Open access • Posted Content • DOI:10.33774/CHEMRXIV-2021-HR1R0-V2

Free and Open Source Software for Computational Chemistry Education

— Source link

Susi Lehtola, Antti J. Karttunen

Institutions: Virginia Tech, Aalto University

Published on: 12 Aug 2021 - ChemRxiv

Topics: Chemistry education, Software and Bring your own device

Free and Open Source Software for Computational Chemistry Education

Susi Lehtola*,† and Antti J. Karttunen‡

†*Molecular Sciences Software Institute, Blacksburg, Virginia 24061, United States*

‡*Department of Chemistry and Materials Science, Aalto University, Espoo, Finland*

E-mail: susi.lehtola@alumni.helsinki.fi

Abstract

Long in the making, computational chemistry for the masses [J. Chem. Educ. 1996, 73, 104] is finally here. We point out the existence of a variety of free and open source software (FOSS) packages for computational chemistry that offer a wide range of functionality all the way from approximate semiempirical calculations with tight-binding density functional theory to sophisticated *ab initio* wave function methods such as coupled-cluster theory, both for molecular and for solid-state systems. By their very definition, FOSS packages allow usage for whatever purpose by anyone, meaning they can also be used in industrial applications without limitation. Also, FOSS software has no limitations to redistribution in source or binary form, allowing their easy distribution and installation by third parties. Many FOSS scientific software packages are available as part of popular Linux distributions, and other package managers such as pip and conda.

Combined with the remarkable increase in the power of personal devices—which rival that of the fastest supercomputers in the world of the 1990s—a decentralized model for teaching computational chemistry is now possible, enabling students to perform reasonable modeling on their own computing devices, in the bring your own device

(BYOD) scheme. In addition to the programs' use for various applications, open access to the programs' source code also enables comprehensive teaching strategies, as actual algorithms' implementations can be used in teaching.

We discuss the availability and use of various FOSS quantum chemistry packages and demonstrate what kinds of calculations are feasible with these programs, assuming only extremely modest computational resources. Our examples confirm that FOSS software enables decentralized approaches to computational chemistry education within the BYOD scheme, affording a democratization of the science of computational chemistry as well.

Contents

1 Introduction

In addition to its widespread use in industry and academia, computational chemistry is utilized in chemical education to provide atomic-level understanding of fundamental chemical concepts and phenomena.[1,2] For example, in undergraduate general and organic chemistry curricula, students get hands-on experience on concepts such as three-dimensional molecular structure, structural isomerism, conformers, and stereochemistry by means of computational exercises or computer laboratory sessions.[3–5] Although some of the aforementioned aspects can in principle be studied even with simpler methodologies such as classical force fields, quantum chemical calculations with state-of-the-art software packages allow students to get hands-on experience on more advanced topics such as molecular orbitals, chemical bonding, energetics,[6] thermodynamics,[7,8] reaction mechanisms,[9] and various spectroscopies.[10–14]

The ability to interpret and understand chemical phenonema with the help of quantum chemical calculations is a valuable skill in every chemists' professional life. Quantum chemical research methods have been used extensively in the chemical industry for decades already.[15–18] Nowadays, a significant portion of even the experimental studies reported in the chemical literature are tightly integrated with quantum chemical investigations. Moreover, as quantum chemistry is the critical bridging component between experimental work and machine learning methods, the ability to run quantum chemical calculations can be expected to become even more increasingly relevant and necessary to work-life in the near future.

Computational chemistry for the masses has been long thought to be coming,[19] but it does not appear to have arrived yet. In their recent overview,[20] Grushow and Reeves have summarized some select landmarks in computational chemistry education, noting at the same time how computational chemistry still has a somewhat limited presence in undergraduate curriculums, which can be attributed at least in part to its history.

In the 1990s, commercial software companies started selling graphical user interfaces to their quantum chemistry packages, some of which were particularly geared towards educational use. Such software was and still is typically used in a computer classroom setting, where a limited number of relatively powerful desktop computers are available for the students during the teaching sessions. The benefit of a computer classroom setting is that all software can be pre-installed for the students and the standardized software environment makes the possibilities (and limitations) of the software setup clear for the teachers in charge of the educational content. However, the computer classroom approach has limited scalability, as the number of students is limited by the number of workstations; this often makes the approach impractical for large-scale undergraduate teaching. Furthermore, while the computer classroom setting may be useful for teaching during contact sessions, the students' possibilities for running calculations outside the contact sessions are limited by the requirement of physical access to the computer classroom—which has proved to be challenging especially during the ongoing global coronavirus disease pandemic which has required social distancing. Lastly, the classroom setting typically limits the teacher and students to the pre-installed software, while costs for the required software licenses can be unfeasibly high for educational institutions with limited budgets.

In the early 2000s, the WebMO package introduced a web-based approach to computational chemistry education, where the quantum chemistry software packages only need to be installed on a central server, and the teachers and students can then access the software through a web browser interface.[21,22] A number of quantum chemistry software packages have been integrated with WebMO, and the molecular editor and analysis tools integrated in WebMO make it a rather low-barrier interface to quantum chemistry. As the users thus only need a web browser to access the computing software, WebMO was the first tool to enable a bring your own device (BYOD) paradigm in computational chemistry, in which the students (and faculty!) can use their personal devices to take part in the teaching.

However, WebMO still requires someone to set up and administer the WebMO server, even though the need to purchase actual server hardware has been removed by the possibility of installing the service on cloud platforms such as the Amazon Web Services or the Google Cloud. Recently, the cloud-based Chem Compute platform has began to offer web access to computational chemistry software and computing resources for undergraduate teaching and research without any cost to the teachers,[23] enabling such access for institutions that do not have the personnel or financial resources to set up their own physical or cloud servers; however, Chem Compute relies on computational resources volunteered by third parties whose availability in the future is not guaranteed.

As discussed above, great advances like WebMO and Chem Compute have been made in the direction of the BYOD paradigm to which many universities have already shifted in order to cut down on the costs associated with the now-deprecated computer classroom model. In this work, we will show that free and open source software (FOSS) can be used in the context of the BYOD paradigm to achieve computational chemistry for the masses, all the while democratizing science by tearing down established power structures and barriers for research and education.

The layout of this work is as follows. In section 2, we will begin by defining what we mean by FOSS (section 2.1). We discuss why FOSS has not been the norm in science (section 2.2), what FOSS enables for the teaching of computational chemistry (section 2.3), and why it would be a good time now to switch over to FOSS in teaching (section 2.4). We present a brief overview of available FOSS packages in section 3 and include several practical demonstrations of using state-of-the-art FOSS programs for computational chemistry education in section 4. The article concludes in a brief summary and discussion in section 5.

2 Free and open source software

2.1 Definitions

As some of the readers may not be familiar with the concept of FOSS, some definitions are necessary before the present discussion can take place. For the purposes of this article, we will adopt three key criteria for FOSS:

1. The ability *of anyone* to freely use the software for *any* purpose.
2. The ability to freely study the operation of the software, and modify it at will.
3. The ability to freely redistribute copies of the software—as well as modified versions thereof—to others.

Consequently, any software that does not satisfy these criteria for FOSS is referred to as proprietary or closed source software.

What is the significance of these criteria? The first criterion means simply that there can be no limitations on potential uses of the software: for instance, in addition to use in academic research and education, commercial use must also be permitted by the license. Moreover, the first criterion bars license terms that prohibit use of the software for purposes deemed questionable by the licensors, such as use in nuclear power plants or in research on genetic engineering. FOSS can be used *by anyone for anything.*

The second criterion means that the source code of the software must not only be available, but also that customizations to the source code must be allowed. This is of major importance for *e.g.* developing new features or computational models. Being able to use software written by other authors to accomplish certain tasks eliminates the need to "reinvent the wheel" and thereby results in faster scientific development.[24] This phenomenon has traditionally been the main enticement of contributing to closed-source or "open teamware"[25] packages, as access to their source code partly eliminates the need to start from scratch, as algorithms implemented by the other contributors can be leveraged to develop new computational models.

However, the control of access to the source code in closed-source programs lead to perpetuating power structures and may inhibit academic collaborations between authors of different program packages,[26] instead of the Popperian ideal of science: the selfless pursuit of truth,[27] and a fair and unbiased competition of ideas and methods in the context of computational chemistry. As quipped already by Max Planck, "A new scientific truth does not triumph by convincing its opponents and making them see the light, but rather because its opponents eventually die, and a new generation grows up that is familiar with it"; this apt observation has recently been verified by experiment.[28]

The third criterion means that *anyone* who has a copy of the software can redistribute it to others. One does not need to ask case-by-case permission from the authors of the software in order to share it with *e.g.* one's collaborators or the reviewers of a scientific paper. It also means that anyone who has added new features to the program can freely distribute their version. This eliminates the problematic role of the gatekeepers in the "open teamware" model, as alternative versions of the software commonly known as *forks* can be distributed. It also eliminates the possibilities of the infamous practice[29] of preventing one's competitors from using one's software, which may have the result of hiding deficiencies and bugs in one's software. Case in point: the "war on supercooled water"[30] exemplifies the problems of having prominent figures as exclusive gatekeepers. The "war" was only resolved once Princeton scientists gained access to their Berkeley competitors' source code and found a coarse error in the Berkeley source code.[31] Such problems are much less likely to exist if FOSS is used, as such programs are freely redistributable and can be thoroughly inspected by anyone.

In our opinion, the three criteria laid out above condense the essence of both the generally accepted 10-item definition for open source software by the Open Source Initiative[32] as well as the four essential freedoms of "free software" or "libre software" defined by the Free Software Foundation.[33] Note that there is a wide variety of FOSS licenses that fit these criteria and that can be adopted by software projects, and that new software projects should choose their license with care.[34] It is always easier to switch to a more permissible license later on than to move to a more restrictive license: any versions released under a FOSS license will continue being FOSS in the future, as well, even if newer versions switch to using a proprietary license, for example.

2.2 Why is free/open-source software not the default?

The ideology of FOSS is in line with the demands of science,[35] as much like the Schrödinger or Dirac equation, computational models should ideally always be publicly available. Moreover, as the initial development and ongoing use of most scientific software has been and is funded by public research funding, the results of this work—the developed program source code—should be available to everyone.

It is worthwhile to comment on the reasons for the longstanding status quo. As discussed by Hinsen[36], before the advent of electronic computers, algorithms were developed with pen and paper, and the traditional paper journal article format is ideally suited to fully describe such algorithms. However, when implemented on a computer, algorithms often become too complicated to thoroughly describe in a journal article, and significant portions of the implementation are always left out. As this tacit information on what happens "under the hood" of various computational chemistry packages is typically passed only within the academic groups contributing to that code, lack of access to the source code creates another barrier of entry for third parties, and again ends up perpetuating established power structures.

However, nowadays there are well-established ways for distributing scientific software. Version control systems such as Git[37] facilitate robust development of software, which can be hosted for free on sites such as GitHub[38] and GitLab[39]. GitHub and GitLab also enable a community approach to code development through the use of public code review, which is leveraged by many program packages to improve code quality and to decrease the learning curve for potential new contributors to the package. Stable releases of software can be made available on Open Science data repositories such as Zenodo[40] with version-specific Digital Object Identifiers (DOIs). Also precompiled versions can nowadays be easily distributed, as we will discuss in section 3.

A commonly referred impediment of FOSS in science is that funding its maintenance and/or user support is challenging.[25,41,42] However, there are several companies whose whole business model is founded on the use and development of FOSS. For instance, Red Hat broke $1 billion in annual revenue in 2012, and its revenue has increased ever since, surpassing $3 billion in 2018.[43] There is clearly money to be made in selling support for FOSS. Moreover, at variance to proprietary software, maintenance and support for FOSS can be acquired from third parties if the original author(s) are either unavailable or unwilling to support for their code; Red Hat does this as well.

As evidenced by forums such as the Computational Chemistry List[44] and the present authors' professional experience, online peer-to-peer user support—whose motivations have been studied e.g. by Constant et al.[45]—is invaluable even in the case of proprietary programs. In the case of FOSS, this peer-to-peer support has an enhanced role, and is one of the keys behind the success of FOSS.[46] Because anyone can modify the software and distribute modified copies thereof, anyone can fix the bugs they run into, and gain fame even for small contributions. Also other aspects of the economic principles of FOSS have been studied extensively:[47–60] FOSS is a public good.[48,49,61] Participation in the development and support of FOSS has been found to be more motivating than that of proprietary software;[62,63] participation in FOSS projects is motivating and carries economic benefits.[64,65] FOSS promotes peer review, free exchange of ideas, and maintainability,[66] and competition of FOSS packages promotes innovation.[67]

The Linux operating system is a prime example of FOSS. Originating from the University of Helsinki, it is nowadays ubiquitous. It is used in billions of mobile phones, laptops, workstations, as well as servers and compute clusters all around the world. All supercomputers on the TOP500 list[68] and the majority of the world's internet servers have run on Linux for a long time; Android smartphones likewise run on Linux. Proprietary operating systems have been irrelevant in computational science for several decades. Chemists had good reasons to switch to Linux already ages ago;[69] the present authors have used Linux as their main computational research platform for over 20 years.

A valuable feature of Linux distributions is that they are usually cross-platform: in addition to the usual x86 and x86-64 platforms (consisting of processors by e.g. the Intel Corporation and Advanced Micro Devices Inc., AMD), Fedora packages are also available on s390x processors used on IBM mainframe computers and ARM processors such as the ones used in Raspberry Pi and new Mac computers, for instance. This versatility allows the use of heterogeneous hardware, ensuring seamless compatibility even if students come from a variety of backgrounds.

Several Linux distributions, such as Ubuntu, Debian, and Fedora Linux have also solved the problem of efficient distribution of software many decades ago. Our criteria for FOSS in section 2.1 allow such scientific software to be packaged as part of Linux distributions, and indeed several powerful program packages are already available as distribution packages thanks to the grand entrance of FOSS software in quantum chemistry in recent years. Some FOSS quantum chemistry packages like Erkale,[70] Psi4[71] and its predecessor Psi3[72], and PySCF[73] have been developed in a fully free/open-source development model since their beginning, while other packages that originated within a closed-source licensing model have also become open-sourced recently, such as OpenMolcas,[74] Dalton,[75] and NWChem.[76]

An example of a successful scientific FOSS can be found in the Libxc library of density functional approximations.[77] The modular library currently implements over 600 density functional approximations such as PBE,[78] B3LYP,[79] and SCAN,[80] and is used by over 30 electronic structure programs ranging from programs using Gaussian basis sets (Erkale, Psi4, PySCF, etc) to plane-wave codes (Quantum Espresso[81]), finite element programs (HelFEM[82–85], DFT-FE[86]), and multiresolution adaptive grids (MADNESS[87]). Libxc recently switched to a more permissible FOSS license that allows an easier inclusion into closed-source programs to allow Libxc to be used more widely in the community. Libxc is now used in several proprietary/commercial software packages, *e.g.* the Slater-type orbital ADF package[88] and the Gaussian-type orbital GAMESS-US,[89] Molpro,[90] and ORCA programs;[91] several other packages are also contemplating to migrate to Libxc. A new density functional approximation only needs to be implemented in Libxc for it to become usable in any of these programs, underlining the efficiency of the modular FOSS model. Moreover, access to the same implementation of a density functional approximation enables *e.g.* the study of reproducibility across various numerical approaches,[92] which is important to be able to compare results obtained with different methods or software packages. Indeed, economic gains in terms of software development productivity and product quality can be achieved by reuse of mature FOSS components that are of the highest quality.[93]

2.3 What does free and open source software offer for teaching?

In addition to its benefits for general use cases,[94] FOSS has three major advantages for teaching: the availability of the source code, the availability of precompiled binaries, as well as the general applicability of the software beyond academia. Starting out with the first advantage, software that satisfies the criteria for FOSS discussed in section 2.1 can be redistributed, and included in Linux distributions, for example. This greatly facilitates the installation of these programs, as prepackaged software can be installed in a matter of minutes on a wide range of hardware, ranging from students' laptops to compute servers. (We wish to note here that although installing scientific software by hand affords customized tunings that may result in faster operation, that is, decreased runtimes of quantum chemistry packages, in many cases the gains realizable in computational chemistry education or small-scale computing are relatively modest and pale in comparison with the ease of effort afforded by the centralized packaging system. Several proprietary programs have likewise adopted a binary-only distribution model with the same limitations.)

