Free and Open Source Software for Computational Chemistry Education

Susi Lehtola[1, a)] and Antti J. Karttunen[2]
[1)] *Molecular Sciences Software Institute, Blacksburg, Virginia 24061, United States*
[2)] *Department of Chemistry and Materials Science, Aalto University, Espoo, Finland*

Long in the making, computational chemistry for the masses [J. Chem. Educ. 1996, 73, 104] is finally here. Our brief review on free and open source software (FOSS) packages points out the existence of software o ering 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, covering both molecular and solid-state systems. Combined with the remarkable increase in the computing power of personal devices, which now rivals that of the fastest supercomputers in the world in the 1990s, we demonstrate that a decentralized model for teaching computational chemistry is now possible thanks to FOSS packages, enabling students to perform reasonable modeling on their own computing devices in the bring your own device (BYOD) scheme. FOSS software can be made trivially simple to install and keep up to date, eliminating the need for departmental support, and also enables comprehensive teaching strategies, as various algorithms' actual implementations can be used in teaching. We exemplify what kinds of calculations are feasible with four FOSS electronic structure programs, assuming only extremely modest computational resources, to illustrate how FOSS packages enable decentralized approaches to computational chemistry education within the BYOD scheme. FOSS also has further benefits driving its adoption: the open access to the source code of FOSS packages democratizes the science of computational chemistry, and FOSS packages can be used without limitation also beyond education, in academic and industrial applications, for example.

CONTENTS

I. INTRODUCTION

Quantum chemical research methods have been used extensively in the chemical industry already for several decades.[1–4] In addition to the widespread use in industry as well as in academia, quantum chemistry is also utilized in chemical education to provide atomic-level understanding of fundamental chemical concepts and phenomena.[5,6] For example, in undergraduate general and organic chemistry curricula, students get hands-on

[a)]Electronic mail: susi.lehtola@alumni.helsinki.fi experience on concepts such as three-dimensional molecular structure, structural isomerism, conformers, and stereochemistry by means of computational exercises or computer laboratory sessions.[7–9]

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 firsthand understanding on more advanced topics such as molecular orbitals, chemical bonding, energetics,[10] thermodynamics,[11,12] reaction mechanisms,[13] and various spectroscopies.[14–18]

The ability to interpret and understand chemical phenomena with the help of quantum chemical calculations is a valuable skill in every chemist's professional life: 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.

Although computational chemistry for the masses—a pervasive inclusion of computational modeling in the chemistry curriculum—has been long thought to be coming,[19] it does not appear to have arrived yet. In their recent overview, Grushow and Reeves [20] have summarized some select landmarks in computational chemistry education. At the same time, Grushow and Reeves note how computational chemistry still has a somewhat limited presence in undergraduate curriculums, which can be attributed at least in part to the history of computational chemistry software.

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. Someone also has to maintain the software on the classroom computers and ensure it is kept up to date.

In the early 2000s, the WebMO package introduced a web-based approach to computational chemistry education, in which the quantum chemistry software only needs to be installed and maintained on a central server, and the teachers and students can then access it through a web browser interface.[21,22] A number of quantum chemistry software packages have been integrated with WebMO whose integrated molecular editor and analysis tools 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 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 also begun to o er 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 continued future availability 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. (Inroads into BYOD in the context of virtual laboratories have also been recently discussed by Kobayashi et al. [24])

The layout of this work is as follows. In section II, we will begin by defining what we mean by FOSS (Subsection II A). Then, we discuss why FOSS has not been the norm in science (Subsection II B), what FOSS enables for the teaching of computational chemistry (Subsection II C), and why it would be a good time now to switch over to FOSS in teaching (Subsection II D). We present a brief overview of available FOSS packages in section III, . We include several practical demonstrations of using state-of-the-art FOSS programs for computational chemistry education in section IV, showcasing the kinds of calculations that are possible assuming only limited computer resources. The article concludes in a brief summary and discussion in section V.

II. FREE AND OPEN SOURCE SOFTWARE

A. Definitions

As our 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 developing new features or computational models, for example. 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.[25] This phenomenon has traditionally been the main enticement of contributing to closed-source or "open teamware"[26] packages, as access to their source code partly eliminates the need to start from scratch, as algorithms implemented in the package by its other contributors can be leveraged to develop new computational models.

However, the control of access to the source code of such closed-source programs lead to perpetuating power structures and may inhibit academic collaborations between authors of di erent program packages,[27] instead of the Popperian ideal of science: the selfless pursuit of truth,[28] and a fair and unbiased competition of ideas and methods in the context of computational chemistry. Key persons in control of the access to the source codes of various software packages are able to hold back equitable competition and collaboration between scientists developing new methods and algorithms. The issue with gatekeepers is not a new phenomenon: as was already quipped 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 is supported by a recent study that investigated the dynamics of scientific evolution with the standard empirical tools of applied microeconomics.[29] This problem is less likely to manifest in FOSS, as will be explained in the next paragraph.

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 one's collaborators or the reviewers of a scientific paper, for instance. 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[30] 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"[31] 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.[32] Such problems are much less likely to exist if FOSS is used, as FOSS 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[33] as well as the four essential freedoms of "free software" or "libre software" defined by the Free Software Foundation.[34] 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.[35] It is always easier to switch to a more permissive 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.

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

1. Code distribution

The ideology of FOSS is in line with the demands of science,[36] 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 continues to be funded by public research funding, the results of such 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[37], 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. But, 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 those codes, 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[38] facilitate robust development of software, which can be hosted at no cost on sites such as GitHub[39] and GitLab[40]. 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[41] with version-specific Digital Object Identifiers (DOIs). Also precompiled versions can nowadays be easily distributed, as we will discuss in section III.

2. Maintenance and user support

A commonly referred impediment to FOSS in science is that funding its maintenance and/or user support is challenging.[26,42,43] However, there are several companies whose whole business model is founded on the use, development, and support of FOSS. For instance, Red Hat Inc broke $1 billion in annual revenue in 2012, and its revenue has increased ever since, surpassing $3 billion in 2018.[44] There is clearly money to be made in selling support for FOSS. Moreover, in contrast 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; this is the key to the Red Hat style business model.

The business model also works for scientitic FOSS. For instance, Kitware Inc., established in 1998,[45] has built its business model around developing and supporting a variety of scientific FOSS. Paraview[46] and ITK[47] enable modeling, visualization and data analysis for large datasets, while the CMake build system has become a quintessential tool for building scientific software.[48] As of 2022, Kitware has more than 200 employees and their FOSS projects span many fields of science and technology, including quantum chemistry.[49]

Due to the relatively small market for specialized scientific software, the availability of public research funding has always played a key role in the development of computational chemistry software. Related to future development of FOSS in science, the European Commission has outlined Open Science as their policy priority and the standard method of working under its research and innovation funding programs.[50]

As evidenced by forums such as the Computational Chemistry List[51] and the present authors' professional experience, online peer-to-peer user support—whose motivations have been studied e.g. by Constant, Sproull, and Kiesler[52]—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.[53] 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.

Importantly, the possibility to contribute bug fixes to FOSS projects reduces the barrier between users and developers, and is the typical route how a project gains new developers. The fostering of new developers can also be greatly aided by practices such as open code review, which serves a double purpose of both ensuring a top quality code base and teaching both the new contributor as well as any other project followers about the structure and design philosophy of the project. This naturally also leads to a more sustainable development environment, since a constant influx of new developers is secured, and enables expert knowledge (also known as tacit knowledge) to be passed onto new members of the development team.

Other aspects of the economic principles of FOSS have also been studied extensively:[54–67] FOSS is a public good.[55,56,68] Participation in the development and support of FOSS has been found to be more motivating than that of proprietary software,[69,70] and participation in FOSS projects is motivating and carries economic benefits.[71,72] FOSS promotes peer review, free exchange of ideas, and maintainability,[73] and competition of FOSS packages promotes innovation.[74]

3. Linux distributions

The Linux operating system is a prime example of FOSS. Originating from the University of Helsinki, Finland, 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[75] and the majority of the world's internet servers have run on Linux for a long time; Android smartphones likewise run on Linux. Because of Linux, proprietary operating systems have been ir-relevant in high-performance computing for many years. Chemists had good reasons to switch to Linux already ages ago;[76] 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 and ensures seamless compatibility even if students have dissimilar computing devices at their disposal.

Several Linux distributions, such as Ubuntu, Debian, and Fedora Linux have also solved the problem of efficient distribution of software decades ago. Our criteria for FOSS in Subsection II A 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,[77] Psi4[78] and its predecessor Psi3[79], and PySCF[80] 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,[81] Dalton,[82] and NWChem.[83]

4. *Case study: Libxc library of density functional approximations*

An example of a successful scientific FOSS project can be found in the Libxc library of density functional approximations.[84] The modular library currently implements over 600 density functional approximations such as PBE,[85] B3LYP,[86] and SCAN,[87] and is used by over 30 electronic structure programs ranging from programs using Gaussian basis sets (Erkale,[77] Psi4,[78], PySCF[80]) to plane-wave codes (ABINIT,[88] INQ,[89] Quantum Espresso[90]), finite element programs (HelFEM[91–94], DFT-FE[95]), and multiresolution adaptive grids (MADNESS[96]). In order to facilitate wider use by the community, Libxc recently switched to a more permissive FOSS license that allows the library to be more easily included in closed-source programs. Libxc is now used in several proprietary and commercial software packages, e.g. the Slater-type orbital ADF package[97], and the Gaussian-type orbital GAMESS-US,[98] Molpro,[99] MRCC,[100] ORCA,[101] and TURBOMOLE[102] programs; several other packages are also contemplating to migrate to Libxc.

