PyDFT and PyRTP: accurate and accessible DFT/RT-TDDFT calculations for all

Matthew Thompson, Matt Watkins

School of Mathematics and Physics
University of Lincoln, UK.

DFT is hard...

Beyond just the theory, quantum chemistry/solid-state physics programs are generally written in low-level languages, making the source codes (and thereby the numerical methods) used by these programs inaccessible to new and inexperienced users [1]. Similarly, testing new methods (or even minor tweaks) to existing QC programs requires an understanding of the dependencies for the entire program.

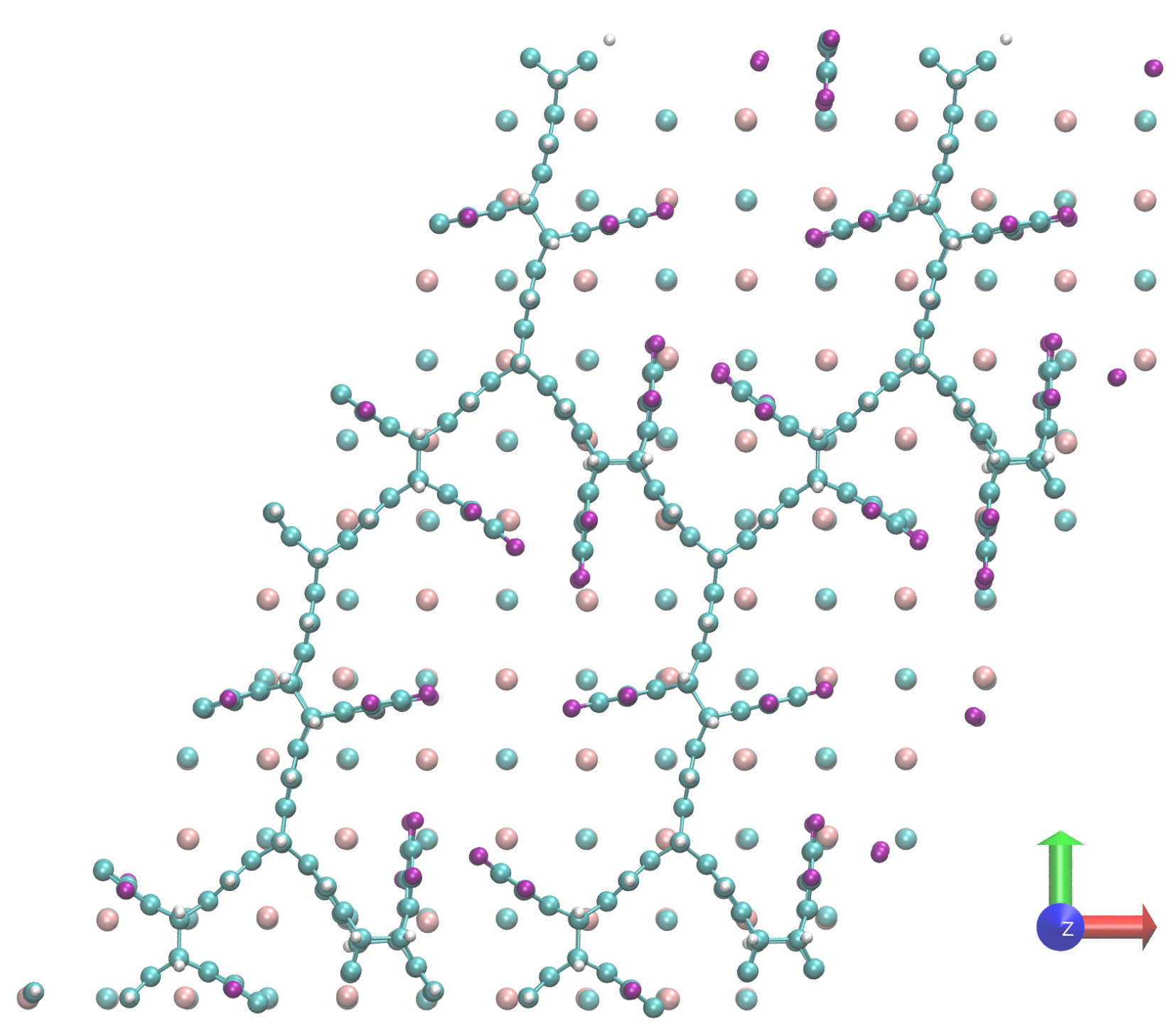


Figure 1. An example system commonly simulated with DFT (Fantrip polymer network deposited onto KBr bulk)

The primary objective of this project was to produce a program which could accurately perform DFT and RT-TDDFT calculations ***as simply as possible***, with the end goal being a simple-to-use tool for both users new to DFT, as well as more experienced users, who may wish to benchmark methods easily.