Second, as the source code is available, it can be used in teaching. For instance, a course on electronic structure calculations can exemplify the basic algorithms by showing how they are implemented in an openly available program. Some codes go even further: for instance, PySCF[73] makes it easy to override and customize all algorithms, as the program is mostly written in Python. This not only facilitates research in and development of new electronic structure methods, but also means that teaching no longer has to be limited to pen and paper exercises but instead can also include real-life demonstrations. For instance, an advanced course on electronic structure theory could involve asking students to write their own, customized solver for self-consistent field theory.[95] Also Psi4Numpy[96] should be mentioned in this context as a project that aims to supply simple, easily modifiable Python algorithms for educational purposes.

Third, because students (like anyone else) can access the full power of various computational chemistry programs instead of simplified frontends such as WebMO, they also have the the possibility to develop more general technical skills such as programming and interfacing programs with each other. As was summarized in the first criterion in section 2.1, FOSS software can also be freely used without limitations in industry to develop new thermoelectric energy conversion materials[97] or semiconductor devices,[98] for instance.

Simplified frontends are often invaluable for initializing, visualizing and analyzing calculations. As will be discussed in section 3.5, several FOSS packages with graphical user interfaces are also available for this purpose; some even come with integration with FOSS electronic structure programs that allow running calculations within a graphical interface.

2.4 Why would it be timely to switch to free/open-source software?

We have argued above that FOSS has important ramifications for the reproducibility of science and also has several advantages for teaching. Although it is possible to switch from proprietary programs to FOSS within the traditional setup based on computer classrooms and/or central compute servers, there is yet another important aspect to consider: the BYOD approach discussed in section 1. In this section, we wish to examine FOSS from the point of view of the ongoing paradigm shift to the BYOD scheme.

As the price of laptop computers has dropped, many students now bring their own devices to the classroom. This paradigm shift has also affected university policies. As the demand for computer classrooms has decreased significantly as most students anyhow prefer using their own devices, universities may find it cheaper to just offer a laptop to all students. For instance, the Faculty of Science of the University of Helsinki, Finland pivoted to such an approach several years ago. As a result, the university has been able to cut down on computer classrooms that are expensive to maintain even while several students opt to using their private laptops instead of checking out the laptop offered by the university.

Although as was already discussed in section 1, a centralized compute server approach is compatible with the BYOD paradigm, the effortless availability of FOSS programs can be used to finally bring computational chemistry to the masses and thereby truly democratize science. As FOSS software packages can be made instantly available to everyone, the FOSS approach is ideally suited for personal devices in the BYOD approach. Such a distributed approach is ideally suited also for massive open online courses (MOOCs), as enrollment does not have to be limited based on the available centralized computer resources; instead, the students can run all of the necessary calculations on their own hardware.

Naturally, certain tradeoffs are implied in a course limited to heterogeneous BYOD approaches, as one cannot assume personal devices to have the same computational power as purpose-built, dedicated compute servers. However, we argue that this is not much of an impediment due to the immense developments in the speed of processors and improved algorithms achieved during the past several decades. A concrete example of this is the TOP500 list of supercomputers, which contains almost 30 years worth of data on the most powerful supercomputers in the world.[99,100] The estimated performance of the best and worst supercomputer on the list on a year-by-year basis is shown in figure 1 in units of 10^9 floating-point operations per second (GFlops). Figure 1 also shows analogous benchmark data for commodity hardware: a cheap tablet computer with an Intel Celeron N4000 processor and a high-end business laptop with an Intel i7-10610U processor of one of the present authors (SL). A Raspberry Pi 4 minicomputer was also assessed, and found to perform similarly to the Celeron N4000 processor.

As figure 1 illustrates, personal devices have performance in the tens to hundreds of gigaflops, which is comparable to the performance of fastest supercomputers of the mid 1990s, or to the slowest supercomputer on the TOP500 list in the mid 2000s. This amazing development in computational power means that the content of classic books on quantum chemistry such as Szabo–Ostlund[101] could be reproduced nowadays on commodity hardware; however, there's no reason to, since better computational methods and basis sets are available nowadays in many FOSS packages. Many calculations could probably be even carried out on an up-to-date smartphone!

The data in figure 1 suggest that a variety of calculations are possible within a reasonable time with personal devices. Combined with FOSS program packages that can be installed in a trivial fashion with a package manager, computational chemistry can finally be made available to the masses. Students are able to run (and modify!) FOSS packages on their own devices; the skills they gain doing so are directly transferable to both research and industry, as the same packages can also be used for heavy-duty calculations on supercomputers which is also freely allowed by their permissive license.

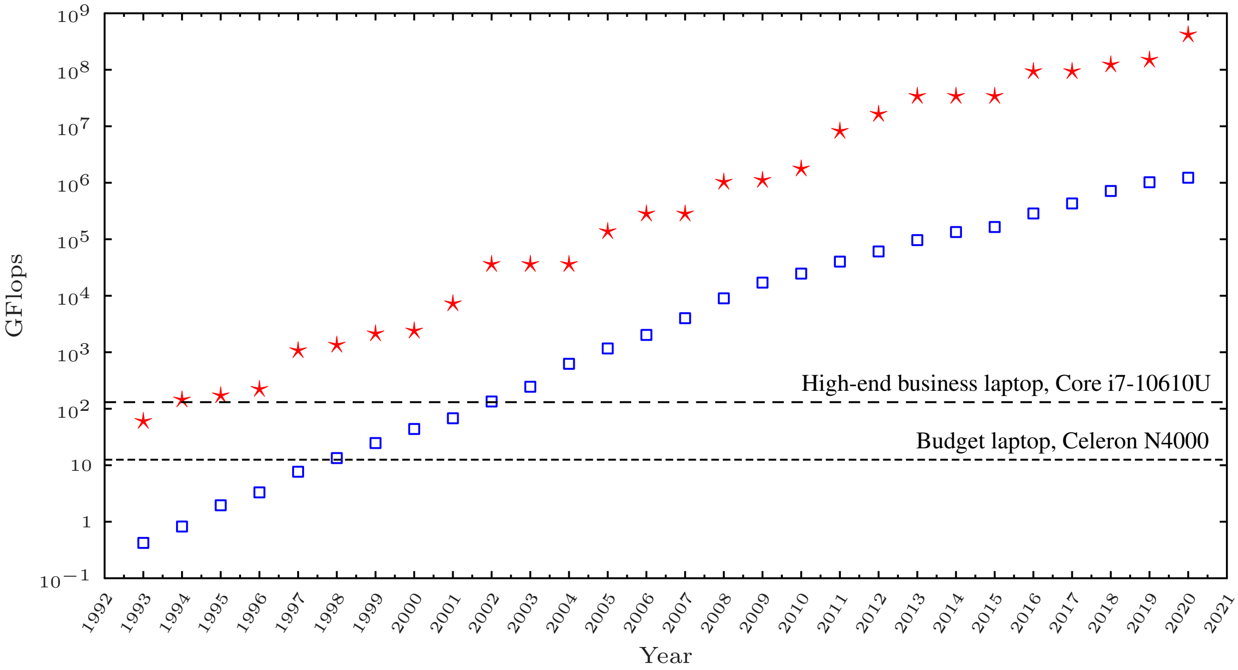

Figure 1: The best-performing (red stars) and worst-performing (blue squares) supercomputer on the TOP500 list,[100] as well as the performance of a budget laptop with a Celeron N4000 processor and a high-end business laptop with a Core i7-10610U processor (see supporting information). The performance of Raspberry Pi 4 was found to be similar to Celeron N4000.

3 Overview of available FOSS program packages

This section presents an overview of available FOSS program packages to illustrate the wide variety of codes that are available for research and educational use in computational chemistry. As the number of FOSS projects has grown immensely in recent years, we restrict the overview to self-contained packages which are able to run quantum electronic structure calculations from atomistic input. FOSS for other types of molecular modelling has been discussed elsewhere,[102,103] while various computational chemistry resources for education have been recently summarized by Rodríguez-Becerra et al.[104].

As FOSS, the programs listed here can be packaged and distributed openly without restriction; several of them are already available as part of Linux distributions such as Debian, Ubuntu, and Fedora Linux. It is not even necessary to be running Linux to use such prepackaged programs. Windows users can run the software under the Windows Subsystem for Linux (WSL), which allows installing and using a Linux distribution easily inside Windows 10. The cross-platform Python Package Index[105] (PyPI) and Conda[106] package managers are other alternatives for easy access to an increasing number of quantum chemistry packages both on Linux, Windows, and macOS. Computer laboratory settings can also be imitated using pre-made, customized live CDs or live USBs, for example.

Because of the large number of packages to review, we organize the discussion into

- programs for molecular calculations with Gaussian basis sets, section 3.1
- programs for solid-state calculations with various numerical approaches, section 3.2
- programs employing fully numerical methods, section 3.3
- programs employing tight-binding methods, section 3.4
- programs for visualization and manipulation, section 3.5

Due to space contraints, we only include minimalistic descriptions of the programs, and advise the reader to look up the programs' evolving capabilities in detail on the internet to assess their usefulness for a given computational chemistry course or other application. Most of the electronic structure programs support either Hartree–Fock (HF) and/or density-functional theory[107,108] (DFT); several molecular programs also support various post-HF methods.

3.1 Programs for molecular calculations with Gaussian basis sets

Bagel[109] is a C++ program package that features e.g. analytical CASPT2 [complete active space perturbation theory at the second order] nuclear energy gradients and derivative couplings, relativistic multireference wave functions based on the Dirac equation, and implementations of novel electronic structure theories.

Chronus Quantum[110] is a C++ program package that focuses on the consistent treatment of time dependence and spin in the electronic wave function, as well as the inclusion of relativistic effects in said treatments.

Dalton[111] is a Fortran program that specializes in molecular properties at various levels of theory, such as frequency-dependent response properties; one-, two-, and three-photon processes, etc. In addition to HF and DFT, Dalton features several post-HF methods like multiconfigurational self-consistent field (MCSCF) theory and coupled-cluster theory.

Ergo[112] is a C++ program for linear-scaling HF and DFT calculations for molecules.

ERKALE[70] is a C++ program implementing HF and DFT that specializes in the modeling of inelastic x-ray spectroscopies, self-interaction corrected density-functional theory, as well as various orbital localization methods.

e^T[113] is a C++ program primarily aimed for coupled-cluster calculations of molecular systems, which specializes in multiscale and multilevel methods, as well as modern Cholesky decomposition techniques for two-electron integrals.

JuliaChem[114] is a Julia package for HF calculations.

LSDalton[111] is a Fortran code targeted for linear-scaling HF and DFT calculations on large molecular systems, and also includes some coupled-cluster capabilities.

MolGW[115] is a Fortran/C++ package that implements HF and DFT, but specializes in many-body perturbation theory: the *GW* approximation and the Bethe–Salpeter equation.

MPQC[116] is a C++ program for massively parallel quantum chemistry, which originally focused on HF and DFT but has later evolved support for post-HF many-body theories.

NWChem[76] is a major quantum chemistry package written in Fortran and has a variety of features for both molecular and solid-state calculations.

Psi4[71] is a modular C++/Python package for HF, DFT and various post-HF calculations that can be used either as a traditional quantum chemistry package with simple and intuitive input files, or as Python modules for running calculations in Python.

PySCF[73] is a collection of Python modules for electronic structure calculations with significant capabilities also for solid-state simulations, including e.g. coupled-cluster implementations for crystalline systems.

PyQuante[117] is a Python package for quantum chemistry with some C extensions that emphasizes ease of understanding the code over performance.

OpenMolcas[74] is a Fortran package that specializes in multiconfigurational approaches to electronic structure theory, but also implements various DFT calculations, for example.

Serenity[118] is a C++ program for subsystem quantum chemical methods.

SlowQuant[119] is a Python program for molecular quantum chemistry that derives its name from the use of Python for even the computational demanding parts of the program.

VeloxChem[120] is a C++/Python package for molecular properties and for modeling various spectroscopies based on response theory.

Uquantchem[121] is a Fortran 90 program written for HF, DFT, Møller–Plesset perturbation theory, configuration interaction singles and doubles, quantum Monte Carlo, etc.

3.2 Programs for solid-state calculations

ABINIT[122] is Fortran program for plane wave calculations that supports density functional theory as well as more advanced formalisms like many-body perturbation theory.

Conquest[123] is a Fortran program for large-scale DFT calculations employing pseudo-atomic orbital basis sets.

CP2K[124] is a Fortran package based on Gaussian basis sets specializing in solid state physics, implementing HF, DFT, Møller–Plesset perturbation theory and the random phase approximation.

DFTK.jl[125] or the density-functional toolkit is a collection of Julia routines for experimenting with plane-wave density functional theory that emphasises simplicity and flexibility.

ELK[126], **EXCITING**[127], **and FLEUR**[128] are Fortran programs for linearised augmented-plane wave calculations which can reach microhartree accurate total energies for carefully chosen basis sets.

GPAW[129] is Python/C electronic structure program for DFT calculations within the projector-augmented wave approach which supports three modes of operation: (i) finite-difference grids, (ii) numerical atomic orbitals, and (iii) plane waves.

JDFTx[130] is a C++ plane wave DFT code aimed to be easy to develop and easy to use, whose key feature is support for joint density-functional theory for the description of electronic systems in contact with molecular liquids.

M-SPARC[131] is a MATLAB package for prototyping DFT calculations employing finite-difference grids and pseudopotentials.

OpenMX[132] is a C package for DFT calculations with pseudopotentials and numerical atomic orbitals.

PWDFT.jl[133] is a Julia package written from scratch to facilitate development of novel computational methods using plane waves.

Siesta[134] is a Fortran program for electronic structure calculations and ab initio molecular dynamics of molecules and solids that employs a basis set of numerical atomic orbitals, which are strictly localized, enabling the use of sparsity.

Qbox[135] is a C++ program aimed for first principles molecular simulations using plane waves and pseudopotentials.