The advantages of the community adoption of Libxc are manifold. A new density functional approximation only needs to be implemented in Libxc to become available in any of the electronic programs that support Libxc, underlining the e ciency 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,[103] which is important to be able to compare results obtained with di erent 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.[104]

We believe that computational chemistry will continue to transform by adopting more and more FOSS components, the Electronic Structure Library (ESL) being one of the notable pushes in this direction.[105] Well-designed, modular FOSS components can be maintained even by a single academic group; the semi-empirical dispersion library of the Grimme group is a successful recent example.[106–108] We will discuss this topic further in Subsection III E.

C. What does free and open source software offer for teaching?

1. *Free redistribution: install and maintenance*

In addition to its benefits for general use cases,[109] 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 Subsection II A 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, simply by running a single command, or alternatively, finding the program in the distribution's graphical application manager and clicking on "Install".

We wish to note here that although installing scientific software by hand by compiling from source code a ords 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 a orded by the centralized packaging system. Compiling from source takes a lot of time as well as expertise, and can lead to poor performance if the compiler options are not adequately chosen; note that several proprietary programs have likewise adopted a binary-only distribution model with the same limitations.

However, installation is only a part of the problem: the software must also be kept up to date. This does not happen automatically, and a constant level of administration e ort is then required to monitor new releases, and to download and install new versions of the software. In contrast, the Linux distribution packages get automatically updated with the rest of the system whenever new package versions come out: Linux package managers not only handle updates to the Linux operating system kernel, but also all other software, such as the internet browsers, the email clients, the o ce productivity software suites, the Fortran and LaTeX compilers, and so on. Also computational chemistry packages get automatically updated.

2. *Access to source code*

The second advantage of FOSS is that 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, Psi4Numpy[110] is a project that aims to supply simple, easily modifiable Python algorithms for educational and proof-of-concept purposes. The PySCF quantum chemistry program[80] makes it easy to override and customize all algorithms, as they are mostly written in Python. Similarly, DFTK[111] has been designed to facilitate algorithmic development and might therefore also be useful for educational purposes.

Access to these kinds of projects 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: instead, it can also include real-life demonstrations. For example, an advanced course on electronic structure theory could involve asking students to write their own, customized solver for self-consistent field theory.[112]

3. *Sophisticated workflows*

The third advantage of FOSS for teaching is that since students (like anyone else) can access the full power of various computational chemistry programs, they also have the the possibility to develop more general technical skills such as programming and interfacing programs with each other, for instance by generating sophisticated workflows that automate complex tasks. Automated workflows are highly useful tools for practical computations, as they can be leveraged to easily run and analyze thousands to even millions of calculations that are needed for high-performance screening of materials, for instance. Several large-scale projects such as Materials Project,[113] Materials Cloud,[114] AiiDA,[115] Atomic Simulation Recipes,[116] and QCEngine[117] are FOSS and provide immediate access to powerful automated workflows for computational chemistry. As was summarized in the first criterion in Subsection II A, FOSS can also be freely used without limitations in industry to develop new thermoelectric energy conversion materials[118] or semiconductor devices,[119] for example, underlining its freedom and flexibility.

D. 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 I. 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 a ected university policies. Students preferring to use their own devices has lead to a significant decrease in the demand for computer classrooms. Universities may now find it cheaper to just o er a laptop to all students. For instance, the Faculty of Science of the University of Helsinki 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 refuse the laptop o ered by the university and opt to using their private laptops instead.

Although as was already discussed in section I, a centralized compute server approach is compatible with the BYOD paradigm, the e ortless 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 optimal 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 tradeo s are implied in a course employing 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.[120,121] The estimated performance of the fastest and slowest 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[122] 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 and kept up to date in a trivial fashion with a package manager, computational chemistry can finally be made available to the masses, as 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 licenses.

III. OVERVIEW OF AVAILABLE FOSS PROGRAM PACKAGES

This section presents an overview of available FOSS program packages for 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,[123,124] while various computational chemistry resources for education have been recently summarized by Rodríguez-Becerra et al.[125].

As the availability of software is a moving goalpost, since new packages appear and old ones become technologically obsolete and stop being maintained, any review can by force of necessity only represent the situation at a given point in time. Continuously updated databases are an alternative that is (hopefully) always up to date,[126] but any observations made on their basis similarly are tied to the time of observation and become outdated as enough time passes. For this reason, new reviews are typically published whenever the availability of software has changed enough.

The main goal of this section is merely to illustrate the breadth of software that is already available for use in computational chemistry. We have assembled the collection of packages by thorough literature and internet searches. Because unmaintained packages are unlikely to be easy to install, or to become available as prepackaged software, we limit the overview to software that shows at least some development activity in recent years, as checked from the upstream development repositories. Even if it later turns out that we have missed some recently published software package in this review, or if some packages become replaced by newer competitors after the publication of this article, our main points should remain una ected: there will likely still be a similar breadth of FOSS packages suitable for a variety of purposes within computational chemistry and computational chemistry education.

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. Linux distribution packages are centrally maintained by the Linux distribution's packagers, and require no special knowledge or local department personnel to install them or keep the software up to date, in contrast to typical proprietary packages. As we show in the Supporting Information, the packages can be installed on the command line; alternatively, they can also be installed using the distribution's application store. Importantly, the software is also automatically kept up to date by the distribution package manager, whereas the installation and upkeep of proprietary packages tends to require significant local expertise and time e ort.

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[127] (PyPI) and Conda[128] package managers are other alternatives for easy access to an increasing number of quantum chemistry packages 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, Subsection III A
- programs for solid-state calculations with various numerical approaches, Subsection III B
- programs employing fully numerical methods, Subsection III C
- programs employing semiempirical methods, Subsection III D

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

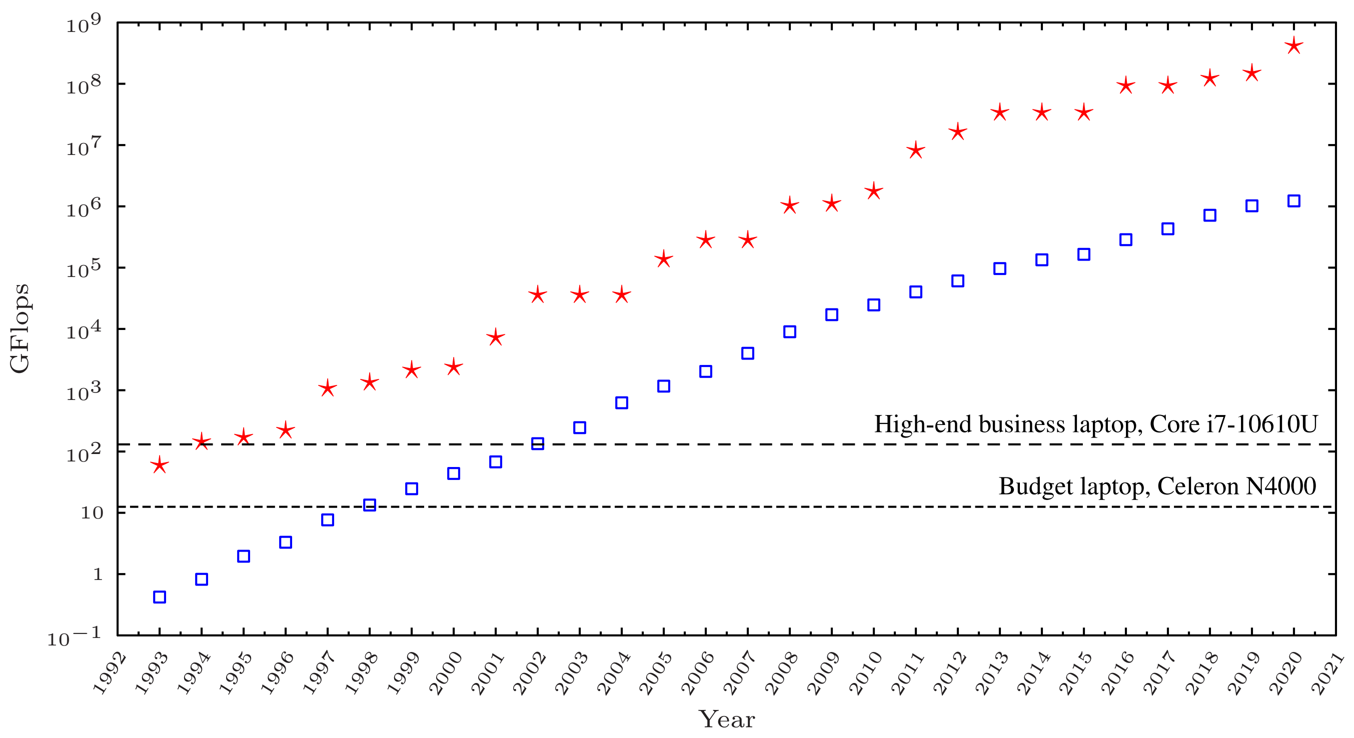

Figure 1. The best-performing (red stars) and worst-performing (blue squares) supercomputer on the TOP500 list,[121] as well as the performance of a budget laptop with a Celeron N4000 processor and a high-end business laptop with a Core i7-10610U processor (see supporting information). Note logaritmic scale on y axis. The performance of Raspberry Pi 4 was found to be similar to Celeron N4000.

chemistry course or other application. Most of the electronic structure programs support either Hartree–Fock (HF) and/or density functional theory[129,130] (DFT); several molecular programs also support various post-HF methods. We will also discuss projects of a more limited scope in Subsection III E.