Considerations

Our criteria for success in this project were, in order of importance, as follows:

- Easy to read
- Accurate methods
- Reflective of methods in other programs
- *Relatively* computationally efficient
- Accessible to all

Based on these considerations, the following key design decisions were made:

- Written exclusively in Python - Free, clear syntax and can be further optimised using external packages.
- Mirrors the methods used by *CP2K* - CP2K is free, open-source and popular within the community [2]. Alongside that, CP2K's GPW method 'Quickstep' is robust and accurate.
- Uses only packages available on a range of OSs - Many QC programs require a Unix-based operating system, which locks out many less experienced users.
- Optimised with multithreading (where possible) - The *Numba* package is used to multithread processes, allowing the programs to be run relatively quickly.
- Available in an .ipynb format - IPython notebooks allow for a blended approach between theory and code, helping students.

@Matt-A-Thompson

DFT Method

PyDFT uses the Kohn-Sham DFT method, and thus, the following equation [3]:

$$E_{KS}[\rho(\boldsymbol{r})] = T_S[\rho(\boldsymbol{r})] + E_H[\rho(\boldsymbol{r})] + E_{xc}[\rho(\boldsymbol{r})] + \int d\boldsymbol{r}\, v_{ext}(\boldsymbol{r})\rho(\boldsymbol{r})$$

The Gaussian and Plane Waves (GPW) method is used [4], meaning ϕ is approximated using a Contracted Gaussian Function:

$$\phi^{CGF}(\boldsymbol{r}) = \sum_{\lambda} C_\lambda \phi_\lambda^{GTO}(\boldsymbol{r})$$

Upon solving for E_{KS} (expressed as a matrix KS), an SCF loop of the following form is completed to converge upon the ground state density matrix $\boldsymbol{P}$.