Quantum Espresso[81] is a Fortran/C program for plane wave calculations with pseudopotentials on a wide range of hardware from laptops to supercomputers.

SPARC[136] is a C program for parallel DFT calculations employing finite-difference grids and pseudopotentials.

Note that the programs listed here that employ (pseudo)atomic basis functions, i.e. CONQUEST, CP2K, GPAW, Siesta, and OpenMX can naturally handle periodicity in 0, 1, 2, or 3 dimensions, corresponding to atoms and molecules, chains, sheets, and crystals, respectively. However, we have listed them as solid state codes because they are most often used for calculations with periodicity in three dimensions.

3.3 Programs relying on fully numerical representations

ACE-Molecule[137] is a C++ program that employs uniform real-space grids of Lagrange sinc functions, and supports density functional calculations on both periodic and non-periodic systems and wave function theory calculations based on Kohn–Sham orbitals.

BigDFT[138] is a Fortran program that is based on the use of pseudopotentials and a two-tier Daubechies wavelet basis to achieve a spatially localized basis.

DFT-FE[86] is a C++ program that employs spectral finite-element basis sets for a local real-space variational formulation of DFT, and is able to handle pseudopotential and all-electron calculations within the same framework and arbitrary periodicity.

HelFEM is a C++ program for fully numerical calculations on atoms[83,85] and diatomic molecules[82] at the HF or DFT levels of theory employing high-order numerical basis functions and yielding fully variational energies.

MADNESS[139] is a C++ program that relies on the use of multiresolution adaptive grids, which has been used in a variety of studies on novel real-space approaches to electron correlation, for instance.

MRChem[140] is a C++ program that also relies on multiresolution adaptive grids for Hartree–Fock and density functional calculations of molecules; its specialty is the computation of magnetic properties such as nuclear magnetic shielding constants.

Octopus[141] is a Fortran program based on pseudopotentials and finite difference grids that focuses on time-dependent density functional theory for handling non-equilibrium phenomena.

PARSEC[142] is a Fortran program based on finite-difference grids for density functional calculations with pseudopotentials.

RMG[143] is a C++/Fortran program employing real space grids and multigrid algorithms for density functional calculations with pseudopotentials.

x2dhf[144] is a Fortran program for non-relativistic finite difference restricted open-shell Hartree–Fock and density functional calculations on diatomic molecules.

3.4 Programs employing tight binding models

DFTB+[145] is a Fortran package for various calculations based on tight binding density functional theory.

Latte[146] is a Fortran program for density functional tight binding molecular dynamics.

Sparrow[147] is a C++/Python program for fast semiempirical quantum chemical calculations.

xtb[148] is a Fortran package that implements various semiempirical eXtended Tight-Binding methods.

3.5 Visualization, manipulation and analysis

For creating models and visualizing computational results, FOSS graphical user interfaces such as Jmol[149], Avogadro[150], IQmol[151] and PyMol[152] can be installed and used. The Atomic Simulation Environment (ASE)[153] contains versatile tools for building molecular and periodic models and enables easy retrieval of molecular structures from structural databases such as PubChem.[154] The interconversion of various input and output file formats between different programs can be carried out for example with the Open Babel[155] and cclib[156] packages. Postprocessing of calculations can be carried out with e.g. the Multiwfn[157] and ORBKIT[158] packages.

4 Illustrations of feasible computations

To enable a practical demonstration of the BYOD paradigm within computational chemistry education, it is time to illustrate the easy access to several powerful FOSS quantum chemistry packages in two widely used Linux distributions, Fedora and Ubuntu. The Supporting Information contains practical step-by-step examples of combining the BYOD paradigm with FOSS packages to run quantum chemical calculations according to the BYOD-FOSS paradigm. Four different program packages are used in the practical illustrations: xtb (section 4.1), NWChem (section 4.2), Psi4 (section 4.3), and Quantum Espresso (section 4.4). Installation instructions are provided for each code and all examples can be run under Linux, macOS, or the Windows Subsystem for Linux. In all cases, the software can be installed in a matter of minutes on a personal computer, either using a Linux distribution package manager or the Conda package manager. For convenience, the supporting information is also available as a git repository.[159]

4.1 xtb

The primary design goal of xtb has been the fast calculation of structures and noncovalent interaction energies for molecular systems with up to roughly 1000 atoms.[148,160] Even though the GFNn-xTB methods implemented in xtb are parametrized semiempirical quantum chemical methods, they are widely applicable as they have been parametrized for the whole periodic table up to radon ($Z = 86$). A highly attractive feature of xtb is its performance: calculations on small molecules (10–20 atoms) finish in matter of seconds even on a low-powered laptop computer. xtb is a powerful tool for example in the pre-optimization of geometries and molecular conformations before computationally more demanding calculations; see ref. 161 for a recent application to water oxidation catalysis, for example.

The Supporting Information includes step-by-step guidelines for installing xtb and using it to study structures, conformations, energetics, and molecular orbitals of inorganic and organic molecules. Calculations on pharmaceutically relevant cisplatin and transplatin molecules shown in figure 2 are briefly summarized here to showcase the basic use of xtb. Cisplatin, *cis*-[$\mathrm{Pt(NH_3)_2Cl_2}$], is a chemotherapy medication used in cancer treatments whose stereoisomer, transplatin, *trans*-[$\mathrm{Pt(NH_3)_2Cl_2}$], is ineffective in cancer treatment.

The Pt(II) atom is square-planar coordinated in both cisplatin and transplatin. Which configuration, *cis* or *trans*, is lower in energy? We use the xtb program to answer this question. The first task is to have initial geometries for the two molecules. In general,

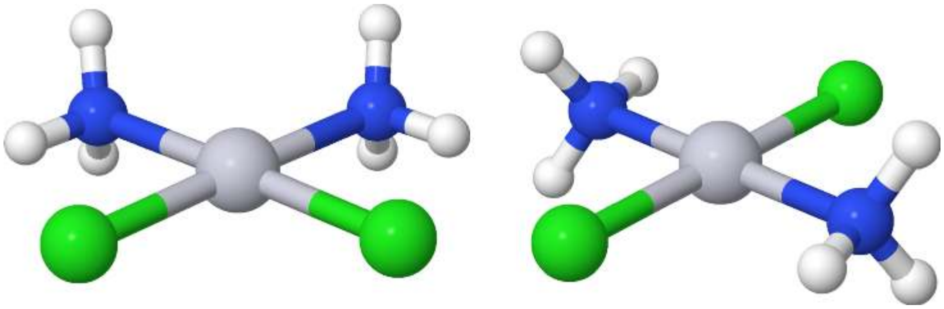

Figure 2: Cisplatin (left) and transplatin (right). Color coding: Pt = gray, Cl = green, N = blue, and H = white.

initial geometries can be obtained from structural databases such as Pubchem;[154] built in a graphical user interface with programs such as Jmol, Avogadro, or IQMol; or built by hand in internal coordinates (bond lengths, angles and dihedrals) in the Z-matrix formalism, for example. Hand-built molecular geometries for cisplatin and transplatin are given in XYZ format in figures 3 and 4, respectively. While these geometries should be sufficiently close to optimal to allow for a straightforward optimization without difficulties, they are still quite rough in that the total energy is expected to change by several millihartrees in the geometry optimization, corresponding to changes in the energy of several kcal/mol.

The next step is to bring both molecules into a (local) minimum of the potential energy surface (PES) by optimizing the geometries with xtb. The point groups of the initial geometries are approximately C_{2v} and C_{2h} for cisplatin and transplatin, respectively, but symmetry is not enforced during the xtb optimizations. The only input needed by xtb in this case are the cartesian coordinates of both molecules in XYZ format, which were given in figures 3 and 4 for cisplatin and transplatin, respectively.

The geometry optimizations complete in seconds even on a low-powered computer; the supporting information (SI) contains all of the necessary inputs. For cisplatin, the optimized Pt–Cl and Pt–N distances are 2.24 Å and 2.15 Å, respectively. Considering the relatively low level of theory, the obtained distances are in reasonable agreement with the Pt–Cl and Pt-N distances of 2.25 Å and 2.06 Å, respectively, obtained with much higher-level methods by Tasinato et al.[162] (they employed coupled-cluster theory with full single and double

```
11
cis-[Pt(NH3)2Cl2] (cisplatin); angstrom units
 Pt                 0.00000000   -0.00000000   -0.19134710
 Cl                 0.00000000    1.61220407    1.42085566
 Cl                 0.00000000   -1.61220407    1.42085566
 N                  0.00000000    1.40714181   -1.59849021
 H                  0.81649658    1.30951047   -2.16752575
 H                 -0.81649658    1.30951047   -2.16752575
 N                  0.00000000   -1.40714181   -1.59849021
 H                 -0.81649658   -1.30951047   -2.16752575
 H                  0.81649658   -1.30951047   -2.16752575
 H                  0.00000000    2.30951093   -1.16752621
 H                  0.00000000   -2.30951093   -1.16752621
```

Figure 3: Molecular geometry of cisplatin in XYZ format.

```
11
trans-[Pt(NH3)2Cl2] (transplatin); angstrom units
 Pt                 0.00000000    0.00000000    0.00000000
 Cl                 2.27999997   -0.00036653    0.00000000
 Cl                -2.27999997    0.00036653    0.00000000
 N                 -0.00031991   -1.98999997    0.00000000
 H                  0.46944690   -2.32340883   -0.81740913
 H                  0.46944690   -2.32340883    0.81740913
 N                  0.00031991    1.98999997    0.00000000
 H                 -0.46944690    2.32340883   -0.81740913
 H                 -0.46944690    2.32340883    0.81740913
 H                  0.94318252    2.32318174    0.00000000
 H                 -0.94318252   -2.32318174    0.00000000
```

Figure 4: Molecular geometry of transplatin in XYZ format.

substitutions and perturbative triple substitutions, CCSD(T)).

Comparing the total energies of the two stereoisomers after geometry optimization shows that the total energy of transplatin is 20 kJ/mol lower, that is, more negative than that of cisplatin. This means that transplatin is the energetically more favorable stereoisomer of diamminedichloroplatinum(II), [Pt(NH_3)$_2$$Cl_2$]. For comparison, Liu and Franke[163] reported an energy difference of 56 kJ/mol with a much higher level of theory: relativistic CCSD(T) employing direct perturbation theory, a 13s9p7d5f2g contracted Gaussian basis for Pt and aug-cc-pVQZ for other elements, evaluated on top of molecular geometries optimized for the Becke'88–Perdew'86 functional.[164,165] The result from xtb, which we were able to get in a matter of seconds, is in good qualitative (or even semiquantitative!) agreement with the result obtained with the high level of theory. Next, in section 4.2, we will revisit cisplatin and transplatin with DFT calculations that afford a step up in accuracy over xtb.

4.2 NWChem

NWChem is a program that has been developed for almost 30 years. Consequently, a large number of features are available in the code: HF, DFT, as well as post-HF calculations, ab initio molecular dynamics, and so on. NWChem has been designed to run on high-performance parallel supercomputers as well as on conventional workstations. The Supporting Information includes step-by-step guidelines for installing NWChem and using it to study the same pharmaceutically relevant cisplatin and transplatin molecules that were studied with xtb in section 4.1.

We choose to use non-empirical DFT in the NWChem examples. Although NWChem also includes more accurate *ab initio* methods such as coupled-cluster theories, we shall not consider them in this work since their proper use requires much more understanding and computational power than DFT does, and as such methods are typically not included in undergraduate level courses. We choose the non-empirical PBE0 hybrid functional[78,166,167] (sometimes also known as hybrid PBE or PBEh) that provides reasonable geometries and energetics across the periodic table and shows good performance for complexes with d- and f-metals.[168,169]

Even though DFT is simpler than many post-HF theories, setting up adequate DFT calculations still requires some considerations. The one-electron basis set is one of the most important aspects to consider in electronic structure calculations in general, like our attempted PBE0 calculation with NWChem, as the choice of the one-electron basis set has an immense importance on the computational cost and accuracy of the resulting calculations. While the GFNn-xTB methods discussed above in section 4.1 did not require the specification of a basis set, as the basis set is an essential part of the specification of the GFNn-xTB methods themselves, the basis set—which parametrizes the allowed degrees of freedom for the movement of the electrons—does need to be specified for HF, DFT and post-HF calculations. Because of the profound importance of the choice of the basis set, various types of Gaussian basis sets have a long history in quantum chemistry,[170] and hundreds of Gaussian basis sets intended for various purposes are available on the Basis Set Exchange,[171] for example.

Although many readers will be familiar with traditional basis sets like STO-3G,[172] 3-21G[173] and 6-31G*,[174] the development of computer processors and quantum chemical models in recent decades have also lead to significant advances in basis set design; we refer the reader to the reviews by Hill[175] and Jensen[176], for example. Because the basis set is an approximation, it is highly desirable to be able to control its accuracy in order to make tradeoffs between the cost of the calculation and the accuracy of the obtained results. Accordingly, modern basis sets typically come in families of varying size:[175,176] the smallest sets enable quick but qualitative calculations, while the larger sets enable quantitative computations at the cost of more computer time. In contrast to traditional basis sets, modern basis set families allow for a cost-efficient approach to the complete basis set limit, at which point the error in the one-electron basis set no longer affects the calculation. Note that also other types of basis sets than Gaussians may be used for quantum chemistry, see ref. 177 for further discussion.

In this work, we will only consider the Karlsruhe def2 family of Gaussian basis sets,[178] which are a good all-round choice for general chemistry as they are available for the whole periodic table up to radon ($Z = 86$). As radon is an element of the 6th period, while relativistic effects are already essential for chemistry of the 5th row,[179,180] relativistic effects are described in the def2 basis sets through the use of effective core potentials (ECPs).[181] The ECP is used to describe the chemically inactive, deep-core electrons only implicitly; this also decreases the overall cost of the calculation.

The Karlsruhe def2 sets come at three levels of accuracy. Split-valence (SV) basis sets are the smallest reasonable basis set for general applications. The def2-SVP basis is a SV basis set with polarization (P) functions, and is similar in size to the 6-31G** also known as the 6-31G(d,p) basis set. Like 6-31G**, the def2-SVP set can also be used without polarization functions on hydrogen atoms; this basis is called def2-SV(P), it is smaller than the 6-31G* basis, and it is often useful for quick qualitative/semi-quantitative calculations. For more quantitative calculations, the def2 series also contains a triple-ζ valence polarization set (def2-TZVP) as well as a quadruple-ζ valence polarization set (def2-QZVP), which typically suffice for achieving the complete basis set limit in HF and DFT calculations. Calculations at post-HF levels of theory, however, require larger basis sets with additional polarization functions; the def2-TZVPP and def2-QZVPP basis sets exist for this purpose. Diffuse functions (D) are necessary for the proper description of anions as well as to model e.g. electric polarizabilities; sets are likewise available at all levels of accuracy (def2-SVPD, def2-TZVPD, def2-TZVPPD, def2-QZVPD, def2-QZVPPD) for this purpose.[182]

For the present demonstration, we choose the def2-TZVP basis set, as triple-ζ basis sets are well-known to yield energies that are sufficiently close to the complete basis set limit (see also the applications in subsections 4.3.1 and 4.3.2). Although hybrid functionals are computationally more demanding than non-hybrid functionals, it is notable that the dispersion-corrected PBE0-D4 functional was recently shown to outperform the dispersion-corrected, meta-GGA-type r^{2}SCAN-D4 functional in accuracy even for reaction energies of metal–organic reactions.[183]

Having completed our introduction to DFT calculations, basis sets, and NWChem, similarly to the workflow in the case of xtb, the first task is to bring both molecules into a (local) minimum of the potential energy surface (PES) by means of geometry optimization. The geometry optimization is started from the same hand-built initial geometries presented in section 4.1. In contrast to xtb, NWChem is capable of employing the point group symmetry (C_{2v} and C_{2h} for cisplatin and transplatin, respectively) during the geometry optimization in order to speed up both the electronic structure calculation as well as the geometry optimization, and will do so by default. This means that the calculation runs faster, but also that the molecule is constrained to the same point group as the initial geometry during the whole optimization. If the user is not careful, this may also be a bad thing, as the use of symmetry may sometimes lead to convergence to a saddle point instead of a local minimum.

