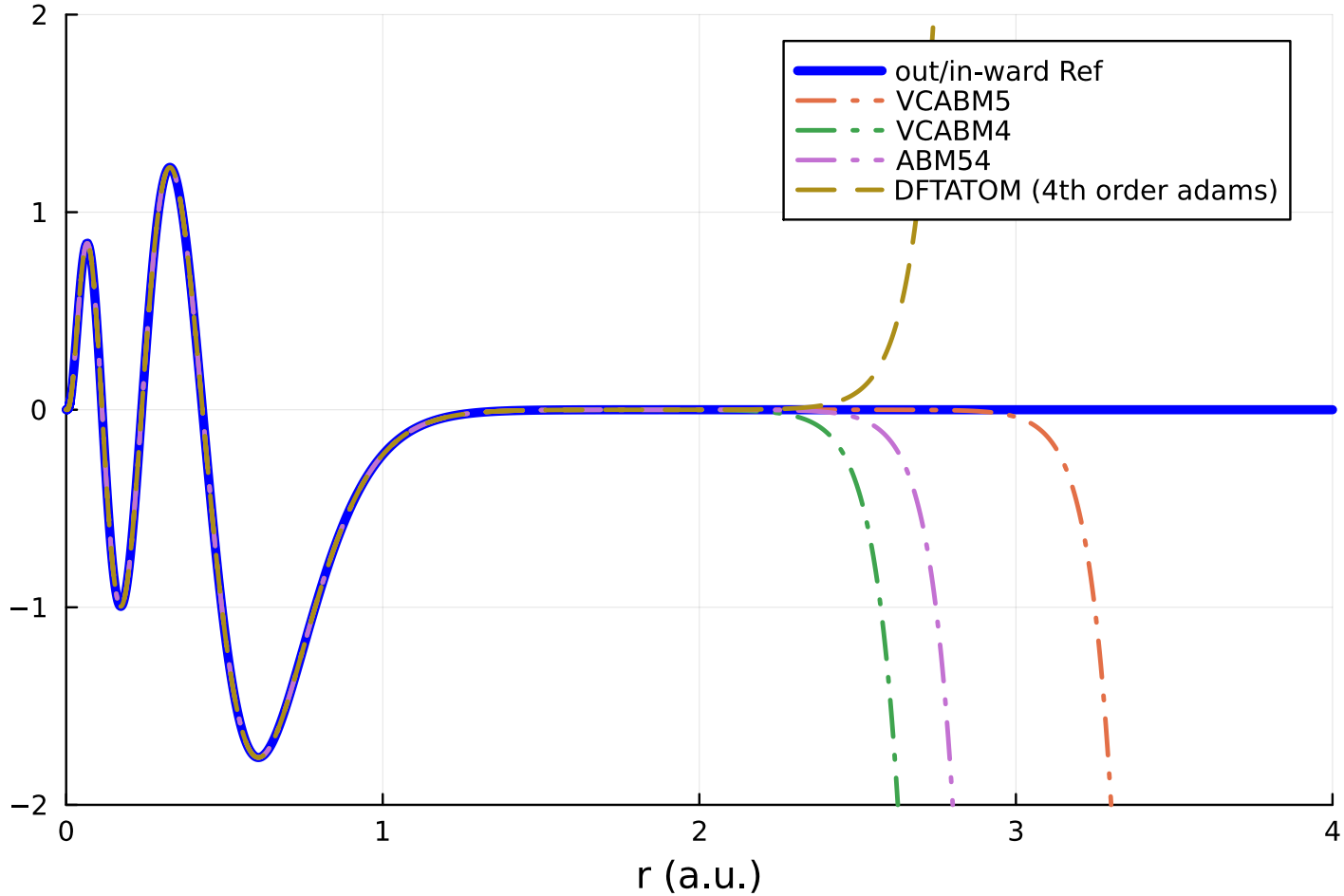
$$\lim_{r\to 0} \to 0, \qquad \lim_{r\to\infty} \to 0,$$

for different solutions moving close to the eigenvalue



Numerical stability and precision of solvers

ODE solver on orbital 6d of atom U (92).



Flavor of Pseudos (impose norm-conserving)

- BHS (1982) [1]: potential is constructed to get pseudo-waves matched at r_c.
- RRKJ (1990) [4]: pseudo-waves expanded by Bessel functions.
- TM (1991) [5]: pseudo-waves expanded by polynomials.