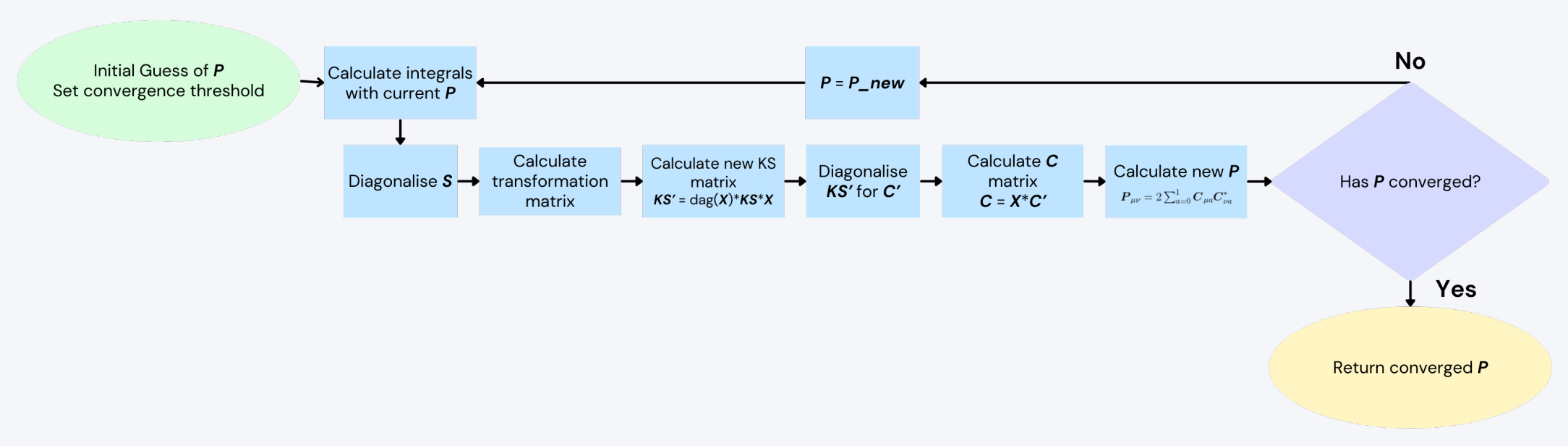

Initial Guess of P
Set convergence threshold
Calculate integrals with current P
P = P_new
No
Diagonalise S
Calculate transformation matrix
Calculate new KS matrix KS′ = dag(X)*KS*X
Diagonalise KS′ for C′
Calculate C matrix C = X*C′
Calculate new P
Has P converged?
Yes
Return converged P

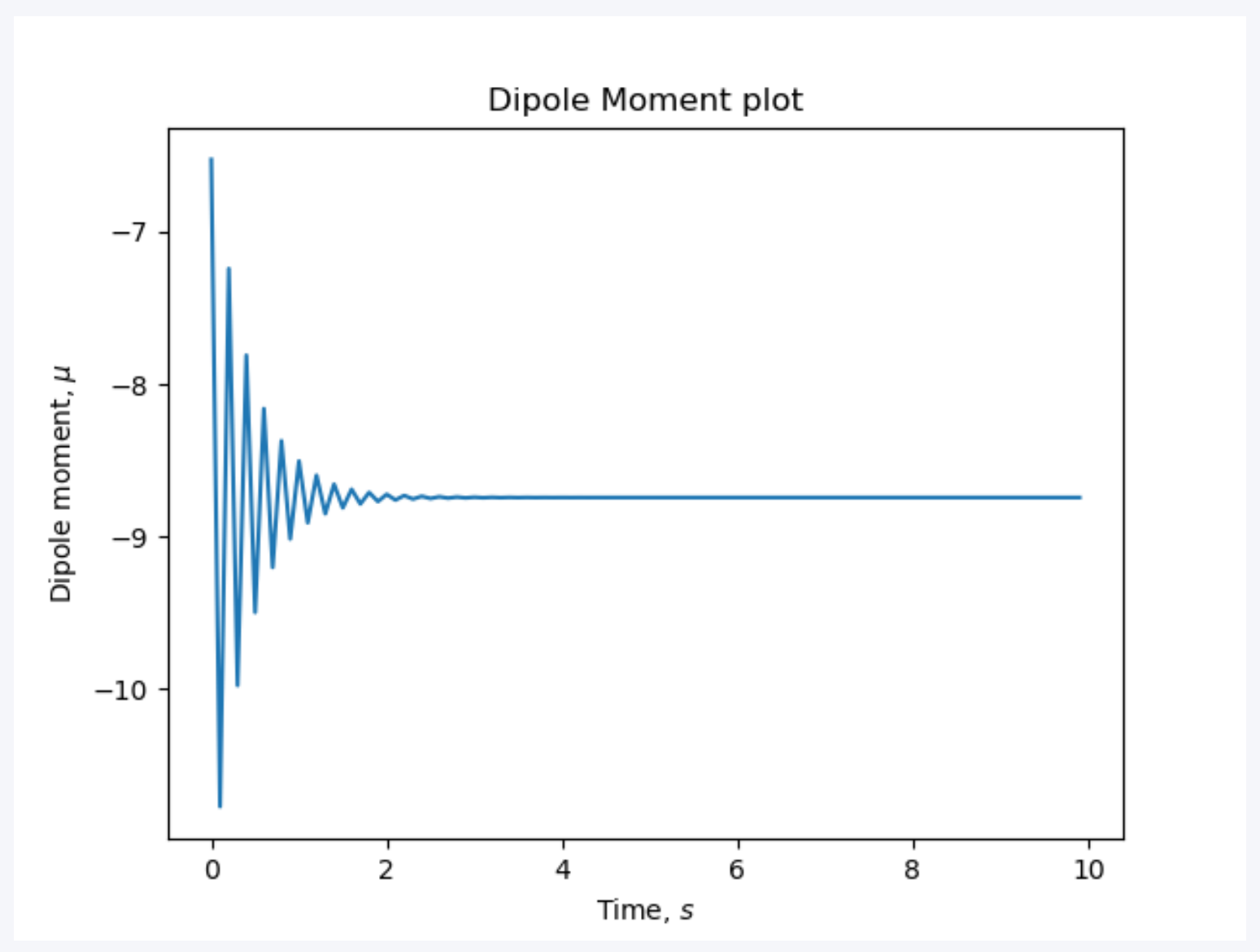
Figure 2. A flowchart of the SCF process used in PyDFT and PyRTP