A. Programs for molecular calculations with Gaussian basis sets

Gaussian basis sets dominate the field of quantum chemistry, since all electrons can e ciently be included in the calculation, the electronic Coulomb integrals can be evaluated analytically in the Gaussian basis,[131] and the evaluation is e cient when recursion relations are used.[132,133] Thanks to many decades of work on the development of Gaussian basis sets,[134–136] basis sets exist for the accurate reproduction of various molecular properties at several levels of theory. Access to analytical integrals greatly facilitates the implementation of post-HF theories, and also guarantees accurate force and Hessian evaluations.

Bagel[137] 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[138] 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 e ects in said treatments.

Dalton[139] 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[140] is a C++ program for linear-scaling HF and DFT calculations for molecules.

ERKALE[77] is a C++ program implementing HF and DFT that specializes in the modeling of inelastic x-ray spectroscopies, self-interaction corrected DFT, as well as various orbital localization methods.

e^T[141] 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.

Fermi.jl[142] is a Julia package for HF and post-HF calculations.

JuliaChem[143] is a Julia package for HF calculations.

LSDalton[139] is a Fortran code targeted for linear-scaling HF and DFT calculations on large molecular systems, and also includes some coupled-cluster capabilities.

MolGW[144] 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[145] 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[83] is a major quantum chemistry package written in Fortran and has a variety of features for both molecular and solid-state calculations.

Psi4[78] 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[80] 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[146] is a Python package for quantum chemistry with some C extensions that emphasizes ease of understanding the code over performance.

OpenMolcas[81] is a Fortran package that specializes in multiconfigurational approaches to electronic structure theory, but also implements various DFT calculations, for example.

Serenity[147] is a C++ program for subsystem quantum chemical methods.

SlowQuant[148] 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[149] is a C++/Python package for molecular properties and for modeling various spectroscopies based on response theory.

Uquantchem[150] is a Fortran 90 program written for HF, DFT, Møller–Plesset perturbation theory, configuration interaction singles and doubles, quantum Monte Carlo, etc.

B. Programs for solid-state calculations

The major di erence between solid-state and molecular calculations is that the orbitals experience exponential decay in molecular calculations, while solid-state calculations are performed on periodic crystals where the wave function has to obey Bloch's theorem.[151] Because of the periodicity, calculations in the solid state are in many ways more di cult than those in molecules due to the need of k-point sampling, for instance; see ref. 152 for a recent introduction. Post-HF methods are much less prominent in the solid state than in molecules. Instead, calculations on solids are typically carried out with DFT and pseudopotentials;[153] pseudopotentials make the calculations less costly while introducing an error which is typically negligible compared to the error in the density functional approximation itself.

The conventional way to model crystalline systems is to use plane waves. However, many other numerical schemes have also been pursued. Note that the programs listed here that employ (pseudo)atomic basis functions can naturally handle periodicity in 0, 1, 2, or 3 dimensions, corresponding to atoms and molecules, chains, sheets, and crystals, respectively. Still, we have listed them as solid state codes because they are most often used for calculations with DFT and pseudopotentials.

ABINIT[88] is Fortran program for plane wave calculations that supports DFT as well as more advanced formalisms like many-body perturbation theory.

ACE-Molecule[154] is a C++ program that employs uniform real-space grids of Lagrange sinc functions and pseudopotentials, and supports density functional calculations on both periodic and non-periodic systems and wave function theory calculations based on Kohn–Sham orbitals.

BigDFT[155] 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.

Conquest[156] is a Fortran program for large-scale DFT calculations employing pseudo-atomic orbital basis sets.

CP2K[157] 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[111] or the density-functional toolkit is a collection of Julia routines for experimenting with plane-wave DFT that emphasises simplicity and flexibility in the aim of facilitating algorithmic and numerical developments and simplify interdisciplinary collaboration in solid-state research.

ELK[158], EXCITING[159], and FLEUR[160] are Fortran programs for linearised augmented-plane wave calculations which can reach microhartree accurate total energies for carefully chosen basis sets.

GPAW[161] is Python/C electronic structure program for DFT calculations within the projector-augmented wave approach which supports three modes of operation: (i) finite-di erence grids, (ii) numerical atomic orbitals, and (iii) plane waves.

INQ[89] is a new, modular implementation of DFT and time-dependent DFT written from scratch to work on graphics processing units (GPUs).

JDFTx[162] is a C++ plane wave DFT code aimed to be easy to develop and easy to use, whose key feature is support for joint DFT for the description of electronic systems in contact with molecular liquids.

M-SPARC[163] is a MATLAB package for prototyping DFT calculations employing finite-di erence grids and pseudopotentials.

Octopus[164] is a Fortran program based on pseudopotentials and finite di erence grids that focuses on time-dependent DFT for handling non-equilibrium phenomena.

OpenMX[165] is a C package for DFT calculations with pseudopotentials and numerical atomic orbitals.

PARSEC[166] is a Fortran program based on finite-di erence grids for density functional calculations with pseudopotentials.

PWDFT.jl[167] is a Julia package written from scratch to facilitate development of novel computational methods using plane waves.

RMG[168] is a C++/Fortran program employing real space grids and multigrid algorithms for density functional calculations with pseudopotentials.

Siesta[169] 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[170] is a C++ program aimed for first principles molecular simulations using plane waves and pseudopotentials.

Quantum Espresso[90] is a Fortran/C program for plane wave calculations with pseudopotentials on a wide range of hardware from laptops to supercomputers.

SPARC[171] is a C program for parallel DFT calculations employing finite-di erence grids and pseudopotentials.

C. Programs relying on fully numerical representations

The idea in modern fully numerical methods is to represent the orbitals directly in real space, and to use a representation of non-uniform accuracy (more grid points near the nuclei and fewer points in empty regions of the system) so that all-electron calculations become feasible. Although fully numerical approaches have a long history for calculations on atoms and diatomic molecules,[172] they are otherwise a relatively recent development in electronic structure theory and have only recently become competitive with e.g. Gaussian-basis calculations whenever high accuracy is needed.[173]

DFT-FE[95] 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[92,94] and diatomic molecules[91] at the HF or DFT levels of theory employing high-order numerical basis functions and yielding fully variational energies.

MADNESS[174] 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[173] 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.

x2dhf[175] is a Fortran program for non-relativistic finite di erence restricted open-shell Hartree–Fock and density functional calculations on diatomic molecules.

D. Programs employing semiempirical models

Semiempirical models o er a ordable techniques for approximate quantum mechanical calculations that fall in accuracy in-between ab initio density functional calculations and force field techniques. Tight-binding DFT[176–178] is probably the best-known semiempirical model, and it is available in several program packages. Other types of semiempirical methods exist as well, please refer to Thiel[179] and Bannwarth et al.[180] for discussion.

DFTB+[181] is a Fortran package for various calculations based on tight-binding DFT.

Latte[182] is a Fortran program for tight-binding DFT molecular dynamics.

Sparrow[183] is a C++/Python program for fast semiempirical quantum chemical calculations, including tight-binding DFT.

xtb[180] is a Fortran package that implements various semiempirical eXtended Tight-Binding methods.

E. Limited-scope projects

Although the main focus of our review is on self-contained packages for quantum electronic structure calculations for computational chemistry education, this narrow scope risks not seeing the forest from the trees. The major part of FOSS—the forest in the analogy—is a huge thriving ecosystem of small projects with limited scope, which wildly outnumber the more conspicuous large program packages—the trees—which exist in synergy with the smaller projects: the smaller subprojects are often used by the larger programs. Thereby, in order to gain a thorough overview of FOSS it is invaluable to extend our review from the self-contained packages reviewed above to projects of a more limited scope which often have little user visibility.

The proliferation of small projects has multiple raisons d'être. The most common one is simply a specific personal need. The good news is that because of the limited e ort required to develop and maintain a code with a well-defined scope, they can be developed and maintained by a single research group, or often even by a single person. The bad news is that probably the majority of all FOSS projects in existence are unmaintained, simply because the authors moved on to other things. As was already mentioned in the beginning of section III, we have not considered such projects in this review.