The input required for NWChem is more complicated than that for xtb. Running NWChem requires setting up an input file that contains various computational parameters in addition to the input geometry. Fully annotated input files can be found in the SI, a shortened example is shown in figure 5.

The geometry optimizations of cisplatin and transplatin finish in a matter of minutes on one processor core, depending on the used computer. The optimized Pt–Cl and Pt–N distances for cisplatin are 2.28 Å and 2.08 Å, respectively. These values are in excellent agreement with the values of Tasinato et al.[184] that were discussed in section 4.1, that is, Pt–Cl and Pt–N distances of 2.25 Å and 2.06 Å, respectively: the geometries agree to 0.03 Å.

Next, comparing the total PBE0/def2-TZVP energies of the two stereoisomers shows that transplatin is 54 kJ/mol lower (more negative) than cisplatin. Our DFT value is in good quantitative agreement with the energy difference of 56 kJ/mol obtained by Liu and Franke[163] using a high-level CCSD(T) method; however, our calculations can be performed

```
title "Cisplatin"
charge 0
geometry units angstroms autosym 0.1
 Pt                0.00000000   -0.00000000   -0.19134710
 Cl                0.00000000    1.61220407    1.42085566
 Cl                0.00000000   -1.61220407    1.42085566
 N                 0.00000000    1.40714181   -1.59849021
 H                 0.81649658    1.30951047   -2.16752575
 H                -0.81649658    1.30951047   -2.16752575
 N                 0.00000000   -1.40714181   -1.59849021
 H                -0.81649658   -1.30951047   -2.16752575
 H                 0.81649658   -1.30951047   -2.16752575
 H                 0.00000000    2.30951093   -1.16752621
 H                 0.00000000   -2.30951093   -1.16752621
end
dft
  xc pbe0
  mult 1
  iterations 100
end
basis spherical
  * library def2-tzvp
end
ecp
  Pt library def2-ecp
end
driver
  maxiter 100
  xyz
end
task dft optimize
```

Figure 5: NWChem example: PBE0/def2-TZVP geometry optimization of cisplatin; for transplatin, the nuclear coordinates given in figure 4 are used, instead.

in a matter of minutes even on a personal computer.

For cisplatin, we also write out the molecular orbitals after the geometry has been optimized. The molecular orbitals provided by from the non-empirical PBE0/def2-TZVP calculations can now be compared to the ones from the semiempirical xtb calculations from section 4.1, see figure 6. The frontier orbitals—the highest occupied molecular orbital (HOMO) as well as the lowest unoccupied molecular orbital (LUMO)—from the xtb and NWChem calculations are in good agreement. Also HOMO−3, HOMO−2 and HOMO−1 appear similar; the HOMO−2 and HOMO−1 orbitals are merely switched between the NWChem and xtb calculations.

From the point of view of crystal field theory, the Pt(II) atom in cisplatin has a square planar coordination and eight $5d$ electrons. The four HOMOs and the LUMO all involve Pt $5d$ orbitals. In line with crystal field theory, both NWChem and xtb show that the LUMO involves the Pt $5d_{x^2-y^2}$ orbital. HOMO−3 involves the Pt $5d_{z^2}$ orbital, while the $5d_{xy}$, $5d_{xz}$, and $5d_{yz}$ orbitals contribute to HOMO−2, HOMO−1, and HOMO. As is clearly seen from the data presented above, the non-empirical PBE0/def2-TZVP and the semiempirical GFN2-xTB level of theory provide a similar description of the frontier orbitals of the Pt(II) complex. Again, the full inputs for the calculations are given in the SI.

4.3 Psi4

While NWChem represented older and more established quantum chemistry codes, Psi4 represents the newer generation of quantum chemistry codes. The origins of Psi4 trace to the Psi3 research code written in C++ for high-accuracy studies on small molecules.[72] At variance to Psi3, Psi4 is meant to be a user-friendly, general-purpose code for fast, automated computations on molecules with hundreds of atoms.[71] Psi4 contains a number of computational methods ranging from HF and DFT to post-HF methods such as Møller–Plesset perturbation theory,[185] coupled-cluster theory,[186] configuration interaction theory, orbital-optimized correlation methods, symmetry-adapted perturbation theory, multirefer-

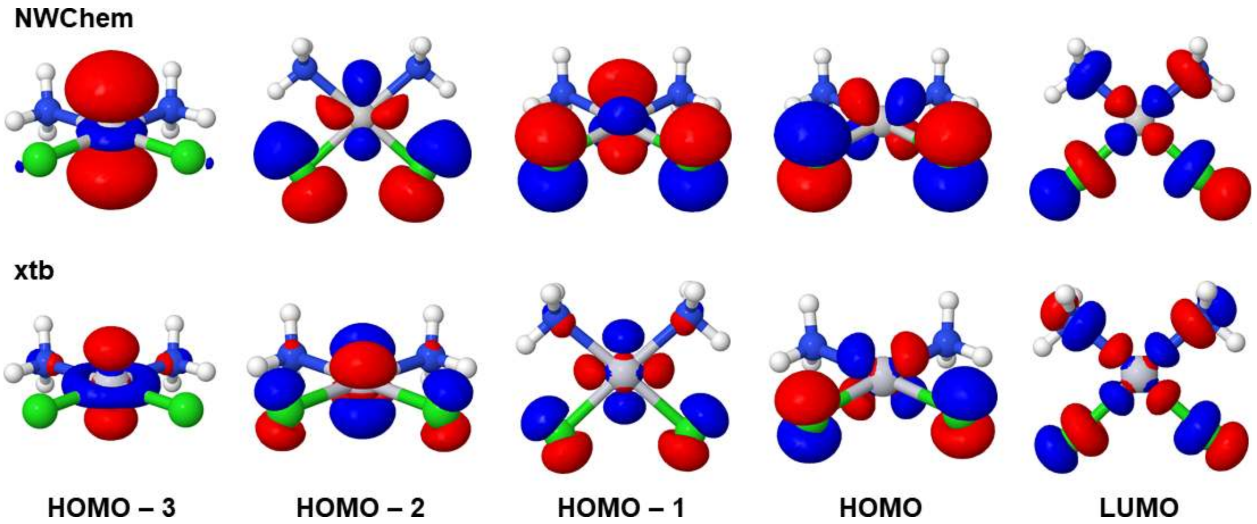

Figure 6: The four highest occupied MOs (HOMOs) and the lowest unoccupied MO (LUMO) of cisplatin as obtained from NWChem (PBE0/def2-TZVP) and xtb (GFN2-xTB). The color code for the nuclei is the same as in figure 2, while red and blue denote positive and negative orbital amplitudes, respectively (note that the overall sign of the orbital can be freely chosen). The isovalue used for the orbitals is 0.04 electrons/Bohr^3.

ence methods *etc.*[71] Although the core of the program is still in C++, Psi4 has thorough Python interfaces and can be used either as a traditional quantum chemistry program with input files, or directly from Python.

We will demonstrate the use of Psi4 in the context of two common exercises in elementary courses on computational chemistry: a conformational study of methylcyclohexane and the reproduction of the molecular geometry of the chromyl fluoride (CrO_2F_2) molecule, with a special consideration to the one-electron basis set. We again focus on the def2 family of basis sets that was introduced in section 4.2.

4.3.1 Methylcyclohexane

Starting out with the conformational study of methylcyclohexane, the workflow is as follows. First, the molecule is built in a molecular editor such as Avogadro, IQmol or Jmol, and the drawn molecular structure is preoptimized using a force field available in the program; the goal of the preoptimization is merely to ensure that the bond lengths are realistic so that the electronic structure calculations during the geometry optimization converge without problems.

In the next step, the molecular structure is reoptimized with xtb, and a conformational search is carried out with xtb with the CREST program [Conformer-Rotamer Ensemble Sampling Tool] which has been shown to reproduce conformational ensembles to good accuracy.[187–189] Again, the SI includes short tutorials for installing and using the CREST code, which employs xtb to carry out conformational searches of molecules.[188]

The six conformers found by CREST are then reoptimized in Psi4 using the PBE0/def2-TZVP[78,166,167,178] level of theory introduced above in section 4.2. Psi4 employs density fitting[190–194] by default; this means that the universal fitting basis for Hartree–Fock calculations[195] is used in the calculation. The Psi4 input file for the first conformer is shown in figure 7. The inputs for the other molecules are analogous and shall not be repeated here; they are, however, available in the SI.

With the PBE0/def2-TZVP optimized geometries at hand for each of the six conformers which are given out by CREST in increasing order in energy, we perform single-point calculations on each conformer in a variety of basis sets; the resulting energy differences to the lowest-energy conformer (#1) are given in table 1. In addition to the def2 family, we also have included data for the MINAO basis consisting of the minimal-basis Hartree–Fock orbitals extracted from the triple-ζ cc-pVTZ basis set,[196] as well as the STO-3G and STO-6G basis sets which are 3-Gaussian and 6-Gaussian function expansions of a minimal-basis Slater-type orbital (STO) basis set, respectively.[172] (It is important to note in this context that not all STO basis sets are minimal: STO basis sets of various sizes ranging up to polarized quadruple-ζ have been reported[197,198] and remain widely used for practical calculations in programs employing STO basis sets.)

The data in table 1 leads us to the following insights. First, even the minimal basis sets successfully predict the energy ordering of the conformers: although MINAO flips the order of conformers 5 and 6, it still predicts conformer 1 to be the lowest in energy. Note that this comparison is restricted to the use of fixed geometries; relaxing the geometries in each basis

```
molecule {
0 1
 C          1.0138209743        0.0000504938       -0.3320143257
 C          0.3011766819       -1.2490459958        0.1880687301
 C          0.3011705532        1.2491334547        0.1881225711
 C         -1.1681795797       -1.2518490761       -0.2231999547
 C         -1.1681646702        1.2518017380       -0.2232353354
 C         -1.8704369468       -0.0000107259        0.2931516386
 C          2.4834462973       -0.0000526508        0.0793244864
 H          0.9578296121        0.0001317793       -1.4269643433
 H          0.3719502854       -1.2738045338        1.2783433519
 H          0.7954821386       -2.1434306899       -0.1983308514
 H          0.7955120205        2.1435500894       -0.1982165735
 H          0.3719418465        1.2737371458        1.2784081110
 H         -1.6613232246       -2.1444573858        0.1682458589
 H         -1.2389521557       -1.2820854090       -1.3132456880
 H         -1.2389247893        1.2821507899       -1.3132924946
 H         -1.6613748878        2.1443291541        0.1682793278
 H         -2.9152913775       -0.0000182176       -0.0249822935
 H         -1.8531032928       -0.0000579011        1.3856452460
 H          2.9894549975        0.8826673032       -0.3058691699
 H          2.5743314259       -0.0008035829        1.1637849282
 H          2.9895575018       -0.8822331319       -0.3068953206
}

set basis def2-tzvp
optimize('pbe0')
```

Figure 7: Psi4 example: PBE0/def2-TZVP geometry optimization for the lowest-lying methylcyclohexane conformer.

might change the conclusion somewhat. The good performance of the minimal basis sets for this application shows that conformational energies enjoy an excellent degree of error cancellation, which is one of the main motivations for using atomic basis sets in the first place.[177]

The shortcomings of minimal basis sets are showcased by the large differences between the results obtained with the MINAO and STO-nG basis sets. MINAO is derived from atomic calculations only, and is thereby fully biased towards atoms. In contrast, the Slater-type orbital basis used by Hehre et al.[172] is optimized for an average molecular environment, which is reflected in the slightly improved results in table 1.

Minimal basis sets are as small as possible and thereby have very little flexibility: good accuracy for one type of system does not translate to good accuracy in another system. Indeed, minimal basis sets generally have poor predictive power for chemistry.[175,176] In this case, the bias towards molecules means that the STO-nG basis sets are not as good for isolated atoms. It is generally preferable to use larger and more flexible basis sets in applications, which guarantee a uniform accuracy for all types of systems, and to try to converge results to the complete basis set limit. This means controllably removing the error made in the one-electron basis set approximation until the error becomes negligible either in absolute value, or in comparison to the other sources of error in the calculation, such as the error inherent in the employed density functional approximation, for example.

As has already been previously discussed, the smallest reasonable basis for general applications is def2-SV(P). It predicts conformational energies roughly within 0.3 kcal/mol compared to the converged quadruple-ζ values, as can be seen from table 1. As shown by the comparison between the def2-SV(P) and def2-SVP data, the role of polarization functions on hydrogen is small for the studied conformational energies.

Systematically more converged energies are obtained by going to the triple-ζ def2-TZVP basis and the quadruple-ζ def2-QZVP basis. The data show that already the triple-ζ calculations are converged to 0.01 kcal/mol in the relative energies, demonstrating the usefulness of modern, systematic basis set families: the complete basis set limit can be reached by using larger and larger basis sets.

For comparison, table 1 also includes data for the GFN2-xTB method.[160] A visual assessment of the data confirms that GFN2-xTB correctly reproduces the energy ordering of the conformers even with the used PBE0/def2-TZVP geometries, and that the relative energies are reproduced at an accuracy comparable to the minimal basis set calculations, with the converged PBE0/def2-QZVP data as reference. This data emphatically suggests that historical applications of minimal basis sets in quantum chemistry can be straightforwardly replaced with modern semiempirical calculations with xtb, for instance, which have much lower computational cost.