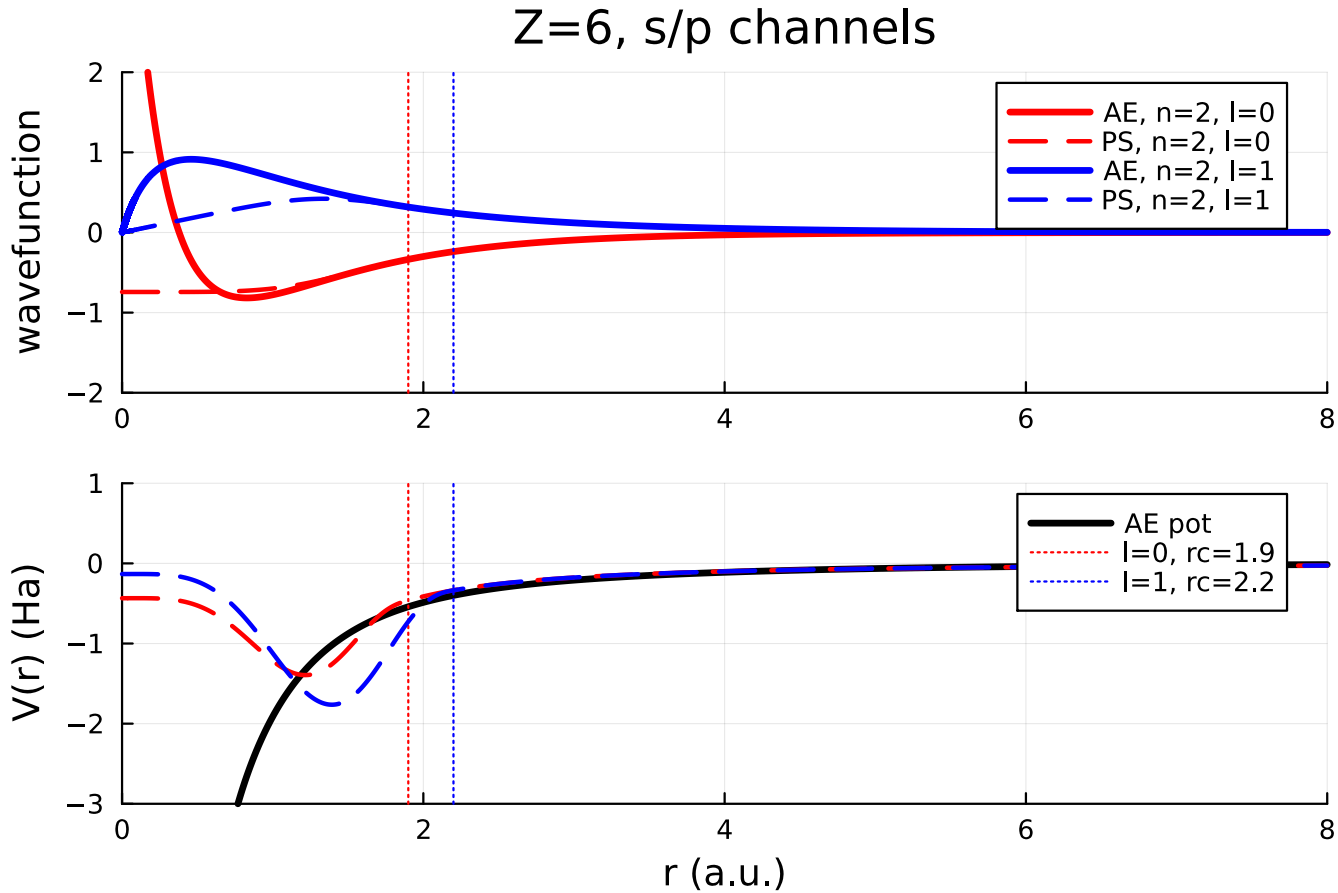
Troullier-Martins (TM) Pseudopotential [5]

$$R_l^P(r) = \begin{cases} R_l^{AE}(r), & \text{if } r > r_l \\ r^l e^{p(r)}, & \text{if } r < r_l \end{cases} \tag{10}$$

$$p(r) = c_0 + c_2 r^2 + c_4 r^4 + c_6 r^6 + c_8 r^8 + c_{10} r^{10} + c_{12} r^{12}, \tag{11}$$

The coefficients are determined by norm-conserving, continuity on 1st to 4th derivative at $r = r_l$ and $c_2^2 + c_4(2l + 5) = 0$ which lead by the needed of zero curvature of potential at the origin.
These multiple constrains forming a standard nonlinear equations to solve.