1. Keys to modular design

There is a systematic reason for the origin of the specific personal need mentioned in the previous paragraph: the DRY [Don't Repeat Yourself] and KISS [Keep It Simple, Stupid!] principles, which have been key principles in software engineering for an extended time and are still used to teach programming.[184]

DRY is a reminder to avoid code duplication: a given functionality should only be programmed once and that implementation called everywhere it is needed, instead of repeating the same functionality in several places of the program. The latter approach would be more verbose, making it less maintainable and more prone to bugs.

In KISS, a complex problem is broken down into smaller subtasks. Once the subtasks—the common pieces of the problem—have been identified, the principle is reapplied to the subtasks themselves: can they be broken down to a compact collection of even simpler tasks?

Once a KISS design has been established, each component has a clear role in the design of the whole program. Even though achieving the best design may in reality require several iterations of refactoring (restructuring) the code, the e ort in each iteration of the refactor is limited because even the code one is starting with should be quite simple if the initial application of KISS was even partly successful.

2. Is modular design a limitation?

A well-made design is like a puzzle: each software component fills in a piece of the puzzle by carrying out a small, well-defined task. Each piece should ideally be so small that a working implementation can be developed in a matter of hours.

The first attempt at the design of the program layout is often not fully successful, because the structure of a scientific problem is not always clear before it has been fully solved. For this reason, program structures tend to develop over time.

If a redesign of the modular structure of a problem leads to a more elegant or e cient implementation, it is often adopted in a new version of the software. Such redesigns are extremely common in software development, and are the reason for versioning software: the major version changes whenever the interface becomes incompatible with the older version.[185] However, the redesign is often achievable through simple reorganizations of the earlier code base. The software does not have to be rewritten, as the existing pieces can just be rearranged to fit the new pattern.

If the design of a modular library changes enough, it can essentially become a wholly new library. In this case, migrating to the newer version of the library may be a significant task for other projects, and the old and the new version of the library may coexist for an extended time. A good example in the field of quantum chemistry is the libint library of two-electron integrals,[186] which is used by several FOSS codes. A new major version of the library was introduced in 2014 to take advantage of the new features a orded by modern processors, but many quantum chemistry programs still use the original version published in the early 2000s, since the functionality provided by the older version su ces for the purposes it was designed for.

3. The importance of interoperability

An example of a modular design that has stood the test of time is the Basic Linear Algebra Subprogram (BLAS) library, which was originally introduced in the late 1970s.[187] BLAS implements elementary linear algebra operations, such as adding, scaling, and multiplying vectors and matrices; operations which hold a central place in most branches of computational science, including quantum chemistry, much of which is linear algebra.

Although a simple for-loop based implementation of BLAS operations, such as matrix-matrix multiplication $C_{ik} = \sum_j A_{ij} B_{jk}$ can be written up in minutes, the mathematical structure of the problems can be employed to design a faster implementation. In a later step, the implementation can even be hand-optimized to the specific processor used in the machine; competing optimized BLAS implementations are an active area of research.[188,189]

Although BLAS was published well before the FOSS movement gained steam via the internet, it serves as an excellent example of what can be achieved by the use of open source, or at least by sharing a common programming interace. BLAS is so pervasive, since it is ubiquitous: everyone uses it, and there are many competing implementations. When individual projects are interoperable, such as in the case of BLAS, the development of e cient programs is greatly hastened. Simply by using an optimized BLAS library instead of the reference implementation can in many cases yield speedups of several orders of magnitude.

Unfortunately, interoperability is still hampered in the field of quantum chemistry since components are not truly interoperable due to the lack of common standards. The evaluation of two-electron integrals is a good example: it is the rate determining step in conventional Hartree–Fock calculations, and several implementations of two-electron integrals have been published.[186,190–192] However, these implementations do not share a common interface. Instead, the interfaces tend to reflect the structure of earlier legacy codes that have a large number of di erring conventions on the ordering, normalization, and signs of Gaussian basis functions, for instance. Despite some attempts,[193,194] two-electron integrals libraries—or quantum chemistry programs, for that matter!—are still not interoperable.

4. *The move to increased modularity*

The situation may, however, be slowly changing. Libxc[84] has already standardized density functional calculations in over 30 electronic structure programs; XCFun[195] is another implementation of density functional approximations like Libxc that has also been adopted by many codes, several of which support both Libxc and XCFun. Other types of libraries are also following suit. There is a growing ecosystem of modular electronic structure libraries as recently discussed by Oliveira et al.[105] in the scope of solid state calculations. We will complement it with a brief overview of some modular open source projects that have become used within several quantum chemistry programs below. The use of common implementations will hopefully lead to more interoperability between electronic structure programs also in other aspects.

Given the multitude of small libraries that are available, the listing in this subsection is likely far from complete; however, its goal is merely to illustrate that there is more to FOSS than the self-contained packages listed above. Specialized projects like these eliminate redundant work and enable rapid implementation of new features in quantum chemistry programs.

Polarization, embedding and quantum chemical models are a good example of modular functionality, since the data structures needed to implement such models fit well in the modular design. Examples of such projects include:

CheMPS2[196] is an implementation of the density matrix renormalization group method.

cppe[197] is an implementation of polarizable embedding.

DFT-D3[198] and DFT-D4[107] are implementations of semiempirical dispersion corrections for density functional calculations.

libefp[199] is an implementation of the e ective fragment potential method.

Libxc[84] contains implementations of density functional approximations which have been generated with computer algebra.

PCMSolver[200] is an open-source library for the polarizable continuum model electrostatic problem.

XCFun[195] contains implementations of density functional approximations which employ automatic differentiation.

There are also several projects that specifically deal with Gaussian basis sets and that are thereby used by several quantum chemistry codes.

The Basis Set Exchange[201] is a Python library for storing and managing Gaussian basis sets and converting basis sets between various program formats; the project also has a web interface at http://www.basissetexchange.org which will be more familiar to most readers.

erd[190] computes two-electron integrals with Rys quadrature.

libint[186] is a library for the evaluation of molecular integrals of many-body operators over Gaussian functions employing Obara–Saika recursion routines.

libcint[191] is an integral library for automatically implementing general integrals for Gaussian-type scalar and spinor basis functions using Rys quadrature.

simint[192] is a vectorized library for electron repulsion integrals employing Obara–Saika recursions.

libecpint[202] is a software library for evaluating e ective core potential integrals.

5. Visualization, manipulation and analysis

The visualization, manipulation, and analysis tools discussed in this subsection are user-facing programs and are thereby a more visible showcase of limited-scope projects than the lower-level libraries that were discussed in Subsection III E 4. Indeed, simplified frontends are often invaluable for initializing, visualizing and analyzing calculations. 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. For creating models and visualizing computational results, FOSS graphical user interfaces such as Jmol[203], Avogadro[204], IQmol[205] and PyMol[206] can be installed and used.

Unfortunately, the interoperability challenges mentioned in Subsection III E a ect visualization and analysis tools especially acutely, because these applications tend to require access to the electronic wave function, for which no universally accepted standard exists. This problem plagues the whole field of computational chemistry, a ecting both FOSS and proprietary programs. In the lack of a universal standard, the interconversion of various input and output file formats between di erent programs can be carried out for example with the Open Babel[207] and cclib[208] packages.

The Atomic Simulation Environment (ASE)[209] contains versatile tools for building molecular and periodic models and enables easy retrieval of molecular structures from structural databases such as PubChem.[210] It can also act as a frontend to several quantum chemical programs, thus o ering a unified interface.

Calculations can be postprocessed with the Multiwfn[211] and ORBKIT[212] packages, for instance, which both support several file formats.

IV. 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 program packages are used in the practical illustrations: xtb (Subsection IV A), NWChem (Subsection IV B), Psi4 (Subsection IV C), and Quantum Espresso (Subsection IV D). 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.[213]

A. 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.[180,214] The GFNn-xTB methods implemented in xtb are semiempirical quantum chemical methods[180] 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-performance laptop computer. xtb is a powerful tool in the pre-optimization of geometries and molecular conformations before computationally more demanding calculations, for instance; see ref. 215 for a recent application to water oxidation catalysis.