Table 1: Relative energies $\Delta E^{\text{conformer } n} = E^{\text{conformer } n} - E^{\text{conformer } 1}$ in kcal/mol and number of basis functions N_{bf} for the methylcyclohexane conformers according to PBE0 calculations with various basis sets, evaluated at the PBE0/def2-TZVP optimized geometries. For comparison, GFN2-xTB data is also included.

method	N_{bf}	conformer 2	conformer 3	conformer 4	conformer 5	conformer 6
PBE0/STO-3G	49	1.19	2.99	1.18	5.54	5.78
PBE0/STO-6G	49	1.25	2.97	1.24	5.57	5.84
PBE0/MINAO	49	0.86	2.46	0.84	5.11	5.05
PBE0/def2-SV(P)	126	2.00	3.36	2.00	6.62	7.07
PBE0/def2-SVP	168	1.97	3.32	1.97	6.57	7.01
PBE0/def2-TZVP	301	2.10	3.04	2.10	6.31	6.74
PBE0/def2-QZVP	819	2.11	3.03	2.11	6.31	6.73
GFN2-xTB		1.35	2.58	1.35	5.21	5.19

4.3.2 Geometry of chromyl fluoride

For a somewhat more complicated example, we study the equilibrium geometry of chromyl fluoride (CrO_2F_2) at various levels of DFT, which is known to be surprisingly accurate for simple transition metal complexes.[199] CrO_2F_2 assumes a tetrahedral geometry. Again, the workflow is to build the molecule in a molecular editor, preoptimize the molecular geometry with xtb, and then run the geometry optimizations in Psi4; however, now the optimization is done separately for each basis set in contrast to the procedure used in section 4.3.1.

For this study, we choose the GFN1-xTB[200] and GFN2-xTB[160] semiempirical methods as well as a set of nonempirical density functionals: the Perdew–Wang 1992 (PW92) local density approximation (LDA),[201–203] the Perdew–Burke–Ernzerhof (PBE) generalized-gradient approximation (GGA),[78] as well as the r^{2}SCAN meta-GGA functional that represents the state of the art in non-empirical density functionals.[204,205] The geometry optimizations are undertaken with very tight convergence thresholds to ensure benchmark quality geometries.

Density fitting is again used in these calculations. As we only consider density functionals that do not contain exact exchange in this application, smaller auxiliary basis sets optimized for reproducing only Coulomb interactions could be employed;[206] however, for simplicity we stick to using the Psi4 default which is to use the larger auxiliary basis sets[195] that also work in the presence of exact exchange.

The results shown in table 2 demonstrate that while the STO-nG minimal basis sets[172,207] yield relatively poor geometries compared to the experimental values from refs. 208 and 209, already the split-valence def2-SV(P) basis set[178] leads to bond lengths that are converged to 0.03 Å and fractions of a degree in angles. The differences become smaller, that is, the bond lengths and angles become more converged going to the larger basis sets, with the differences between the def2-TZVP and def2-QZVP results being already negligible.

The bond lengths from the PBE/def2-QZVP calculations are in excellent agreement with the older experimental values from ref. 208; the bond angles are in reasonable agreement with the experimental data from the same reference. r^{2}SCAN/def2-QZVP, in turn, is in excellent agreement with the newer experimental bond lengths from ref. 209.

4.4 Quantum Espresso

Quantum Espresso (QE) is an integrated suite of FOSS codes for electronic structure calculations based on DFT, plane waves, and pseudopotentials. The QE distribution consists of a set of core components and programs, set of plug-ins for more advanced tasks, and a number of third-party packages designed to be interoperable with the core components. QE can be

Table 2: Geometric parameters of chromyl fluoride (CrO_2F_2) at various levels of theory. [a]Experimental values from ref. 208. [b]Experimental values from ref. 209.

method	basis	r(CrF) (Å)	r(CrO) (Å)	∠(OCrO) (°)	∠(FCrF) (°)
GFN1-xTB		1.525	1.597	111.37	106.53
GFN2-xTB		1.548	1.671	111.50	110.38
PW92	STO-3G	1.491	1.584	109.44	108.14
	STO-6G	1.495	1.589	109.59	107.71
	def2-SV(P)	1.548	1.684	108.41	110.80
	def2-SVP	1.541	1.675	108.35	110.58
	def2-TZVP	1.551	1.693	108.33	110.26
	def2-QZVP	1.554	1.695	108.20	110.48
PBE	STO-3G	1.504	1.606	109.47	108.05
	STO-6G	1.507	1.611	109.61	107.65
	def2-SV(P)	1.565	1.713	108.41	110.75
	def2-SVP	1.557	1.704	108.38	110.48
	def2-TZVP	1.568	1.721	108.45	110.01
	def2-QZVP	1.571	1.724	108.30	110.23
r^2SCAN	STO-3G	1.497	1.602	109.98	106.94
	STO-6G	1.500	1.605	110.26	106.22
	def2-SV(P)	1.553	1.700	108.83	109.48
	def2-SVP	1.545	1.692	108.77	109.25
	def2-TZVP	1.554	1.706	108.89	108.80
	def2-QZVP	1.556	1.708	108.76	108.96
experiment[a]		1.575	1.720	107.8	111.9
experiment[b]		1.55	1.71		

used to study the geometries, energetics, thermodynamics, electronic properties, response properties, spectroscopic properties, and transport properties of solid-state materials. The Supporting Information includes step-by-step guidelines for installing QE and using it to study two polymorphs of zinc(II) sulfide, ZnS.

ZnS crystallizes in two principal forms, sphalerite and wurtzite (figure 8). Sphalerite is a naturally occurring mineral belonging to the cubic crystal system with space group F[-]43m (No. 216). Both Zn and S atoms are tetrahedrally coordinated in the sphalerite structure and the crystal structure can be considered as a diamond lattice with two atom types. Wurtzite is also a naturally occurring mineral and it can be considered as a hexagonal polymorph of sphalerite, crystallizing in the space group $P6_3mc$ (No. 186). The coordination with nearest and next-nearest neighbors in wurtzite is identical to that in sphalerite. The first structural differences between the two polymorphs arise only in the third shell of neighbors.[210] From a thermodynamical point of view, sphalerite is the low-temperature ZnS polymorph in bulk form and the transition temperature to wurtzite is 1293 ± 10 K.[211] Wurtzite-ZnS is thus metastable at room temperature, but it is found in nature and can also be produced synthetically.

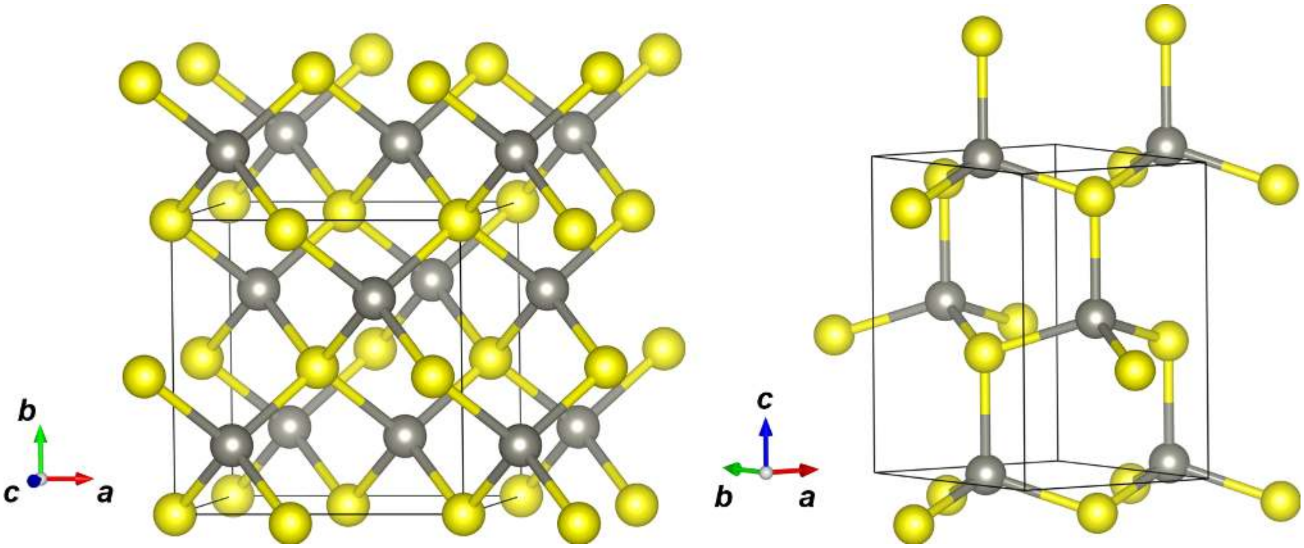

Figure 8: Two polymorphs of ZnS: sphalerite (left) and wurtzite (right).

The illustrative QE calculations are carried out with the non-empirical PBE exchange-

```
&CONTROL
  calculation=’vc-relax’
  prefix=’zns’
/
&SYSTEM
  space_group=216 ! Space group
  a=5.4093        ! Lattice parameter a in angstroms
  nat=2           ! Number of atoms in the asymmetric unit
  ntyp=2          ! Number of different atom types. Here, Zn and S.
  ecutwfc=40      ! Kinetic energy cutoff for wavefunctions (Ry)
  ecutrho=200     ! Kinetic energy cutoff for charge density and potential (Ry)
/
ATOMIC_SPECIES
  Zn 65.38 zn_pbe_v1.uspp.F.UPF
  S  32.065 s_pbe_v1.4.uspp.F.UPF
ATOMIC_POSITIONS crystal_sg
Zn 0.00000 0.00000 0.00000
S  0.25000 0.25000 0.25000
K_POINTS automatic
8 8 8  0 0 0
```

Figure 9: Quantum Espresso example: Geometry optimization of sphalerite-ZnS with PBE functional and GBRV pseudopotentials. Fully annotated input files can be found from the SI.

correlation functional.[78] To run the calculations with QE, we need pseudopotentials that have been developed for this functional. Here we use the ultrasoft Garrity–Bennett–Rabe–Vanderbilt (GBRV) pseudopotentials, which form a highly accurate and computationally inexpensive open-source pseudopotential library that has been designed and optimized for use in high-throughput DFT calculations.[212] The main attractive feature of the GBRV pseudopotentials is that they have been designed for relatively low plane wave cutoffs of 40 Rydberg for wave functions, and 200 Rydberg for the charge density and potential,[212] which results in relatively low computational costs.

To study sphalerite-ZnS and wurtzite-ZnS with QE, we need their crystal structures. A good source for crystal structure data is the Crystallography Open Database (COD),[213] and the crystal structures used as a starting point have been obtained from there in the Crystallographic Information File (CIF) format and they are included in the Supporting Information. There are several ways in which to enter the crystal structures in QE input files. In the example here, we have directly used the crystallographic information to create an input file, which is shown in figure 9; a very helpful resource for building QE input files is afforded by the QE input generator and structure visualizer provided by the Materials Cloud.[214]

Before geometry optimizations, it is important to decide how dense a sampling of the reciprocal space (k-sampling) is needed to describe the materials sufficiently accurately. The convergence tests described in the Supporting Information show that a $8 \times 8 \times 8$ k-mesh leads to a truncation error smaller than 1 meV for sphalerite-ZnS. A comparable k-point spacing is then also used for wurtzite-ZnS.

The geometry optimization of sphalerite-ZnS finishes in a few minutes, while the wurtzite-ZnS may take tens of minutes when run on a single processor core. The optimized lattice parameters are in good agreement with the experimental lattice parameters found on COD. The optimized lattice parameters are $a = 5.447$ Å for sphalerite-ZnS and $a = 3.846$ Å and $c = 6.304$ Å for wurtzite-ZnS, whereas the experimental lattice parameters are $a = 5.4093$

Å for sphalerite-ZnS and a = 3.811 Å and c = 6.234 Å for wurtzite-ZnS.[213] This means that the computations overestimate the lattice parameters by approximately 1% over the experiment.

The energy comparison of the optimized sphalerite-ZnS and wurtzite-ZnS structures shows that the total energies differ by only 0.6 kJ/mol per formula unit. This value is in good agreement with Cardona et al.[215] who reported an energy difference of less than 0.008 eV (0.8 kJ/mol) per formula unit from LDA and GGA calculations on ZnS polymorphs. The energy difference is so small, because the crystal structures are so similar: differences arise only in the third-nearest neighbor shell. Note that so far we have only compared electronic total energies; Gibbs free energies should be considered instead for a full understanding of the thermodynamics, but this is beyond the scope of this work.

The second practical example illustrates how the electronic band structure of sphalerite-ZnS can be calculated and plotted with QE. In any band structure calculation, the band path in the reciprocal space has to be defined in terms of k-points. The band path depends on the Bravais lattice of the crystal structure. An excellent source for band paths is the SeeK-path service,[216] which readily provides crystal-structure-based band paths for several program packages. Here, we use the face centered cubic (FCC) band path from Setyawan and Curtarolo[217], and the resulting electronic band structure of sphalerite-ZnS is illustrated in figure 10.

From the band structure plot in figure 10, we can see that sphalerite-ZnS has a direct band gap of about 2 eV at the Γ point when using the PBE functional and the GBRV pseudopotentials. The band structure in figure 10 is in good agreement with the PBE band structure available in the Materials Project.[218] However, the PBE calculations severely underestimate the experimental band gap measured at 10 K, which is about 3.8 eV.[219] The agreement with experiment could be improved for example with the DFT+U approach or with hybrid density functionals which are outside the scope of this work.

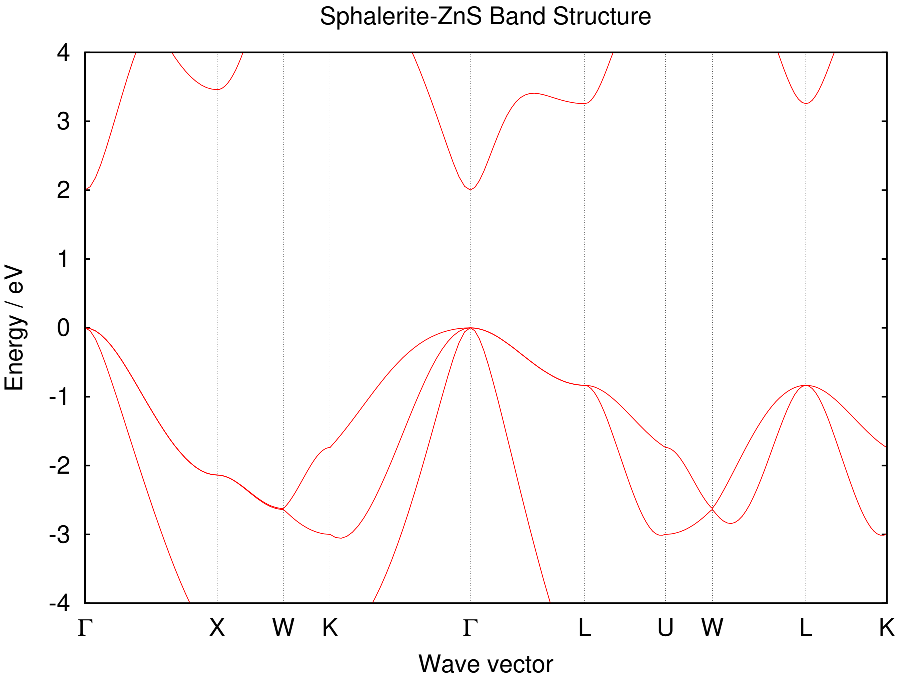

Figure 10: Electronic band structure of sphalerite-ZnS obtained with PBE functional and GBRV pseudopotentials.