Troullier-Martins (TM) Pseudopotential



`PseudopotentialGenerator.jl/examples/logder.jl`

$\tan^{-1}$ Logarithmic derivative

The metric to measure the discrepancy between AE wavefunction and PS wavefunction in the atom level can be defined as[2],

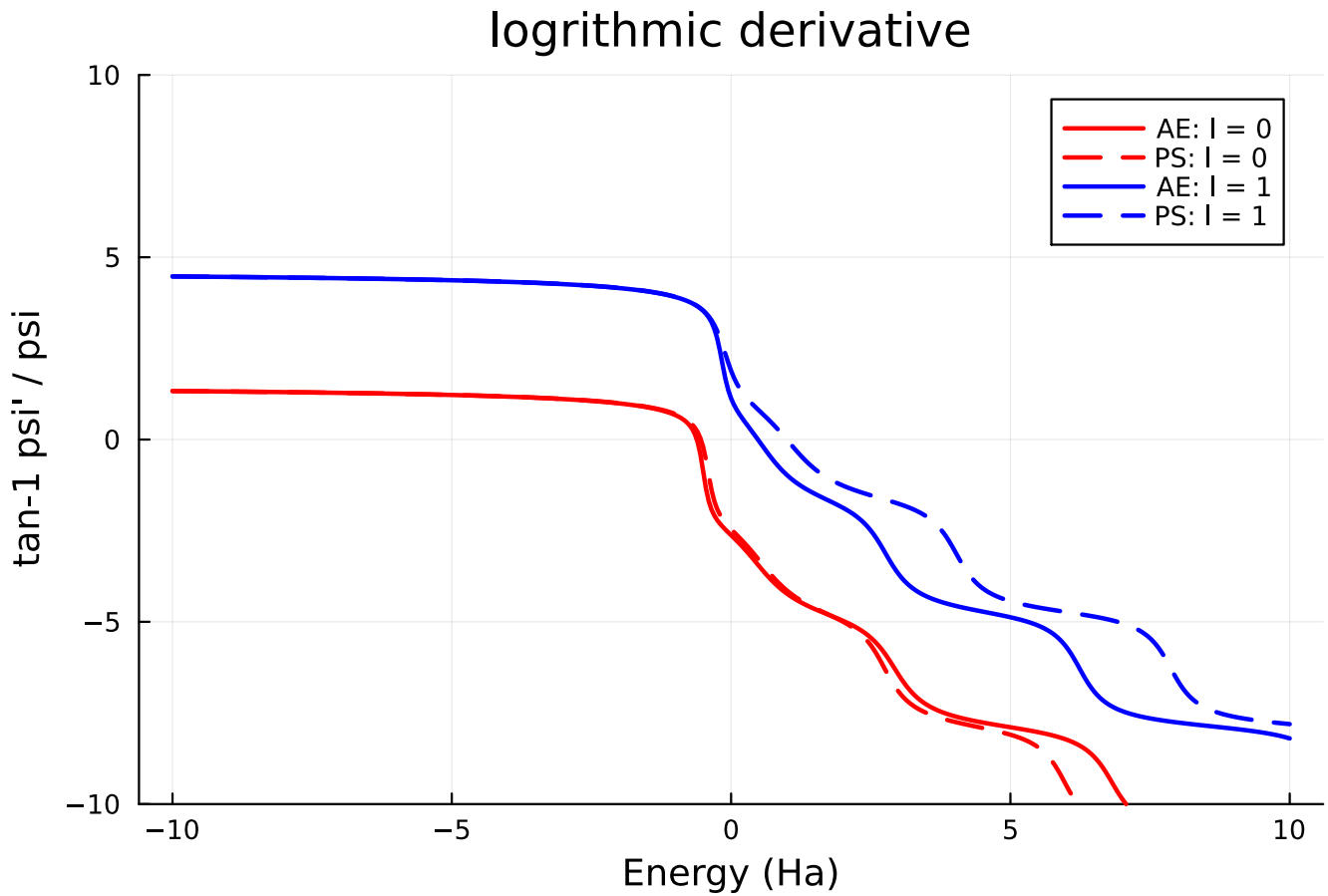
$$S_{a,l} = \sqrt{\frac{1}{N}\sum_{E=E_1}^{E_2}(A_{l,PS}(E) - A_{l,AE}(E))^2}, \tag{12}$$

where

$$A_l(E) = \tan^{-1}\frac{\Psi_l(E,r)'}{\Psi_l(E,r)}\bigg|_{r=r_c} \tag{13}$$

is the $\tan^{-1}$ Logarithmic derivative.