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-$[Pt(NH_3)_2Cl_2]$, is a chemotherapy medication used in cancer treatments whose stereoisomer, transplatin, trans-$[Pt(NH_3)_2Cl_2]$, is ine ective 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, initial geometries can be obtained from structural databases such as Pubchem;[210] 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 su ciently close to optimal to allow for a straightforward optimization without di culties, 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

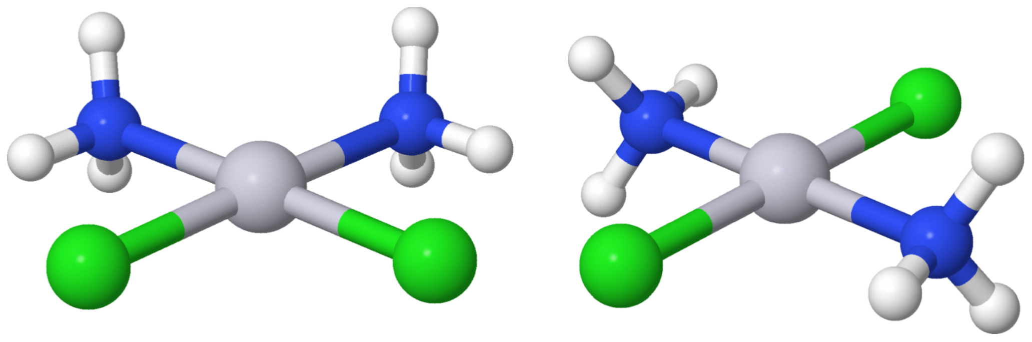

Figure 2. Cisplatin (left) and transplatin (right). Color coding: Pt = gray, Cl = green, N = blue, and H = white.

```
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.

on a low-performance 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 the much higher-level methods of Tasinato, Puzzarini, and Barone [216] who employed coupled-cluster theory with full single and double 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$(NH_3)_2Cl_2$]. For comparison, Liu and Franke [217] reported an energy di erence 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.[218,219] 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 Subsection IV B, we will revisit cisplatin and transplatin with DFT calculations that a ord a step up in accuracy over xtb.

B. 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 Subsection IV A.

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[85,220,221] (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.[222,223]

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 any electronic structure calculation in general, such as our attempted PBE0 calculation with NWChem. 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 Subsection IV A did not require the specification of a basis set, as the basis set is already 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.[134] Although many readers will be familiar with traditional basis sets like STO-3G,[224] 3-21G[225] and 6-31G*,[226] the development of computer processors and quantum chemical models in recent decades have also lead to significant advances in basis set design. Hundreds of Gaussian basis sets intended for various purposes are nowadays available on the Basis Set Exchange,[201] for example.

Because the basis set is an approximation, it is highly desirable to be able to control its accuracy in order to make tradeo s between the cost of the calculation and the accuracy of the obtained results. Accordingly, modern basis sets typically come in families of varying size:[135,136] 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-e cient approach to the complete basis set limit, at which point the error in the one-electron basis set no longer a ects the calculation. Note that also other types of basis sets than Gaussians may be used for quantum chemistry, see ref. 172 for further discussion.

In this work, we will only consider the Karlsruhe def2 family of Gaussian basis sets,[227] 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 e ects are already essential for chemistry of the 5th row,[228,229] relativistic e ects are described in the def2 basis sets through the use of e ective core potentials (ECPs).[230] 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 in 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 su ce 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. Di use 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.[231]

For the present demonstration, we choose the def2-TZVP basis set, as triple-ζ basis sets are well-known to yield energies that are su ciently close to the complete basis set limit (see also the applications in Subsections IV C 1 and IV C 2). Although hybrid functionals are computationally more demanding than non-hybrid functionals, it is notable that the dispersion-corrected hybrid PBE0-D4 generalized gradient approximation (GGA) functional was recently shown to outperform the dispersion-corrected, meta-GGA-type non-hybrid r^{2}SCAN-D4 functional in accuracy even for reaction energies of metal–organic reactions.[232]

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 Subsection IV A. 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 ex-

```
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.

cellent agreement with the values of Tasinato, Puzzarini, and Barone[216] that were discussed in Subsection IV A, 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 di erence of 56 kJ/mol obtained by Liu and Franke[217] using a high-level CCSD(T) method; however, in contrast to their CCSD(T) calculations, our DFT calculations can be performed 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 Subsection IV A, 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. The energetical ordering of orbitals can easily switch when the orbitals have similar energies; reorderings of the occupied orbitals have no e ect on the properties of the system.

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.

C. 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.[79] Compared to Psi3, Psi4 is designed to be a user-friendly, general-purpose code for fast, automated computations on molecules with hundreds of atoms.[78] Psi4 contains a number of computational methods ranging from HF and DFT to post-HF methods such as Møller–Plesset perturbation theory,[233] coupled-cluster theory,[234] configuration interaction theory, orbital-optimized correlation methods, symmetry-adapted perturbation theory, multireference methods etc.[78] 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 special consideration on the one-electron basis set. We will again focus on the def2 family of basis sets that was introduced in Subsection IV B.

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 editor; 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

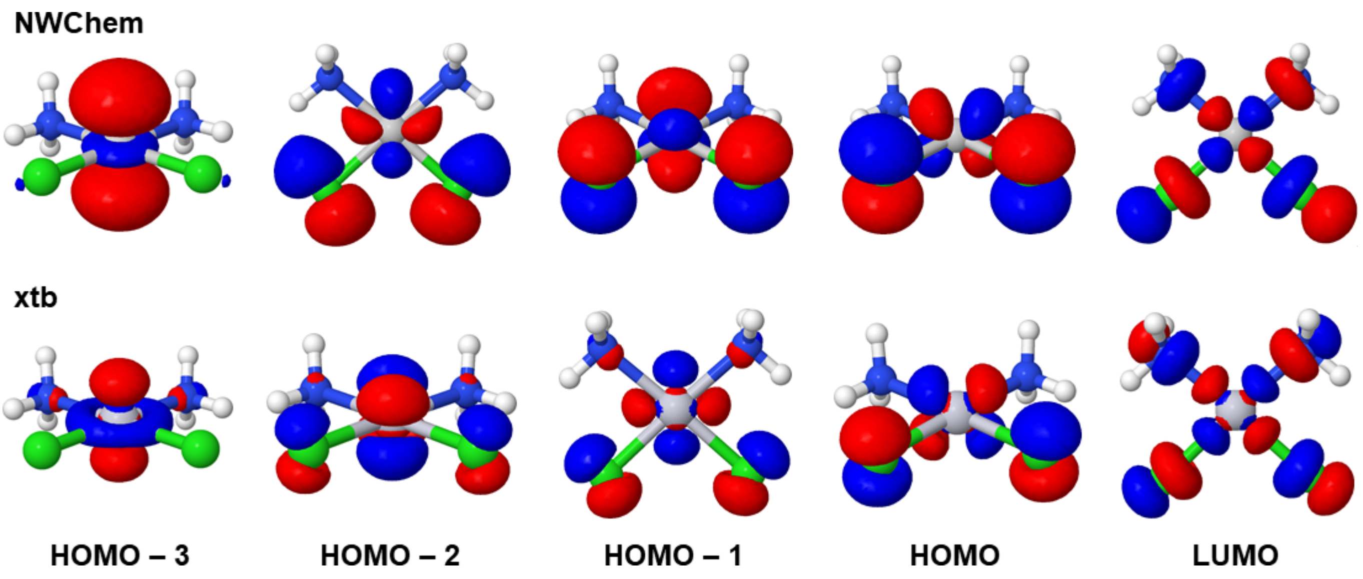

Figure 6. The four highest occupied MOs (HOMOs) and the lowest unoccupied MO (LUMO) of cisplatin as obtained from NWChem (PBE0/def2-TZVP) and xtb (GFN2-xTB). The color code for the nuclei is the same as in figure 2, while red and blue denote positive and negative orbital amplitudes, respectively (note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

without problems, and so that the bonding pattern does not change.

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.[235–237] Again, the SI includes short tutorials for installing and using the CREST code, which employs xtb to carry out conformational searches of molecules.[236] CREST finds four conformers, and outputs them in an increasing order in energy.

The four conformers are then reoptimized in Psi4 using the PBE0/def2-TZVP[85,220,221,227] level of theory introduced above in Subsection IV B. Psi4 employs density fitting[238–242] by default; this means that the universal fitting basis for Hartree–Fock calculations[243] 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 four conformers, we perform single-point calculations on each conformer in a variety of basis sets; the resulting energy di erences to the lowest-energy conformer (#1) are given in table I. 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,[244] 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.[224] (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[245,246] and remain widely used for practical calculations in programs employing STO basis sets.)

The data in table I 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 3 and 4, 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 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.[172]

The shortcomings of minimal basis sets are showcased by the large di erences between the results obtained with the MINAO and STO-nG basis sets. 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, and minimal basis sets generally have poor predictive power for chemistry.[135,136] MINAO is derived from atomic calculations only, and is thereby fully biased towards atoms, while the Slater-type orbital basis used by Hehre, Stewart, and Pople[224] is optimized for an average molecular environment, which is reflected in the slightly improved results in table I. However, this is only achieved at the cost of a bias towards molecules, meaning that the STO-nG basis sets are not as good for isolated atoms.