RT-TDDFT Method

From Runge-Gross theorem, a unitary operator (usually referred to as the propagator) exactly maps the electron density from t_0 to t [5]. This operator is exact and thus, the following approximations to it are available in PyRTP:

- Crank-Nicholson
- Exponential Midpoint
- Enforced Time-Reversal Symmetry (ETRS)
- Approximated ETRS (AETRS)
- Corrected-Approximated ETRS (CAETRS)
- Commutator-Free 4th Order Magnus

Using this, time-dependent phenomena (such as dipole moment, von Neumann entropy, etc.) can be determined.



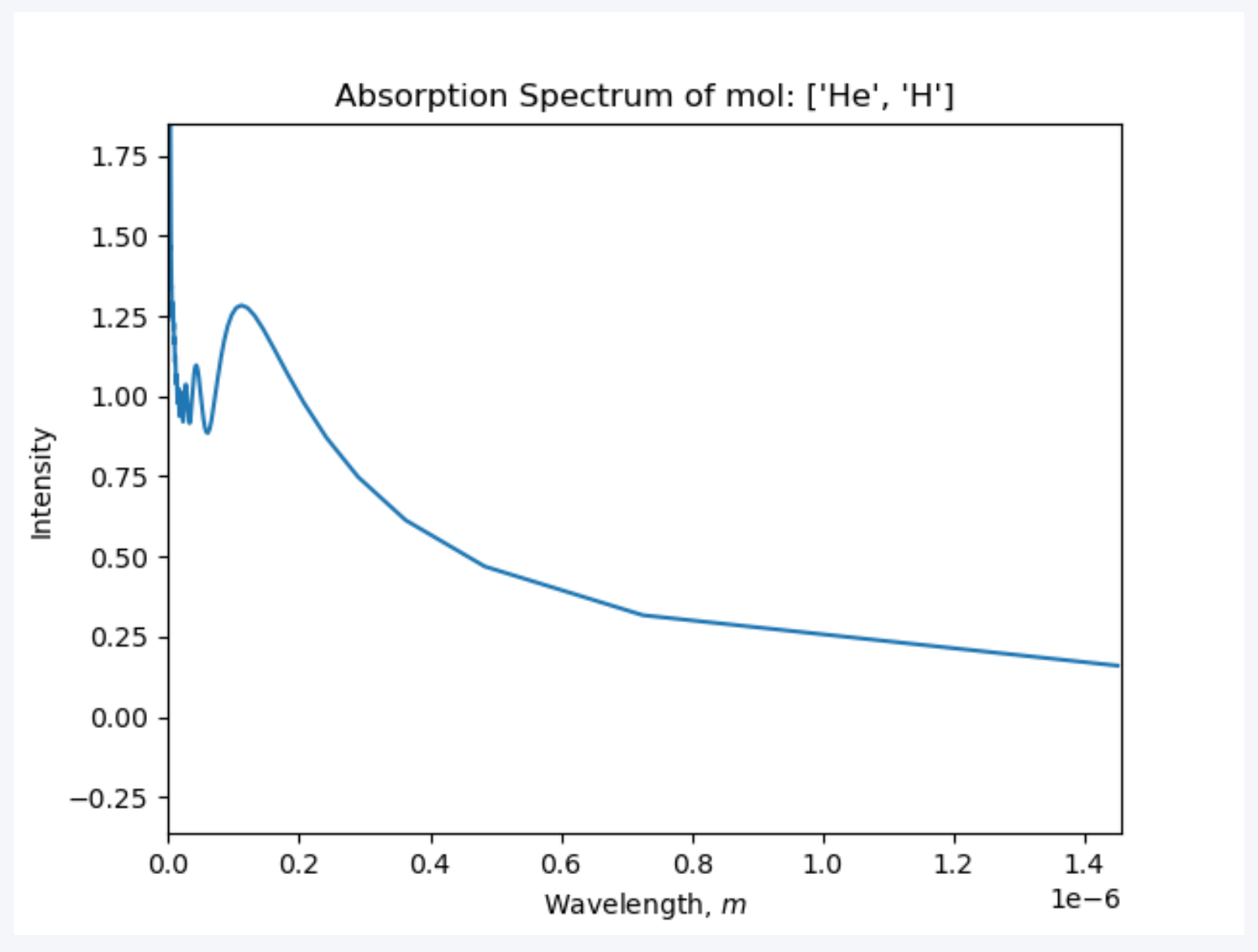
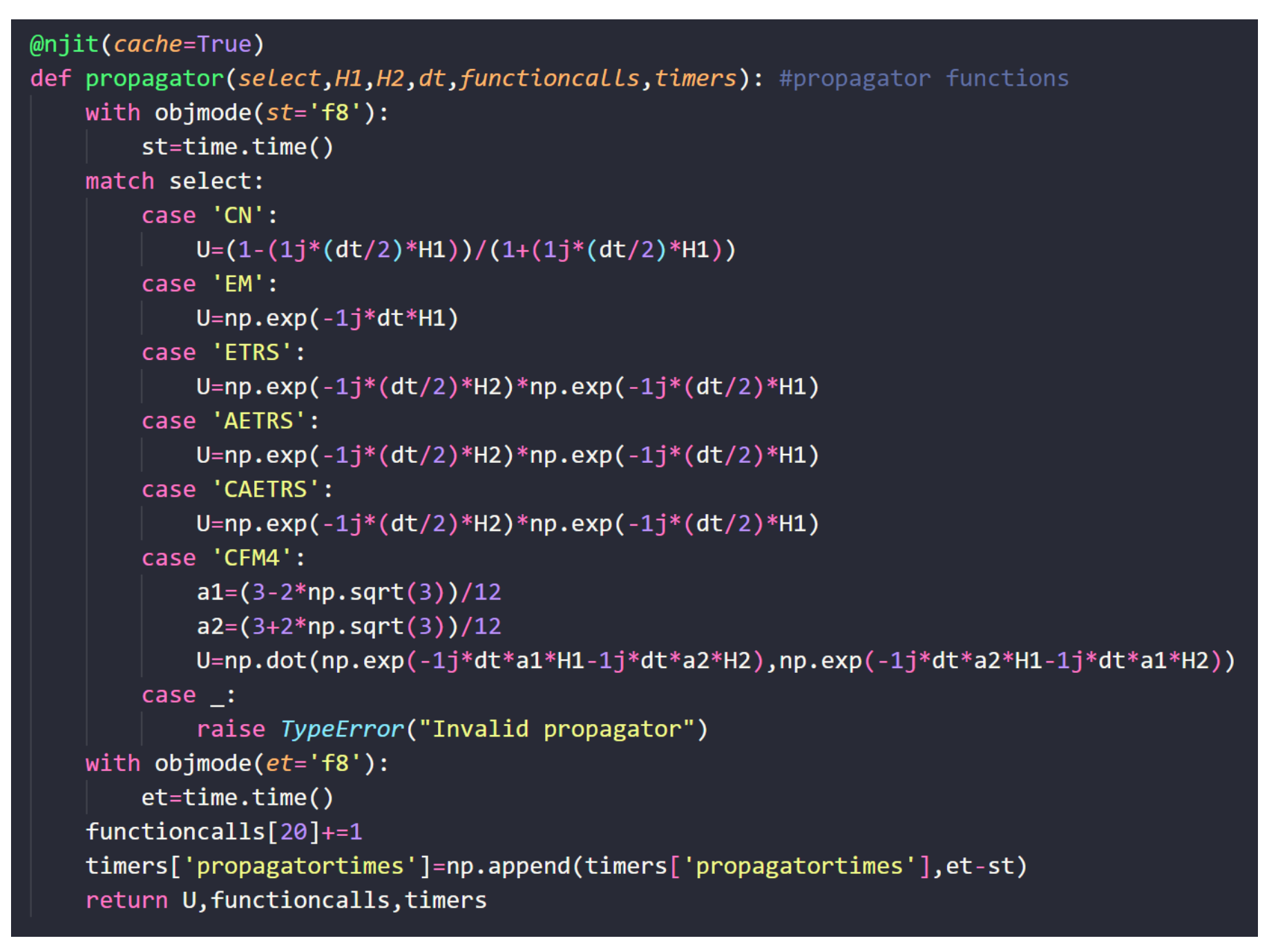