5 Summary and conclusions

We have argued that free and open source software (FOSS) allows for a bring your own device (BYOD) approach to the teaching of computational chemistry, and finally affords computational chemistry for the masses, thereby also democratizing the science of computational chemistry. The distributed BYOD approach to computational chemistry also supports the delivery of massive open online courses (MOOCs), avoiding the need to organize computing resources for a large number of students in a cost-effective and secure way. We have briefly reviewed the current selection of FOSS programs for electronic structure calculations, and illustrated the installation and practical use of several programs for computational chemistry education on personal computers. As the technical barriers for running quantum chemical calculations on personal laptops have practically vanished, educators can shift their focus on content creation and developing practices for sharing and co-creating computational chemistry teaching material as Open Educational Resources.[220] The Psi4Education project[1] is one such attempt at open teaching materials; we hope such materials become more readily available and more thoroughly used in the future.

Supporting Information Available

Full input and output files for the practical examples discussed in this work, together with step-by-step instructions for installing and running the required program packages on Linux, macOS, or Windows Subsystem for Linux. The supporting information is also available as a git repository.[159]

Acknowledgments

We thank Paul Saxe and Jonathan Moussa for invaluable comments on the manuscript. A. J. K. thanks Business Finland for Co-Innovation funding (Grant No. 3767/31/2019).

References

(1) Fortenberry, R. C.; McDonald, A. R.; Shepherd, T. D.; Kennedy, M.; Sherrill, C. D. *The Promise of Chemical Education: Addressing our Students' Needs*; American Chemical Society, 2015; pp 85–98.

(2) Grushow, A.; Reeves, M. S. e. *Using Computational Methods To Teach Chemical Principles*; American Chemical Society, 2019.

(3) Esselman, B. J.; Hill, N. J. Integration of Computational Chemistry into the Undergraduate Organic Chemistry Laboratory Curriculum. *J. Chem. Educ.* **2016**, *93*, 932–936.

(4) Winfield, L. L.; McCormack, K.; Shaw, T. Using iSpartan To Support a Student-Centered Activity on Alkane Conformations. *J. Chem. Educ.* **2018**, *96*, 89–92.

(5) Esselman, B. J.; Hill, N. J. *Using Computational Methods To Teach Chemical Principles*; Chapter 11, pp 139–162.

(6) Phillips, J. A. *Using Computational Methods To Teach Chemical Principles*; Chapter 4, pp 33–50.

(7) Reeves, M. S.; Berghout, H. L.; Perri, M. J.; Singleton, S. M.; Whitnell, R. M. *Using Computational Methods To Teach Chemical Principles*; Chapter 5, pp 51–63.

(8) Martini, S. R.; Hartzell, C. J. Integrating Computational Chemistry into a Course in Classical Thermodynamics. *Journal of Chemical Education* **2015**, *92*, 1201–1203.

(9) Stocker, K. M. *Using Computational Methods To Teach Chemical Principles*; Chapter 3, pp 21–32.

(10) Snyder, H. D.; Kucukkal, T. G. Computational Chemistry Activities with Avogadro and ORCA. *J. Chem. Educ.* **2021**, *98*, 1335–1341.

(11) Hoover, G. C.; Dicks, A. P.; Seferos, D. S. Upper-Year Materials Chemistry Computational Modeling Module for Organic Display Technologies. *J. Chem. Educ.* **2021**, *98*, 805–811.

(12) Furlan, P. Y.; Bell-Loncella, E. T. Integrating Computation and Visualization to Enhance Learning IR Spectroscopy in the General Chemistry Laboratory: Computer-Assisted Learning of IR Spectroscopy. *Spectrosc. Lett.* **2010**, *43*, 618–625.

(13) Martin, W. R.; Ball, D. W. *Using Computational Methods To Teach Chemical Principles*; Chapter 8, pp 93–107.

(14) DeVore, T. C. *Using Computational Methods To Teach Chemical Principles*; Chapter 9, pp 109–125.

(15) Westmoreland, P. e. *Applying Molecular and Materials Modeling*, 1st ed.; Springer Netherlands, 2002.

(16) Head-Gordon, M.; Artacho, E. Chemistry on the computer. *Phys. Today* **2008**, *61*, 58–63.

(17) Deglmann, P.; Schäfer, A.; Lennartz, C. Application of quantum calculations in the chemical industry – An overview. *Int. J. Quantum Chem.* **2015**, *115*, 107–136.

(18) Weiss, H.; Deglmann, P.; In'T Veld, P.; Cetinkaya, M.; Schreiner, E. Multiscale Materials Modeling in an Industrial Environment. *Annu. Rev. Chem. Biomol. Eng.* **2016**, *7*, 65–86.

(19) JCE staff, Computational Chemistry for the Masses. *J. Chem. Educ.* **1996**, *73*, 104.

(20) Grushow, A.; Reeves, M. S. *Using Computational Methods To Teach Chemical Principles*; 2019; Chapter 1, pp 1–10.

(21) WebMO, A web-based interface to computational chemistry packages. https://www.webmo.net/, Accessed 8 May 2021.

(22) Polik, W. F.; Schmidt, J. R. WebMO: Web-based computational chemistry calculations in education and research. *WIREs Computational Molecular Science* e1554.

(23) Perri, M. J.; Akinmurele, M.; Haynie, M. *Using Computational Methods To Teach Chemical Principles*; Chapter 7, pp 79–92.

(24) Schwalbe, S.; Fiedler, L.; Kraus, J.; Kortus, J.; Trepte, K.; Lehtola, S. PyFLOSIC: Python-based Fermi–Löwdin orbital self-interaction correction. *J. Chem. Phys.* **2020**, *153*, 084104.

(25) Krylov, A. I.; Herbert, J. M.; Furche, F.; Head-Gordon, M.; Knowles, P. J.; Lindh, R.; Manby, F. R.; Pulay, P.; Skylaris, C.-K.; Werner, H.-J. What Is the Price of Open-Source Software? *J. Phys. Chem. Lett.* **2015**, *6*, 2751–2754.

(26) Jacob, C. R. How Open Is Commercial Scientific Software? *J. Phys. Chem. Lett.* **2016**, *7*, 351–353.

(27) Li, L. Why Should Anyone Become a Scientist? The Ideal of Science and Its Importance. *J. Chem. Educ.* **1999**, *76*, 20.

(28) Azoulay, P.; Fons-Rosen, C.; Zivin, J. S. G. Does Science Advance One Funeral at a Time? *Am. Econ. Rev.* **2019**, *109*, 2889–2920.

(29) Giles, J. Software company bans competitive users. *Nature* **2004**, *429*, 231–231.

(30) Smart, A. G. The war over supercooled water. Physics Today, 2018.

(31) Palmer, J. C.; Haji-Akbari, A.; Singh, R. S.; Martelli, F.; Car, R.; Panagiotopoulos, A. Z.; Debenedetti, P. G. Comment on "The putative liquid-liquid transition is a liquid-solid transition in atomistic models of water" [I and II: J. Chem. Phys. 135, 134503 (2011); J. Chem. Phys. 138, 214504 (2013)]. *J. Chem. Phys.* **2018**, *148*, 137101.

(32) Open Source Initiative, The Open Source Definition. https://opensource.org/osd, Accessed May 13 2021.

(33) Free Software Foundation, What is free software? https://www.gnu.org/philosophy/free-sw.html.en, Accessed May 13 2021.

(34) Stahl, M. T. Open-source software: not quite endsville. *Drug Discovery Today* **2005**, *10*, 219–222.

(35) Gezelter, J. D. Open Source and Open Data Should Be Standard Practices. *J. Phys. Chem. Lett.* **2015**, *6*, 1168–1169.

(36) Hinsen, K. Computational science: shifting the focus from tools to models. *F1000Research* **2014**, *3*, 101.

(37) Git community, Git, a free and open source distributed version control system. https://git-scm.com/, Accessed 20 May 2021.

(38) GitHub, Inc., GitHub collaboration platform. https://github.com/, Accessed 20 May 2021.

(39) GitLab, Inc., GitLab collaboration platform. https://gitlab.com/, Accessed 20 May 2021.

(40) European Organization For Nuclear Research,; OpenAIRE, Zenodo. 2013; `https://www.zenodo.org/`.

(41) Swarts, J. Open-Source Software in the Sciences: The Challenge of User Support. *J. Bus. Tech. Commun.* **2018**, *33*, 60–90.

(42) Dalke, A. The chemfp project. *J. Cheminf.* **2019**, *11*, 76.

(43) Haff, G. *How open source ate software: Understand the open source movement and so much more*; 2018; pp 1–180.

(44) Wieber, F.; Pisanty, A.; Hocquet, A. "We were here before the Web and hype...": a brief history of and tribute to the Computational Chemistry List. *J. Cheminf.* **2018**, *10*, 67.

(45) Constant, D.; Sproull, L.; Kiesler, S. The Kindness of Strangers: The Usefulness of Electronic Weak Ties for Technical Advice. *Organ. Sci.* **1996**, *7*, 119–135.

(46) Lakhani, K. R.; von Hippel, E. How open source software works: "free" user-to-user assistance. *Res. Policy* **2003**, *32*, 923–943.

(47) Schiff, A. The Economics of Open Source Software: A Survey of the Early Literature. *Rev. Netw. Econ.* **2002**, *1*, 66–74.

(48) Myatt, D. P. Equilibrium Selection and Public-good Provision: The Development of Open-source Software. *Oxford Rev. Econ. Pol.* **2002**, *18*, 446–461.

(49) Johnson, J. P. Open Source Software: Private Provision of a Public Good. *J. Econ. Manage. Strat.* **2002**, *11*, 637–662.

(50) Mustonen, M. Copyleft—the economics of Linux and other open source software. *Inf. Econ. Policy* **2003**, *15*, 99–121.

(51) Lerner, J.; Tirole, J. Some Simple Economics of Open Source. *The Journal of Industrial Economics* **2003**, *50*, 197–234.

(52) Bonaccorsi, A.; Rossi, C. Why Open Source software can succeed. *Res. Policy* **2003**, *32*, 1243–1258.

(53) Hawkins, R. E. The economics of open source software for a competitive firm. *NETNOMICS: Economic Research and Electronic Networking* **2004**, *6*, 103–117.

(54) Bitzer, J. Commercial versus open source software: the role of product heterogeneity in competition. *Economic Systems* **2004**, *28*, 369–381.

(55) Lerner, J.; Tirole, J. The Economics of Technology Sharing: Open Source and Beyond. *J. Econ. Perspect.* **2005**, *19*, 99–120.

(56) Lerner, J. The Scope of Open Source Licensing. *Journal of Law, Economics, and Organization* **2005**, *21*, 20–56.

(57) Bitzer, J.; Schröder, P. J. H. Bug-fixing and code-writing: The private provision of open source software. *Inf. Econ. Policy* **2005**, *17*, 389–406.

(58) West, J.; Gallagher, S. Challenges of open innovation: the paradox of firm investment in open-source software. *R and D Management* **2006**, *36*, 319–331.

(59) Rossi, M. A. *The Economics of Open Source Software Development*; Elsevier, 2006; pp 15–55.

(60) Gaudeul, A. Do Open Source Developers Respond to Competition? The LATEX Case Study. *Rev. Netw. Econ.* **2007**, *6*, 239–263.

(61) von Krogh, G.; von Hippel, E. The Promise of Research on Open Source Software. *Manage. Sci.* **2006**, *52*, 975–983.

(62) Hars, A.; Ou, S. Working for Free? Motivations for Participating in Open-Source Projects. *Int. J. Electron. Comm.* **2002**, *6*, 25–39.

(63) Bitzer, J.; Schrettl, W.; Schröder, P. J. H. Intrinsic motivation in open source software development. *J. Comp. Econ.* **2007**, *35*, 160–169.

(64) Lerner, J.; Pathak, P. A.; Tirole, J. The Dynamics of Open-Source Contributors. *Am. Econ. Rev.* **2006**, *96*, 114–118.

(65) Fershtman, C.; Gandal, N. Open source software: Motivation and restrictive licensing. *International Economics and Economic Policy* **2007**, *4*, 209–225.

(66) Johnson, J. P. Collaboration, peer review and open source software. *Inf. Econ. Policy* **2006**, *18*, 477–497.

(67) Bitzer, J.; Schröder, P. J. H. *The Economics of Open Source Software Development*; Elsevier, 2006; pp 219–246.

(68) top500.org, TOP500 Operating System Statistics. https://www.top500.org/statistics/details/osfam/1/, Accessed 6 July 2021.

(69) Moore, J. F.; McCann, M. P. Linux and the Chemist. *J. Chem. Educ.* **2003**, *80*, 219.

(70) Lehtola, J.; Hakala, M.; Sakko, A.; Hämäläinen, K. ERKALE – A flexible program package for X-ray properties of atoms and molecules. *J. Comput. Chem.* **2012**, *33*, 1572–1585.

(71) Smith, D. G. A. et al. Psi4 1.4: Open-source software for high-throughput quantum chemistry. *J. Chem. Phys.* **2020**, *152*, 184108.

(72) Crawford, T. D.; Sherrill, C. D.; Valeev, E. F.; Fermann, J. T.; King, R. A.; Leininger, M. L.; Brown, S. T.; Janssen, C. L.; Seidl, E. T.; Kenny, J. P.; Allen, W. D. PSI3: An open-source Ab Initio electronic structure package. *J. Comput. Chem.* **2007**, *28*, 1610–1616.

(73) Sun, Q. et al. Recent developments in the PySCF program package. *J. Chem. Phys.* **2020**, *153*, 024109.

(74) Aquilante, F. et al. Modern quantum chemistry with [Open]Molcas. *J. Chem. Phys.* **2020**, *152*, 214117.

(75) Olsen, J. M. H. et al. Dalton Project: A Python platform for molecular- and electronic-structure simulations of complex systems. *J. Chem. Phys.* **2020**, *152*, 214115.

(76) Aprà, E. et al. NWChem: Past, present, and future. *J. Chem. Phys.* **2020**, *152*, 184102.

(77) Lehtola, S.; Steigemann, C.; Oliveira, M. J. T.; Marques, M. A. L. Recent developments in LIBXC — A comprehensive library of functionals for density functional theory. *SoftwareX* **2018**, *7*, 1–5.

(78) Perdew, J. P.; Burke, K.; Ernzerhof, M. Generalized Gradient Approximation Made Simple. *Phys. Rev. Lett.* **1996**, *77*, 3865–3868.

(79) Stephens, P. J.; Devlin, F. J.; Chabalowski, C. F.; Frisch, M. J. Ab Initio Calculation of Vibrational Absorption and Circular Dichroism Spectra Using Density Functional Force Fields. *J. Phys. Chem.* **1994**, *98*, 11623–11627.

(80) Sun, J.; Ruzsinszky, A.; Perdew, J. Strongly Constrained and Appropriately Normed Semilocal Density Functional. *Phys. Rev. Lett.* **2015**, *115*, 036402.

(81) Giannozzi, P.; Baseggio, O.; Bonfà, P.; Brunato, D.; Car, R.; Carnimeo, I.; Cavazzoni, C.; de Gironcoli, S.; Delugas, P.; Ferrari Ruffino, F.; Ferretti, A.; Marzari, N.; Timrov, I.; Urru, A.; Baroni, S. QUANTUM ESPRESSO toward the exascale. *J. Chem. Phys.* **2020**, *152*, 154105.