It is generally preferable to use larger and more flexi-

```
molecule {
0 1
 C    -1.0139237009     0.0001157060    -0.3320119090
 C    -0.3010211074     1.2491572923     0.1879180723
 C    -0.3011951696    -1.2490517349     0.1878718396
 C     1.1683390004     1.2516621049    -0.2233071254
 C     1.1681695646    -1.2517772582    -0.2232981267
 C     1.8703096243    -0.0000923733     0.2934985390
 C    -2.4834630882     0.0000222911     0.0795247173
 H    -0.9582190930     0.0002005854    -1.4269602139
 H    -0.3718670923     1.2740378936     1.2781840671
 H    -0.7951641526     2.1435985756    -0.1985907954
 H    -0.7954642203    -2.1434127996    -0.1986469736
 H    -0.3720420205    -1.2738839625     1.2781559690
 H     1.6616052523     2.1443202680     0.1678151692
 H     1.2391547021     1.2815104197    -1.3133695212
 H     1.2390062002    -1.2817145988    -1.3133390411
 H     1.6612233508    -2.1444918905     0.1679208818
 H     2.9153982958    -0.0001245162    -0.0238784763
 H     1.8521966765    -0.0001224730     1.3859698783
 H    -2.5743116471     0.0004900512     1.1639789401
 H    -2.9899376694     0.8819226593    -0.3066017637
 H    -2.9892458557    -0.8827595520    -0.3054682049
}

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

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

ble basis sets in applications, which guarantee a uniform accuracy for all types of systems, and to try to converge the 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 I. 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 conformer energy di erences, demonstrating the usefulness of modern, systematic basis set families: the complete basis set limit can be reached simply by using larger and larger basis sets.

For comparison, table I also includes data for the GFN2-xTB method.[214] 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 conformer energy di erences 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.

Studying a single molecular geometry is in general insu cient, if the molecule has the potential for multiple low-lying conformers. The data in table I demonstrates the importance of proper conformational sampling in applications to thermochemistry or chemical reactions, for instance: in the case of methylcyclohexane, insu cient conformational sampling can cause errors of up to 7 kcal/mol which may easily surpass the error arising from the level of theory or the basis set.

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.[247] 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

method	N_{bf}	conformer 2	conformer 3	conformer 4
PBE0/STO-3G	49	1.19	5.54	5.78
PBE0/STO-6G	49	1.25	5.57	5.84
PBE0/MINAO	49	0.85	5.08	5.05
PBE0/def2-SV(P)	126	2.00	6.62	7.07
PBE0/def2-SVP	168	1.97	6.57	7.01
PBE0/def2-TZVP	301	2.10	6.31	6.74
PBE0/def2-QZVP	819	2.11	6.31	6.73
GFN2-xTB		1.51	5.32	5.36

Table I. Conformer energy differences $\Delta E^{\mathrm{conformer}\ n} = E^{\mathrm{conformer}\ n} - E^{\mathrm{conformer}\ 1}$ 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, the GFN2-xTB data from the CREST output is also included.

run the geometry optimizations in Psi4; however, now the optimization is done separately for each basis set in contrast to the procedure used in Subsection IV C 1.

For this study, we choose the GFN1-xTB[248] and GFN2-xTB[214] semiempirical methods as well as a set of nonempirical density functionals: the Perdew–Wang 1992 (PW92) local density approximation (LDA),[151,249,250] the Perdew–Burke–Ernzerhof (PBE) GGA,[85] as well as the r^{2}SCAN meta-GGA functional that represents the state of the art in non-empirical density functionals.[251,252] 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;[253] however, for simplicity we stick to using the Psi4 default which is to use the larger auxiliary basis sets[243] that also work in the presence of exact exchange, such as the PBE0 functional used in Subsections IV B and IV C 1.

The results shown in table II demonstrate that while the STO-nG minimal basis sets[224,254] yield relatively poor geometries compared to the experimental values from refs. 255 and 256, already the split-valence def2-SV(P) basis set[227] leads to bond lengths that are converged to 0.03 Å and fractions of a degree in angles. The di erences become smaller, that is, the bond lengths and angles become more converged going to the larger basis sets, with the di erences 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. 255; 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. 256.

D. 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, a 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 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 di erences between the two polymorphs arise only in the third shell of neighbors.[257] 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.[258] Wurtzite-ZnS is thus metastable at room temperature, but it is found in nature and can also be produced synthetically.

The illustrative QE calculations are carried out with the non-empirical PBE exchange-correlation functional.[85] 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.[259] The main attractive feature of the GBRV pseudopotentials is that they are tailored for relatively small plane wave cuto s of 40 Rydberg for wave functions and 200 Rydberg for the charge density and potential,[259] resulting 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),[260] which is where we obtained the structures in the Crystallographic Information File (CIF) format; the COD structures are available in the Supporting Information.

There are several ways in which the crystal structures can be entered in QE input files. In the example here,

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^{2}SCAN	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		

Table II. Geometric parameters of chromyl fluoride (CrO_2F_2) at various levels of theory. [a]Experimental values from ref. 255. [b]Experimental values from ref. 256.

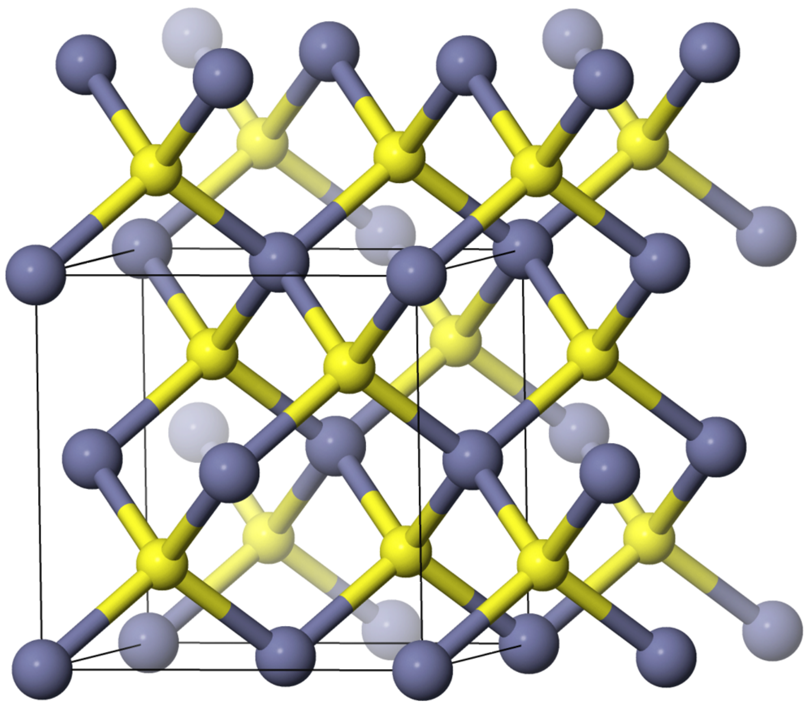
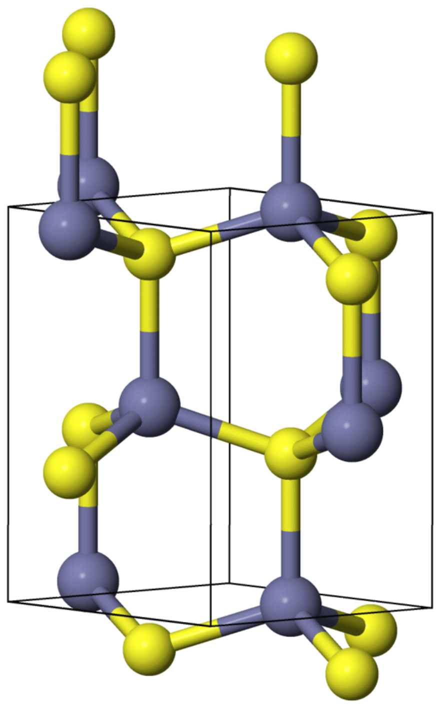

Figure 8. Two polymorphs of ZnS: sphalerite (left) and wurtzite (right). Zinc atoms in blue, sulfur atoms in yellow. For wurtzite, the c-axis points upwards.

```
&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 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.

we have directly used the crystallographic information to create an input file, which is shown in figure 9; a helpful resource for building QE input files is a orded by the QE input generator and structure visualizer provided by the Materials Cloud.[261]

1. *Optimal geometry*

Before attempting any calculations, it is important to determine how dense a sampling of the reciprocal space (k-sampling) is needed to describe the materials su -ciently accurately. The convergence tests described in the Supporting Information show that a 8×8×8 Monkhorst–Pack[262] k-point 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.[260] 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 di er by only 0.6 kJ/mol per formula unit. This value is in good agreement with Cardona et al. [263] who reported an energy di erence of less than 0.008 eV (0.8 kJ/mol) per formula unit from LDA and GGA calculations on ZnS polymorphs. The energy di erence is so small, because the crystal structures are so similar: differences arise only in the third-nearest neighbor shell, as was already mentioned above. 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.