Figure 3. Examples of time-dependent phenomena observed in PyRTP

Applications

The source code for this project is written as clearly as possible, with large portions of the code having comments to explain specific steps. Alongside this, having the program available as a IPython notebook allows for theory to be discussed alongside working code.



```python
@njit(cache=True)
def propagator(select,H1,H2,dt,functioncalls,timers): #propagator functions
    with objmode(st='f8'):
        st=time.time()
    match select:
        case 'CN':
            U=(1-(1j*(dt/2)*H1))/(1+(1j*(dt/2)*H1))
        case 'EM':
            U=np.exp(-1j*dt*H1)
        case 'ETRS':
            U=np.exp(-1j*(dt/2)*H2)*np.exp(-1j*(dt/2)*H1)
        case 'AETRS':
            U=np.exp(-1j*(dt/2)*H2)*np.exp(-1j*(dt/2)*H1)
        case 'CAETRS':
            U=np.exp(-1j*(dt/2)*H2)*np.exp(-1j*(dt/2)*H1)
        case 'CFM4':
            a1=(3-2*np.sqrt(3))/12
            a2=(3+2*np.sqrt(3))/12
            U=np.dot(np.exp(-1j*dt*a1*H1-1j*dt*a2*H2),np.exp(-1j*dt*a2*H1-1j*dt*a1*H2))
        case _:
            raise TypeError("Invalid propagator")
    with objmode(et='f8'):
        et=time.time()
    functioncalls[20]+=1
    timers['propagatortimes']=np.append(timers['propagatortimes'],et-st)
    return U,functioncalls,timers
```

Figure 4. An example of the programming in PyRTP

Both programs can also accept basis set and pseudopotential files designed for CP2K, allowing a user to adjust their inputs according to the needs of the system they are simulating.

Outlook

This project is still ongoing, and we seek to complete the following tasks in the future:

- **Continue benchmarking and optimising to ensure accuracy in a variety of systems** - some very large systems have not been tested thoroughly with these methods. While this is not the intended use case, testing with these larger systems will ensure the processes are robust.
- **Collate the programs into a Python package** - by producing packages and distributing it through existing package managers (i.e. *conda*, *pip*, etc.), it will be easier for students to access these methods.
- **Produce thorough documentation and teaching materials** - in line with the aims of this project, good documentation will be crucial in enabling students to develop their understanding of DFT/RT-TDDFT. As well as this, producing suitable teaching resources will enable institutions to easily deliver content.

References

[1] M. Ward and S. Rd, "A definition of abstraction," *Journal of Software Maintenance: Research and Practice*, vol. 7, 11 1995.

[2] T. D. K. et al., "CP2K: An electronic structure and molecular dynamics software package - Quickstep: Efficient and accurate electronic structure calculations," *The Journal of Chemical Physics*, vol. 152, no. 19, p. 194103, May 2020. [Online]. Available: https://doi.org/10.1063/5.0007045

[3] W. Kohn and L. J. Sham, "Self-consistent equations including exchange and correlation effects," *Phys. Rev.*, vol. 140, pp. A1133–A1138, Nov 1965. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRev.140.A1133

[4] G. Lippert, J. Hutter, and M. Parrinello, "The gaussian and augmented-plane-wave density functional method for ab initio molecular dynamics simulations," *Theoretical Chemistry Accounts*, vol. 103, no. 2, pp. 124–140, 1999. [Online]. Available: https://doi.org/10.1007/s002140050523

[5] E. Runge and E. K. U. Gross, "Density-functional theory for time-dependent systems," *Physical Review Letters*, vol. 52, no. 12, pp. 997–1000, March 1984.

MatThompson@lincoln.ac.uk