$\tan^{-1}$ Logarithmic derivative



`PseudopotentialGenerator.jl/examples/logder.jl`

Theory and equations

Implementation in Julia

Why julia? The PGEN.JL pkg and pkgs ready to use

1. https://github.com/unkcpz/PseudopotentialGenerator.jl⇒ (P)seudopotential (GEN)eration .(JL)
2. Libxc.jl (xc term)
3. NonlinearSolve.jl (SCF and Troullier-Martins)
4. DifferentialEquation.jl (Solve KS equation)
5. Actively maintained AutoDiff pkgs which boost ML requirements in PP optimization.

Functionals with `Libxc.jl`

```
1  function compute_vxc(rho::Vector{Float64}, xc::Symbol)
2      # compute the exchange-correlation potential
3      if xc == :lda
4          func_x = Functional(:lda_x)
5          func_c = Functional(:lda_c_vwn)
6      else
7          throw(ArgumentError("Unsupported exchange-
               correlation functional $xc"))
8      end
9
10     result_x = evaluate(func_x, rho = rho)
11     result_c = evaluate(func_c, rho = rho)
12
13     v_xc = result_x.vrho[:] + result_c.vrho[:]
14     e_xc = result_x.zk + result_c.zk
15     (; v_xc = v_xc, e_xc = e_xc)
16 end
```

SCF with NonlinearSolve.jl

```
function fixpoint_ks(vin, params)
    rho = v2rho!(vin, params, saved)
    vout = rho2v!(rho, params, saved)

    saved.rho = rho
    iter += 1

    delta_v = vin - vout
    delta_v
end

prob = NonlinearProblem(fixpoint_ks, v0, params)
solve(
    prob,
    alg = NLsolveJL(; method = :anderson, beta =
        mixing_beta);
    abstol = abstol,
    maxiters = maxiters_scf,
)
```

TM with NonlinearSolve.jl

PseudopotentialGenerator.jl/src/pseudolize.jl(TM)

$v_{\text{eff}} \to \psi$ with `DifferentialEquation.jl`

The problem is a 2nd-ODE [eq.3]. First convert it to two 1st-ODEs [eq.4, eq.5]. Then from radial mesh to uniform mesh and do discrete integration.
The two 1st-ODEs in uniform mesh is defined as `f!` (which is the difficult part but all just parts requires pen and paper.). It then passed it to solver to be solved.

```
1 prob = DiscreteProblem(f!, u0, (1, N))
2 sol = solve(prob, VCABM5(), dt = 1, adaptive = false,
      callback = cb)
```

`PseudopotentialGenerator.jl/src/ode_sch.jl(outward)`

Where I am and what are missing?

Missing (TODO list)

- To scalar/full-relativistic PP (1w)
- To RRKJ and BHS PP (1w)
- Logder analysis for Kleinman-Bylander form (1w)
- Extend to integrate to `DFTK.jl` (2w)
- Support ONCV PP (1m)
- Performance! Performance! (—)
- PAW PP (??m)
- meta-GGA (??m)

Ack

- Discussion and inspirations: Giovanni Pizzi, Matteo Giantomassi, Gian-Marco Rignanese, Austin Zadoks, Junfeng Qiao, Michael Herbst, Niklas Schmitz, Gian Parusa.
- DFTATOM[3] as tests, benchmarks and implementation reference on AE part.
- Initial inspiration also from PStudio by Davide Ceresoli.
- Shuyao and Dian since this is a night project in around 2 months.

References I

[1] Giovanni B Bachelet, Don R Hamann, and Michael Schlüter. "Pseudopotentials that work: From H to Pu". In: *Physical Review B* 26.8 (1982), p. 4199.

[2] Casey N Brock, Alan R Tackett, and D Greg Walker. "Metric based on the arctangents of the logderivatives for evaluating scattering properties of pseudopotentials". In: *Computer Physics Communications* 247 (2020), p. 106929.

[3] Ondřej Čertík, John E Pask, and Jiří Vackář. "dftatom: A robust and general Schrödinger and Dirac solver for atomic structure calculations". In: *Computer Physics Communications* 184.7 (2013), pp. 1777–1791.

[4] Andrew M Rappe et al. "Optimized pseudopotentials". In: *Physical Review B* 41.2 (1990), p. 1227.

References II

[5] Norman Troullier and José Luís Martins. "Efficient pseudopotentials for plane-wave calculations". In: *Physical review B* 43.3 (1991), p. 1993.