2. *Band structure*

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,[264] 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 [265], 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.[113] However, the PBE calculations severely underestimate the experimental band gap measured at 10 K, which is about 3.8 eV.[266] The agreement with experiment could be improved for exam-

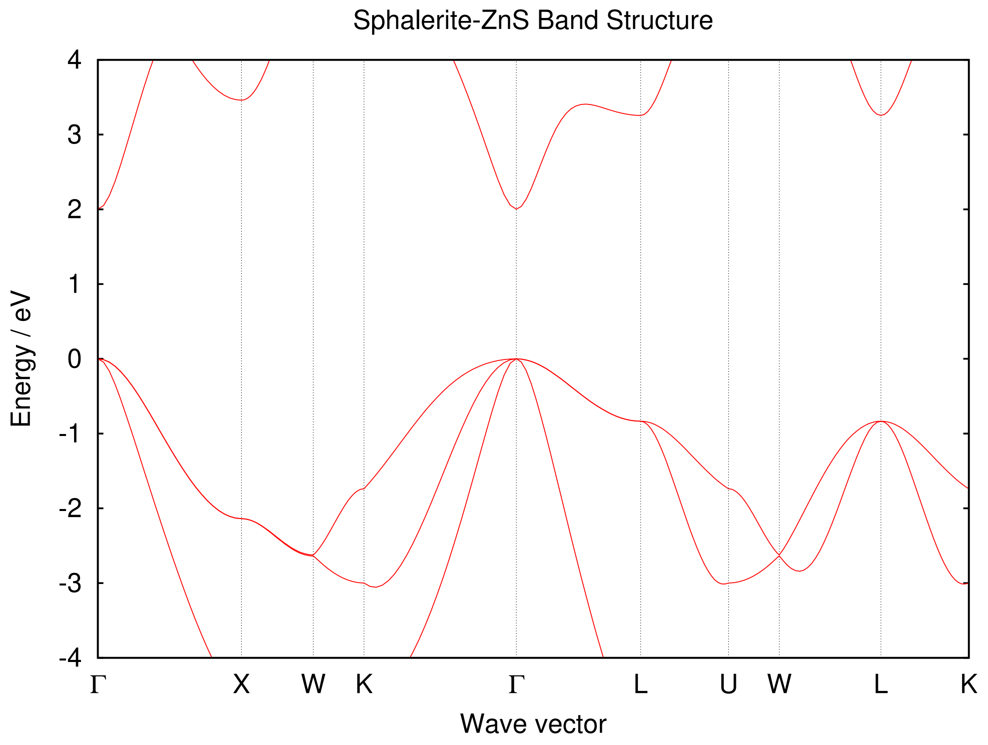

Figure 10. Electronic band structure of sphalerite-ZnS obtained with PBE functional and GBRV pseudopotentials.

ple with the DFT+U approach or with hybrid density functionals, both of which are outside the scope of this work.

V. 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 a ords 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-e ective 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 focus on content creation and developing practices for sharing and co-creating computational chemistry teaching material as Open Educational Resources.[267] The Psi4Education project[5,268] is one such attempt at open teaching materials. We hope open materials become more readily available and more thoroughly used in the future.

On a final note, we would like to point out that the free availability of FOSS operating system kernels, compilers, debuggers as well as user-space tools—which have not been discussed in this review—have had a critical role in enabling the development of the plethora of the FOSS projects discussed within this work, as well as our own work. We would like to thank the entire FOSS community for providing high-quality tools for a variety of purposes, and invite our readers to join the FOSS movement.

SUPPORTING INFORMATION

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.[213]

ACKNOWLEDGMENTS

We thank Paul Saxe and Jonathan Moussa for invaluable comments on an early stage of this manuscript. We also thank all the anonymous peer reviewers of this manuscript for constructive criticisms which have similarly helped to improve the structure and content of this paper. A. J. K. thanks Business Finland for Co-Innovation funding (Grant No. 3767/31/2019).

REFERENCES

[1]P. Westmoreland, *Applying Molecular and Materials Modeling*, 1st ed. (Springer Netherlands, 2002).
[2]M. Head-Gordon and E. Artacho, "Chemistry on the computer," Phys. Today **61**, 58–63 (2008).
[3]P. Deglmann, A. Schäfer, and C. Lennartz, "Application of quantum calculations in the chemical industry – An overview," Int. J. Quantum Chem. **115**, 107–136 (2015).
[4]H. Weiß, P. Deglmann, P. J. in 't Veld, M. Cetinkaya, and E. Schreiner, "Multiscale materials modeling in an industrial environment," Annu. Rev. Chem. Biomol. Eng. **7**, 65–86 (2016).
[5]R. C. Fortenberry, A. R. McDonald, T. D. Shepherd, M. Kennedy, and C. D. Sherrill, "PSI4Education: Computational chemistry labs using free software," in *The Promise of Chemical Education: Addressing our Students' Needs* (American Chemical Society, 2015) pp. 85–98.
[6]A. Grushow and M. Reeves, *Using Computational Methods To Teach Chemical Principles* (American Chemical Society, 2019).
[7]B. J. Esselman and N. J. Hill, "Integration of computational chemistry into the undergraduate organic chemistry laboratory curriculum," J. Chem. Educ. **93**, 932–936 (2016).
[8]L. L. Winfield, K. McCormack, and T. Shaw, "Using iSpartan to support a student-centered activity on alkane conformations," J. Chem. Educ. **96**, 89–92 (2018).
[9]B. J. Esselman and N. J. Hill, "Integrating computational chemistry into an organic chemistry laboratory curriculum using webmo," in *Using Computational Methods To Teach Chemical Principles*, Chap. 11, pp. 139–162.
[10]J. A. Phillips, "Modeling reaction energies and exploring noble gas chemistry in the physical chemistry laboratory," in *Using Computational Methods To Teach Chemical Principles*, Chap. 4, pp. 33–50.
[11]M. S. Reeves, H. L. Berghout, M. J. Perri, S. M. Singleton, and R. M. Whitnell, "How can you measure a reaction enthalpy without going into the lab?" in *Using Computational Methods To Teach Chemical Principles*, Chap. 5, pp. 51–63.
[12]S. R. Martini and C. J. Hartzell, "Integrating computational chemistry into a course in classical thermodynamics," J. Chem. Educ. **92**, 1201–1203 (2015).
[13]K. M. Stocker, "Using electronic structure calculations to investigate the kinetics of gas-phase ammonia synthesis," in *Using Computational Methods To Teach Chemical Principles*, Chap. 3, pp. 21–32.
[14]H. D. Snyder and T. G. Kucukkal, "Computational chemistry activities with Avogadro and ORCA," J. Chem. Educ. **98**, 1335–1341 (2021).
[15]G. C. Hoover, A. P. Dicks, and D. S. Seferos, "Upper-year materials chemistry computational modeling module for organic display technologies," J. Chem. Educ. **98**, 805–811 (2021).
[16]P. Y. Furlan and E. T. Bell-Loncella, "Integrating computation and visualization to enhance learning IR spectroscopy in the general chemistry laboratory: Computer-assisted learning of IR spectroscopy," Spectrosc. Lett. **43**, 618–625 (2010).
[17]W. R. Martin and D. W. Ball, "Using computational chemistry to extend the acetylene rovibrational spectrum to C_2T_2," in *Using Computational Methods To Teach Chemical Principles*, Chap. 8, pp. 93–107.
[18]T. C. DeVore, "Introducing quantum calculations into the physical chemistry laboratory," in *Using Computational Methods To Teach Chemical Principles*, Chap. 9, pp. 109–125.
[19]JCE staff, "Computational chemistry for the masses," J. Chem. Educ. **73**, 104 (1996).
[20]A. Grushow and M. S. Reeves, "Using computational methods to teach chemical principles: Overview," in *Using Computational Methods To Teach Chemical Principles* (2019) Chap. 1, pp. 1–10.
[21]WebMO, "A web-based interface to computational chemistry packages," https://www.webmo.net/, accessed 8 May 2021.
[22]W. F. Polik and J. R. Schmidt, "WebMO: Web-based computational chemistry calculations in education and research," Wiley Interdiscip. Rev. Comput. Mol. Sci. (2021), 10.1002/wcms.1554.
[23]M. J. Perri, M. Akinmurele, and M. Haynie, "Chem compute science gateway: An online computational chemistry tool," in *Using Computational Methods To Teach Chemical Principles*, Chap. 7, pp. 79–92.
[24]R. Kobayashi, T. P. M. Goumans, N. O. Carstensen, T. M. Soini, N. Marzari, I. Timrov, S. Poncé, E. B. Linscott, C. J. Sewell, G. Pizzi, F. Ramirez, M. Bercx, S. P. Huber, C. S. Adorf, and L. Talirz, "Virtual computational chemistry teaching laboratories—hands-on at a distance," J. Chem. Educ. **98**, 3163–3171 (2021).
[25]S. Schwalbe, L. Fiedler, J. Kraus, J. Kortus, K. Trepte, and S. Lehtola, "PyFLOSIC: Python-based Fermi–Löwdin orbital self-interaction correction," J. Chem. Phys. **153**, 084104 (2020).
[26]A. I. Krylov, J. M. Herbert, F. Furche, M. Head-Gordon, P. J. Knowles, R. Lindh, F. R. Manby, P. Pulay, C.-K. Skylaris, and H.-J. Werner, "What is the price of open-source software?" J. Phys. Chem. Lett. **6**, 2751–2754 (2015).
[27]C. R. Jacob, "How open is commercial scientific software?" J. Phys. Chem. Lett. **7**, 351–353 (2016).
[28]L. Li, "Why should anyone become a scientist? the ideal of science and its importance," J. Chem. Educ. **76**, 20 (1999).
[29]P. Azoulay, C. Fons-Rosen, and J. S. G. Zivin, "Does science advance one funeral at a time?" Am. Econ. Rev. **109**, 2889–2920 (2019).
[30]J. Giles, "Software company bans competitive users," Nature **429**, 231–231 (2004).
[31]A. G. Smart, "The war over supercooled water," Physics Today (2018).
[32]J. C. Palmer, A. Haji-Akbari, R. S. Singh, F. Martelli, R. Car, A. Z. Panagiotopoulos, and P. G. Debenedetti, "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. **148**, 137101 (2018).
[33]Open Source Initiative, "The open source definition," https://opensource.org/osd, accessed May 13 2021.
[34]Free Software Foundation, "What is free software?" https://www.gnu.org/philosophy/free-sw.html.en, accessed May 13 2021.
[35]M. T. Stahl, "Open-source software: not quite endsville," Drug Discovery Today **10**, 219–222 (2005).
[36]J. D. Gezelter, "Open source and open data should be standard practices," J. Phys. Chem. Lett. **6**, 1168–1169 (2015).
[37]K. Hinsen, "Computational science: shifting the focus from tools to models," F1000Research **3**, 101 (2014).
[38]Git community, "Git, a free and open source distributed version control system," https://git-scm.com/, accessed 20 May 2021.
[39]GitHub, Inc., "Github collaboration platform," https://github.com/, accessed 20 May 2021.
[40]GitLab, Inc., "Gitlab collaboration platform," https://gitlab.com/, accessed 20 May 2021.
[41]European Organization For Nuclear Research and OpenAIRE, "Zenodo," (2013).
[42]J. Swarts, "Open-source software in the sciences: The challenge of user support," J. Bus. Tech. Commun. **33**, 60–90 (2018).
[43]A. Dalke, "The chemfp project," J. Cheminf. **11**, 76 (2019).
[44]G. Haff, *How Open Source Ate Software* (Apress, 2018).
[45]Kitware Inc., "About Kitware," https://www.kitware.com/about/, accessed 28 January 2022.
[46]J. Ahrens, B. Geveci, and C. Law, "ParaView: An end-user tool for large-data visualization," in *Visualization Handbook* (Else-