(82) Lehtola, S. Fully numerical Hartree–Fock and density functional calculations. II. Diatomic molecules. *Int. J. Quantum Chem.* **2019**, *119*, e25944.

(83) Lehtola, S. Fully numerical Hartree–Fock and density functional calculations. I. Atoms. *Int. J. Quantum Chem.* **2019**, *119*, e25945.

(84) Lehtola, S.; Dimitrova, M.; Sundholm, D. Fully numerical electronic structure calculations on diatomic molecules in weak to strong magnetic fields. *Mol. Phys.* **2020**, *118*, e1597989.

(85) Lehtola, S. Fully numerical calculations on atoms with fractional occupations and range-separated exchange functionals. *Phys. Rev. A* **2020**, *101*, 012516.

(86) Motamarri, P.; Das, S.; Rudraraju, S.; Ghosh, K.; Davydov, D.; Gavini, V. DFT-FE – A massively parallel adaptive finite-element code for large-scale density functional theory calculations. *Comput. Phys. Commun.* **2020**, *246*, 106853.

(87) MADNESS – Multiresolution Adaptive Numerical Environment for Scientific Simulation. `https://github.com/m-a-d-n-e-s-s/madness`.

(88) te Velde, G.; Bickelhaupt, F. M.; Baerends, E. J.; Fonseca Guerra, C.; van Gisbergen, S. J. A.; Snijders, J. G.; Ziegler, T. Chemistry with ADF. *J. Comput. Chem.* **2001**, *22*, 931–967.

(89) Barca, G. M. J. et al. Recent developments in the general atomic and molecular electronic structure system. *J. Chem. Phys.* **2020**, *152*, 154102.

(90) Werner, H.-J. et al. The Molpro quantum chemistry package. *J. Chem. Phys.* **2020**, *152*, 144107.

(91) Neese, F.; Wennmohs, F.; Becker, U.; Riplinger, C. The ORCA quantum chemistry program package. *J. Chem. Phys.* **2020**, *152*, 224108.

(92) Lejaeghere, K. et al. Reproducibility in density functional theory calculations of solids. *Science* **2016**, *351*, aad3000–aad3000.

(93) Ajila, S. A.; Wu, D. Empirical study of the effects of open source adoption on software development economics. *J. Syst. Software* **2007**, *80*, 1517–1529.

(94) DeLano, W. L. The case for open-source software in drug discovery. *Drug Discovery Today* **2005**, *10*, 213–217.

(95) Lehtola, S.; Blockhuys, F.; Van Alsenoy, C. An Overview of Self-Consistent Field Calculations Within Finite Basis Sets. *Molecules* **2020**, *25*, 1218.

(96) Smith, D. G. A. et al. PSI4NumPy: An Interactive Quantum Chemistry Programming Environment for Reference Implementations and Rapid Development. *J. Chem. Theory Comput.* **2018**, *14*, 3504–3511.

(97) Samsonidze, G.; Kozinsky, B. Half-Heusler Compounds For Use In Thermoelectric Generators. U.S. Patent 20170141282, May 2017.

(98) Ye, J.-H.; Huang, C.-L. Method for crystallizing metal oxide semiconductor layer, semiconductor structure, active array substrate, and indium gallium zinc oxide crystal. U.S. Patent 20180166474, June 2018.

(99) Strohmaier, E.; Meuer, H. W.; Dongarra, J.; Simon, H. D. The TOP500 List and Progress in High-Performance Computing. *Computer* **2015**, *48*, 42–49.

(100) Meuer, H. W.; Strohmaier, E.; Dongarra, J.; Simon, H.; Meuer, M. TOP500. https://top500.org/, Accessed Thu May 20 2021.

(101) Szabo, A.; Ostlund, N. S. *Modern quantum chemistry: introduction to advanced electronic structure theory*; Dover Pubns, 1996.

(102) Geldenhuys, W. J.; Gaasch, K. E.; Watson, M.; Allen, D. D.; der Schyf, C. J. V. Optimizing the use of open-source software applications in drug discovery. *Drug Discovery Today* **2006**, *11*, 127–132.

(103) Pirhadi, S.; Sunseri, J.; Koes, D. R. Open source molecular modeling. *J. Mol. Graph. Model.* **2016**, *69*, 127–143.

(104) Rodríguez-Becerra, J.; Cáceres-Jensen, L.; Díaz, T.; Druker, S.; Padilla, V. B.; Pernaa, J.; Aksela, M. Developing technological pedagogical science knowledge through educational computational chemistry: a case study of pre-service chemistry teachers' perceptions. *Chem. Educ. Res. Pract.* **2020**, *21*, 638–654.

(105) Python Package Index – PyPI. https://pypi.org/, Accessed 7 July 2021.

(106) Continuum Analytics, Conda package manager. https://conda.io/, Accessed 26 May 2021.

(107) Hohenberg, P.; Kohn, W. Inhomogeneous Electron Gas. *Phys. Rev.* **1964**, *136*, B864–B871.

(108) Kohn, W.; Sham, L. J. Self-Consistent Equations Including Exchange and Correlation Effects. *Phys. Rev.* **1965**, *140*, A1133–A1138.

(109) Shiozaki, T. BAGEL: Brilliantly Advanced General Electronic-structure Library. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **2018**, *8*, e1331.

(110) Williams-Young, D. B.; Petrone, A.; Sun, S.; Stetina, T. F.; Lestrange, P.; Hoyer, C. E.; Nascimento, D. R.; Koulias, L.; Wildman, A.; Kasper, J.; Goings, J. J.; Ding, F.; DePrince, A. E.; Valeev, E. F.; Li, X. The Chronus Quantum software package. *WIREs Comput. Mol. Sci.* **2020**, *10*, e1436.

(111) Aidas, K. et al. The Dalton quantum chemistry program system. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **2014**, *4*, 269–284.

(112) Rudberg, E.; Rubensson, E. H.; Sałek, P.; Kruchinina, A. Ergo: An open-source program for linear-scaling electronic structure calculations. *SoftwareX* **2018**, *7*, 107–111.

(113) Folkestad, S. D. et al. e^T 1.0: An open source electronic structure program with emphasis on coupled cluster and multilevel methods. *J. Chem. Phys.* **2020**, *152*, 184103.

(114) Poole, D.; Vallejo, J. L. G.; Gordon, M. S. A New Kid on the Block: Application of Julia to Hartree–Fock Calculations. *J. Chem. Theory Comput.* **2020**, *16*, 5006–5013.

(115) Bruneval, F.; Rangel, T.; Hamed, S. M.; Shao, M.; Yang, C.; Neaton, J. B. molgw 1: Many-body perturbation theory software for atoms, molecules, and clusters. *Comput. Phys. Commun.* **2016**, *208*, 149–161.

(116) Peng, C.; Lewis, C. A.; Wang, X.; Clement, M. C.; Pierce, K.; Rishi, V.; Pavošević, F.; Slattery, S.; Zhang, J.; Teke, N.; Kumar, A.; Masteran, C.; Asadchev, A.; Calvin, J. A.; Valeev, E. F. Massively Parallel Quantum Chemistry: A high-performance research platform for electronic structure. *J. Chem. Phys.* **2020**, *153*, 044120.

(117) Mueller, R. P. PyQuante: Python Quantum Chemistry. http://pyquante.sourceforge.net/, Accessed 6 July 2021.

(118) Unsleber, J. P.; Dresselhaus, T.; Klahr, K.; Schnieders, D.; Böckers, M.; Barton, D.; Neugebauer, J. Serenity: A subsystem quantum chemistry program. *J. Comput. Chem.* **2018**, *39*, 788–798.

(119) Kjellgren, E. SlowQuant. https://github.com/erikkjellgren/SlowQuant, Accessed 6 July 2021.

(120) Rinkevicius, Z.; Li, X.; Vahtras, O.; Ahmadzadeh, K.; Brand, M.; Ringholm, M.; List, N. H.; Scheurer, M.; Scott, M.; Dreuw, A.; Norman, P. VeloxChem: A Python-driven density-functional theory program for spectroscopy simulations in high-performance computing environments. *WIREs Comput. Mol. Sci.* **2019**, *10*, e1457.

(121) Souvatzis, P. Uquantchem: A versatile and easy to use quantum chemistry computational software. *Computer Physics Communications* **2014**, *185*, 415–421.

(122) Romero, A. H. et al. ABINIT: Overview and focus on selected capabilities. *J. Chem. Phys.* **2020**, *152*, 124102.

(123) Nakata, A.; Baker, J. S.; Mujahed, S. Y.; Poulton, J. T. L.; Arapan, S.; Lin, J.; Raza, Z.; Yadav, S.; Truflandier, L.; Miyazaki, T.; Bowler, D. R. Large scale and linear scaling DFT with the CONQUEST code. *J. Chem. Phys.* **2020**, *152*, 164112.

(124) Kühne, T. D. et al. CP2K: An electronic structure and molecular dynamics software package - Quickstep: Efficient and accurate electronic structure calculations. *J. Chem. Phys.* **2020**, *152*, 194103.

(125) Herbst, M. F.; Levitt, A.; Cancès, E. DFTK: A Julian approach for simulating electrons in solids. *JuliaCon Proceedings* **2021**, *3*, 69.

(126) The Elk Code. `http://elk.sourceforge.net/`.

(127) Gulans, A.; Kontur, S.; Meisenbichler, C.; Nabok, D.; Pavone, P.; Rigamonti, S.; Sagmeister, S.; Werner, U.; Draxl, C. exciting: a full-potential all-electron package implementing density-functional theory and many-body perturbation theory. *J. Phys. Condens. Matter* **2014**, *26*, 363202.

(128) FLEUR. `http://www.flapw.de`.

(129) Enkovaara, J. et al. Electronic structure calculations with GPAW: a real-space implementation of the projector augmented-wave method. *J. Phys. Condens. Matter* **2010**, *22*, 253202.

(130) Sundararaman, R.; Letchworth-Weaver, K.; Schwarz, K. A.; Gunceler, D.; Ozhabes, Y.; Arias, T. A. JDFTx: Software for joint density-functional theory. *SoftwareX* **2017**, *6*, 278–284.

(131) Xu, Q.; Sharma, A.; Suryanarayana, P. M-SPARC: Matlab-Simulation Package for Ab-initio Real-space Calculations. *SoftwareX* **2020**, *11*, 100423.

(132) Ozaki, T.; Kino, H. Numerical atomic basis orbitals from H to Kr. *Phys. Rev. B* **2004**, *69*, 195113.

(133) Fathurrahman, F.; Agusta, M. K.; Saputro, A. G.; Dipojono, H. K. PWDFT.jl: A Julia package for electronic structure calculation using density functional theory and plane wave basis. *Comput. Phys. Commun.* **2020**, *256*, 107372.

(134) García, A. et al. SIESTA: Recent developments and applications. *J. Chem. Phys.* **2020**, *152*, 204108.

(135) Gygi, F. Architecture of Qbox: A scalable first-principles molecular dynamics code. *IBM J. Res. Dev.* **2008**, *52*, 137–144.

(136) Xu, Q.; Sharma, A.; Comer, B.; Huang, H.; Chow, E.; Medford, A. J.; Pask, J. E.; Suryanarayana, P. SPARC: Simulation Package for Ab-initio Real-space Calculations. *SoftwareX* **2021**, *15*, 100709.

(137) Kang, S.; Woo, J.; Kim, J.; Kim, H.; Kim, Y.; Lim, J.; Choi, S.; Kim, W. Y. ACE-Molecule: An open-source real-space quantum chemistry package. *J. Chem. Phys.* **2020**, *152*, 124110.

(138) Ratcliff, L. E.; Dawson, W.; Fisicaro, G.; Caliste, D.; Mohr, S.; Degomme, A.; Videau, B.; Cristiglio, V.; Stella, M.; D'Alessandro, M.; Goedecker, S.; Nakajima, T.; Deutsch, T.; Genovese, L. Flexibilities of wavelets as a computational basis set for large-scale electronic structure calculations. *J. Chem. Phys.* **2020**, *152*, 194110.

(139) Harrison, R. J. et al. MADNESS: A Multiresolution, Adaptive Numerical Environment for Scientific Simulation. *SIAM J. Sci. Comput.* **2016**, *38*, S123–S142.

(140) Jensen, S. R.; Flå, T.; Jonsson, D.; Monstad, R. S.; Ruud, K.; Frediani, L. Magnetic properties with multiwavelets and DFT: the complete basis set limit achieved. *Phys. Chem. Chem. Phys.* **2016**, *18*, 21145–21161.

(141) Tancogne-Dejean, N. et al. Octopus, a computational framework for exploring light-driven phenomena and quantum dynamics in extended and finite systems. *J. Chem. Phys.* **2020**, *152*, 124119.

(142) Saad, Y.; Chelikowsky, J. R.; Shontz, S. M. Numerical Methods for Electronic Structure Calculations of Materials. *SIAM Rev.* **2010**, *52*, 3–54.

(143) Briggs, E. L.; Sullivan, D. J.; Bernholc, J. Real-space multigrid-based approach to large-scale electronic structure calculations. *Phys. Rev. B* **1996**, *54*, 14362–14375.

(144) Kobus, J. A finite difference Hartree–Fock program for atoms and diatomic molecules. *Comput. Phys. Commun.* **2013**, *184*, 799–811.

(145) Hourahine, B. et al. DFTB+, a software package for efficient approximate density functional theory based atomistic simulations. *J. Chem. Phys.* **2020**, *152*, 124101.

(146) Bock, N.; Cawkwell, M. J.; Coe, J. D.; Krishnapriyan, A.; Kroonblawd, M. P.; Lang, A.; ; Liu, C.; Saez, E. M.; Mniszewski, S. M.; Negre, C. F. A.; Niklasson, A. M. N.; Sanville, E.; Wood, M. A.; Yang, P. LATTE. https://github.com/lanl/LATTE, 2008; `https://github.com/lanl/LATTE`, Accessed 12 July 2021.

(147) Husch, T.; Reiher, M. Comprehensive Analysis of the Neglect of Diatomic Differential Overlap Approximation. *J. Chem. Theory Comput.* **2018**, *14*, 5169–5179.

(148) Bannwarth, C.; Caldeweyher, E.; Ehlert, S.; Hansen, A.; Pracht, P.; Seibert, J.; Spicher, S.; Grimme, S. Extended tight-binding quantum chemistry methods. *WIREs Comput. Mol. Sci.* **2021**, *11*, e1493.

(149) Jmol: an open-source Java viewer for chemical structures in 3D. http://www.jmol.org.

(150) Hanwell, M.; Curtis, D.; Lonie, D.; Vandermeerschd, T.; Zurek, E.; Hutchison, G. Avogadro: An advanced semantic chemical editor, visualization, and analysis platform. *Journal of Cheminformatics* **2012**, *4*.

(151) Gilbert, A. IQmol, a free open-source molecular editor and visualization package. http://iqmol.org, Accessed June 26 2021.

(152) Schrödinger Inc., PyMol, a molecular visualization system. https://github.com/schrodinger/pymol-open-source, Accessed 6 July 2021.

(153) Larsen, A. H. et al. The atomic simulation environment—a Python library for working with atoms. *J. Phys.: Condens. Matter* **2017**, *29*, 273002.

(154) Kim, S.; Chen, J.; Cheng, T.; Gindulyte, A.; He, J.; He, S.; Li, Q.; Shoemaker, B. A.; Thiessen, P. A.; Yu, B.; Zaslavsky, L.; Zhang, J.; Bolton, E. E. PubChem in 2021: new data content and improved web interfaces. *Nucleic Acids Research* **2020**, *49*, D1388–D1395.

(155) O'Boyle, N. M.; Banck, M.; James, C. A.; Morley, C.; Vandermeersch, T.; Hutchison, G. R. Open Babel: An open chemical toolbox. *J. Cheminf.* **2011**, *3*, 33.

(156) O'Boyle, N. M.; Tenderholt, A. L.; Langner, K. M. cclib: A library for package-independent computational chemistry algorithms. *J. Comput. Chem.* **2008**, *29*, 839–845.

(157) Lu, T.; Chen, F. Multiwfn: A multifunctional wavefunction analyzer. *J. Comput. Chem.* **2011**, *33*, 580–592.

(158) Hermann, G.; Pohl, V.; Tremblay, J. C.; Paulus, B.; Hege, H.-C.; Schild, A. ORBKIT: A modular python toolbox for cross-platform postprocessing of quantum chemical wavefunction data. *J. Comput. Chem.* **2016**, *37*, 1511–1520.

(159) Lehtola, S.; Karttunen, A. J. git repository containing a copy of the supporting information. https://github.com/susilehtola/fosschemistry, Accessed 8 August 2021.

(160) Bannwarth, C.; Ehlert, S.; Grimme, S. GFN2-xTB—An Accurate and Broadly Parametrized Self-Consistent Tight-Binding Quantum Chemical Method with Multipole Electrostatics and Density-Dependent Dispersion Contributions. *J. Chem. Theory Comput.* **2019**, *15*, 1652–1671.

(161) Menzel, J. P.; Kloppenburg, M.; Belić, J.; Groot, H. J. M.; Visscher, L.; Buda, F. Efficient workflow for the investigation of the catalytic cycle of water oxidation catalysts: Combining GFN-xTB and density functional theory. *J. Comput. Chem.* **2021**,

(162) Tasinato, N.; Puzzarini, C.; Barone, V. Correct Modeling of Cisplatin: a Paradigmatic Case. *Angew. Chem. - Int. Ed.* **2017**, *56*, 13838–13841.

(163) Liu, W.; Franke, R. Comprehensive relativistic ab initio and density functional theory studies on PtH, PtF, PtCl, and $Pt(NH_3)_2Cl_2$. *J. Comput. Chem.* **2002**, *23*, 564–575.

(164) Becke, A. D. Density-functional exchange-energy approximation with correct asymptotic behavior. *Phys. Rev. A* **1988**, *38*, 3098–3100.

(165) Perdew, J. P. Density-functional approximation for the correlation energy of the inhomogeneous electron gas. *Phys. Rev. B* **1986**, *33*, 8822–8824.

(166) Adamo, C.; Barone, V. Toward reliable density functional methods without adjustable parameters: The PBE0 model. *J. Chem. Phys.* **1999**, *110*, 6158–6170.

(167) Ernzerhof, M.; Scuseria, G. E. Assessment of the Perdew–Burke–Ernzerhof exchange-correlation functional. *J. Chem. Phys.* **1999**, *110*, 5029–5036.

(168) Vetere, V.; Adamo, C.; Maldivi, P. Performance of the 'parameter free' PBE0 functional for the modeling of molecular properties of heavy metals. *Chem. Phys. Lett.* **2000**, *325*, 99–105.

(169) Bühl, M.; Reimann, C.; Pantazis, D. A.; Bredow, T.; Neese, F. Geometries of Third-Row Transition-Metal Complexes from Density-Functional Theory. *J. Chem. Theory Comput.* **2008**, *4*, 1449–1459.

(170) Davidson, E. R.; Feller, D. Basis set selection for molecular calculations. *Chem. Rev.* **1986**, *86*, 681–696.

(171) Pritchard, B. P.; Altarawy, D.; Didier, B.; Gibson, T. D.; Windus, T. L. New Basis Set Exchange: An Open, Up-to-Date Resource for the Molecular Sciences Community. *J. Chem. Inf. Model.* **2019**, *59*, 4814–4820.

(172) Hehre, W. J.; Stewart, R. F.; Pople, J. A. Self-Consistent Molecular-Orbital Methods. I. Use of Gaussian Expansions of Slater-Type Atomic Orbitals. *J. Chem. Phys.* **1969**, *51*, 2657.

(173) Binkley, J. S.; Pople, J. A.; Hehre, W. J. Self-consistent molecular orbital methods. 21. Small split-valence basis sets for first-row elements. *J. Am. Chem. Soc.* **1980**, *102*, 939.

(174) Hehre, W. J.; Ditchfield, R.; Pople, J. A. Self-Consistent Molecular Orbital Methods. XII. Further Extensions of Gaussian-Type Basis Sets for Use in Molecular Orbital Studies of Organic Molecules. *J. Chem. Phys.* **1972**, *56*, 2257–2261.

(175) Hill, J. G. Gaussian basis sets for molecular applications. *Int. J. Quantum Chem.* **2013**, *113*, 21–34.

(176) Jensen, F. Atomic orbital basis sets. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **2013**, *3*, 273–295.

(177) Lehtola, S. A review on non-relativistic, fully numerical electronic structure calculations on atoms and diatomic molecules. *Int. J. Quantum Chem.* **2019**, *119*, e25968.

(178) Weigend, F.; Ahlrichs, R. Balanced basis sets of split valence, triple zeta valence and quadruple zeta valence quality for H to Rn: Design and assessment of accuracy. *Phys. Chem. Chem. Phys.* **2005**, *7*, 3297–305.

(179) Pyykkö, P. Relativistic effects in chemistry: more common than you thought. *Annu. Rev. Phys. Chem.* **2012**, *63*, 45–64.

(180) Pyykkö, P. The physics behind chemistry and the periodic table. *Chem. Rev.* **2012**, *112*, 371–84.

(181) Dolg, M. *Theoretical and Computational Chemistry*; Elsevier, 2002; pp 793–862.

(182) Rappoport, D.; Furche, F. Property-optimized gaussian basis sets for molecular response calculations. *J. Chem. Phys.* **2010**, *133*, 134105.

(183) Ehlert, S.; Huniar, U.; Ning, J.; Furness, J. W.; Sun, J.; Kaplan, A. D.; Perdew, J. P.; Brandenburg, J. G. r2SCAN-D4: Dispersion corrected meta-generalized gradient approximation for general chemical applications. *J. Chem. Phys.* **2021**, *154*, 061101.

(184) Tasinato, N.; Puzzarini, C.; Barone, V. Correct Modeling of Cisplatin: a Paradigmatic Case. *Angewandte Chemie International Edition* **2017**, *56*, 13838–13841.

(185) Møller, C.; Plesset, M. S. M. Note on an Approximation Treatment for Many-Electron Systems. *Phys. Rev.* **1934**, *46*, 618–622.

(186) Čížek, J. On the Correlation Problem in Atomic and Molecular Systems. Calculation of Wavefunction Components in Ursell-Type Expansion Using Quantum-Field Theoretical Methods. *J. Chem. Phys.* **1966**, *45*, 4256–4266.

(187) Grimme, S. Exploration of Chemical Compound, Conformer, and Reaction Space with Meta-Dynamics Simulations Based on Tight-Binding Quantum Chemical Calculations. *J. Chem. Theory Comput.* **2019**, *15*, 2847–2862.

(188) Pracht, P.; Bohle, F.; Grimme, S. Automated exploration of the low-energy chemical space with fast quantum chemical methods. *Phys. Chem. Chem. Phys.* **2020**, *22*, 7169–7192.

(189) Pracht, P.; Grimme, S. Calculation of absolute molecular entropies and heat capacities made simple. *Chem. Sci.* **2021**, *12*, 6551–6568.

(190) Whitten, J. L. Coulombic potential energy integrals and approximations. *J. Chem. Phys.* **1973**, *58*, 4496.

(191) Baerends, E. J.; Ellis, D. E.; Ros, P. Self-consistent molecular Hartree–Fock–Slater calculations I. The computational procedure. *Chem. Phys.* **1973**, *2*, 41–51.

(192) Dunlap, B. I.; Connolly, J. W. D.; Sabin, J. R. On the applicability of LCAO-Xα methods to molecules containing transition metal atoms: The nickel atom and nickel hydride. *Int. J. Quantum Chem.* **1977**, *12*, 81–87.

(193) Dunlap, B. I.; Connolly, J. W. D.; Sabin, J. R. On some approximations in applications of Xα theory. *J. Chem. Phys.* **1979**, *71*, 3396.

(194) Dunlap, B. I.; Rösch, N.; Trickey, S. B. Variational fitting methods for electronic structure calculations. *Mol. Phys.* **2010**, *108*, 3167–3180.

(195) Weigend, F. Hartree–Fock exchange fitting basis sets for H to Rn. *J. Comput. Chem.* **2008**, *29*, 167–175.

(196) Dunning, T. H. Gaussian basis sets for use in correlated molecular calculations. I. The atoms boron through neon and hydrogen. *J. Chem. Phys.* **1989**, *90*, 1007.

(197) Van Lenthe, E.; Baerends, E. J. Optimized Slater-type basis sets for the elements 1-118. *J. Comput. Chem.* **2003**, *24*, 1142–56.

(198) Chong, D. P.; van Lenthe, E.; Van Gisbergen, S.; Baerends, E. J. Even-tempered Slater-type orbitals revisited: from hydrogen to krypton. *J. Comput. Chem.* **2004**, *25*, 1030–6.

(199) Bühl, M.; Kabrede, H. Geometries of Transition-Metal Complexes from Density-Functional Theory. *J. Chem. Theory Comput.* **2006**, *2*, 1282–1290.

(200) Grimme, S.; Bannwarth, C.; Shushkov, P. A Robust and Accurate Tight-Binding Quantum Chemical Method for Structures, Vibrational Frequencies, and Noncovalent Interactions of Large Molecular Systems Parametrized for All spd-Block Elements (Z = 1–86). *J. Chem. Theory Comput.* **2017**, *13*, 1989–2009.

(201) Bloch, F. Bemerkung zur Elektronentheorie des Ferromagnetismus und der elektrischen Leitfähigkeit. *Z. Phys.* **1929**, *57*, 545–555.

(202) Dirac, P. A. M. Note on Exchange Phenomena in the Thomas Atom. *Math. Proc. Cambridge Philos. Soc.* **1930**, *26*, 376–385.

(203) Perdew, J. P.; Wang, Y. Accurate and simple analytic representation of the electron-gas correlation energy. *Phys. Rev. B* **1992**, *45*, 13244–13249.

(204) Furness, J. W.; Kaplan, A. D.; Ning, J.; Perdew, J. P.; Sun, J. Accurate and Numerically Efficient r^{2}SCAN Meta-Generalized Gradient Approximation. *J. Phys. Chem. Lett.* **2020**, *11*, 8208–8215.

(205) Furness, J. W.; Kaplan, A. D.; Ning, J.; Perdew, J. P.; Sun, J. Correction to "Accurate and Numerically Efficient r^{2}SCAN Meta-Generalized Gradient Approximation". *J. Phys. Chem. Lett.* **2020**, *11*, 9248–9248.

(206) Weigend, F. Accurate Coulomb-fitting basis sets for H to Rn. *Phys. Chem. Chem. Phys.* **2006**, *8*, 1057–65.

(207) Pietro, W. J.; Hehre, W. J. Molecular orbital theory of the properties of inorganic and organometallic compounds. 3.STO-3G basis sets for first- and second-row transition metals. *J. Comput. Chem.* **1983**, *4*, 241–251.

(208) French, R. J.; Hedberg, L.; Hedberg, K.; Gard, G. L.; Johnson, B. M. Molecular structure and quadratic force field of chromyl fluoride, CrO_2F_2. *Inorg. Chem.* **1983**, *22*, 892–895.

(209) Levason, W. L.; Ogden, J. S.; Saad, A. K.; Young, N. A.; Brisdon, A. K.; Holliman, P. J.; Holloway, J. H.; Hope, E. G. Metal *K*-edge EXAFS (extended x-ray absorption fine structure) studies of CrO_2F_2 and MnO_3F at 10K. *J. Fluorine Chem.* **1991**, *53*, 43–51.

(210) Gilbert, B.; Frazer, B. H.; Zhang, H.; Huang, F.; Banfield, J. F.; Haskel, D.; Lang, J. C.; Srajer, G.; Stasio, G. D. X-ray absorption spectroscopy of the cubic and hexagonal polytypes of zinc sulfide. *Phys. Rev. B* **2002**, *66*, 245205.

(211) Gardner, P. J.; Pang, P. Thermodynamics of the zinc sulphide transformation, sphalerite → wurtzite, by modified entrainment. *J. Chem. Soc., Faraday Trans. 1* **1988**, *84*, 1879.

(212) Garrity, K. F.; Bennett, J. W.; Rabe, K. M.; Vanderbilt, D. Pseudopotentials for high-throughput DFT calculations. *Comput. Mater. Sci.* **2014**, *81*, 446–452.

(213) Team, C. Crystallography Open Database. http://www.crystallography.net/cod/, Accessed 20 July 2021.

(214) Materials Cloud, Quantum ESPRESSO input generator and structure visual-

izer. https://www.materialscloud.org/work/tools/qeinputgenerator, Accessed 20 July 2021.

(215) Cardona, M.; Kremer, R. K.; Lauck, R.; Siegle, G.; Muñoz, A.; Romero, A. H.; Schindler, A. Electronic, vibrational, and thermodynamic properties of ZnS with zincblende and rocksalt structure. *Phys. Rev. B* **2010**, *81*, 075207.

(216) Materials Cloud, SeeK-path: the k-path finder and visualizer. https://www.materialscloud.org/work/tools/seekpath, Accessed 20 July 2021.

(217) Setyawan, W.; Curtarolo, S. High-throughput electronic band structure calculations: Challenges and tools. *Comput. Mater. Sci.* **2010**, *49*, 299–312.

(218) Jain, A.; Ong, S. P.; Hautier, G.; Chen, W.; Richards, W. D.; Dacek, S.; Cholia, S.; Gunter, D.; Skinner, D.; Ceder, G.; Persson, K. A. Commentary: The Materials Project: A materials genome approach to accelerating materials innovation. *APL Mater.* **2013**, *1*, 011002.

(219) Tran, T. K.; Park, W.; Tong, W.; Kyi, M. M.; Wagner, B. K.; Summers, C. J. Photoluminescence properties of ZnS epilayers. *J. Appl. Phys.* **1997**, *81*, 2803–2809.

(220) McDonald, A. R.; Nash, J. A.; Nerenberg, P. S.; Ball, K. A.; Sode, O.; Foley, J. J.; Windus, T. L.; Crawford, T. D. Building capacity for undergraduate education and training in computational molecular science: A collaboration between the MERCURY consortium and the Molecular Sciences Software Institute. *Int. J. Quantum Chem.* **2020**, *120*, e26359.