vier, 2005) pp. 717–731.

[47]M. McCormick, X. Liu, J. Jomier, C. Marion, and L. Ibanez, "ITK: enabling reproducible research and open science," Front. Neuroinf. **8**, 13 (2014).

[48]B. Hoffman, D. Cole, and J. Vines, "Software process for rapid development of HPC software using CMake," in *2009 DoD High Performance Computing Modernization Program Users Group Conference* (IEEE, 2009).

[49]M. D. Hanwell, C. Harris, A. Genova, M. Haghighatlari, M. E. Khatib, P. Avery, J. Hachmann, and W. A. Jong, "Open chemistry, JupyterLab, REST, and quantum chemistry," Int. J. Quantum Chem. **121**, e26472 (2021).

[50]European Commission, "Open science," https://ec.europa.eu/info/research-and-innovation/strategy/strategy-2020-2024/our-digital-future/open-science, accessed 28 January 2022.

[51]F. Wieber, A. Pisanty, and A. Hocquet, ""We were here before the Web and hype...": a brief history of and tribute to the computational chemistry list," J. Cheminf. **10**, 67 (2018).

[52]D. Constant, L. Sproull, and S. Kiesler, "The kindness of strangers: The usefulness of electronic weak ties for technical advice," Organ. Sci. **7**, 119–135 (1996).

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

[54]A. Schiff, "The economics of open source software: A survey of the early literature," Rev. Netw. Econ. **1**, 66–74 (2002).

[55]D. P. Myatt, "Equilibrium selection and public-good provision: The development of open-source software," Oxford Rev. Econ. Pol. **18**, 446–461 (2002).

[56]J. P. Johnson, "Open source software: Private provision of a public good," J. Econ. Manage. Strat. **11**, 637–662 (2002).

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

[58]J. Lerner and J. Tirole, "Some simple economics of open source," The Journal of Industrial Economics **50**, 197–234 (2003).

[59]A. Bonaccorsi and C. Rossi, "Why open source software can succeed," Res. Policy **32**, 1243–1258 (2003).

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

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

[62]J. Lerner and J. Tirole, "The economics of technology sharing: Open source and beyond," J. Econ. Perspect. **19**, 99–120 (2005).

[63]J. Lerner, "The scope of open source licensing," Journal of Law, Economics, and Organization **21**, 20–56 (2005).

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

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

[66]M. A. Rossi, "Decoding the free/open source software puzzle," in *The Economics of Open Source Software Development* (Elsevier, 2006) pp. 15–55.

[67]A. Gaudeul, "Do open source developers respond to competition? The (LA)TEX case study," Rev. Netw. Econ. **6**, 239–263 (2007).

[68]G. von Krogh and E. von Hippel, "The promise of research on open source software," Manage. Sci. **52**, 975–983 (2006).

[69]A. Hars and S. Ou, "Working for free? Motivations for participating in open-source projects," Int. J. Electron. Comm. **6**, 25–39 (2002).

[70]J. Bitzer, W. Schrettl, and P. J. H. Schröder, "Intrinsic motivation in open source software development," J. Comp. Econ. **35**, 160–169 (2007).

[71]J. Lerner, P. A. Pathak, and J. Tirole, "The dynamics of open-source contributors," Am. Econ. Rev. **96**, 114–118 (2006).

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

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

[74]J. Bitzer and P. J. H. Schröder, "The impact of entry and competition by open source software on innovation activity," in *The Economics of Open Source Software Development* (Elsevier, 2006) pp. 219–246.

[75]top500.org, "Top500 operating system statistics," https://www.top500.org/statistics/details/osfam/1/, accessed 6 July 2021.

[76]J. F. Moore and M. P. McCann, "Linux and the Chemist," J. Chem. Educ. **80**, 219 (2003).

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

[78]D. G. A. Smith, L. A. Burns, A. C. Simmonett, R. M. Parrish, M. C. Schieber, R. Galvelis, P. Kraus, H. Kruse, R. Di Remigio, A. Alenaizan, A. M. James, S. Lehtola, J. P. Misiewicz, M. Scheurer, R. A. Shaw, J. B. Schriber, Y. Xie, Z. L. Glick, D. A. Sirianni, J. S. O'Brien, J. M. Waldrop, A. Kumar, E. G. Hohenstein, B. P. Pritchard, B. R. Brooks, H. F. Schaefer, A. Y. Sokolov, K. Patkowski, A. E. DePrince, U. Bozkaya, R. A. King, F. A. Evangelista, J. M. Turney, T. D. Crawford, and C. D. Sherrill, "PSI4 1.4: Open-source software for high-throughput quantum chemistry," J. Chem. Phys. **152**, 184108 (2020).

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

[80]Q. Sun, X. Zhang, S. Banerjee, P. Bao, M. Barbry, N. S. Blunt, N. A. Bogdanov, G. H. Booth, J. Chen, Z.-H. Cui, J. J. Eriksen, Y. Gao, S. Guo, J. Hermann, M. R. Hermes, K. Koh, P. Koval, S. Lehtola, Z. Li, J. Liu, N. Mardirossian, J. D. McClain, M. Motta, B. Mussard, H. Q. Pham, A. Pulkin, W. Purwanto, P. J. Robinson, E. Ronca, E. R. Sayfutyarova, M. Scheurer, H. F. Schurkus, J. E. T. Smith, C. Sun, S.-N. Sun, S. Upadhyay, L. K. Wagner, X. Wang, A. White, J. D. Whitfield, M. J. Williamson, S. Wouters, J. Yang, J. M. Yu, T. Zhu, T. C. Berkelbach, S. Sharma, A. Y. Sokolov, and G. K.-L. Chan, "Recent developments in the PYSCF program package," J. Chem. Phys. **153**, 024109 (2020), arXiv:2002.12531.

[81]F. Aquilante, J. Autschbach, A. Baiardi, S. Battaglia, V. A. Borin, L. F. Chibotaru, I. Conti, L. De Vico, M. Delcey, I. Fdez. Galván, N. Ferré, L. Freitag, M. Garavelli, X. Gong, S. Knecht, E. D. Larsson, R. Lindh, M. Lundberg, P. Å. Malmqvist, A. Nenov, J. Norell, M. Odelius, M. Olivucci, T. B. Pedersen, L. Pedraza-González, Q. M. Phung, K. Pierloot, M. Reiher, I. Schapiro, J. Segarra-Martí, F. Segatta, L. Seijo, S. Sen, D.-C. Sergentu, C. J. Stein, L. Ungur, M. Vacher, A. Valentini, and V. Veryazov, "Modern quantum chemistry with [Open]Molcas," J. Chem. Phys. **152**, 214117 (2020).

[82]J. M. H. Olsen, S. Reine, O. Vahtras, E. Kjellgren, P. Reinholdt, K. O. Hjorth Dundas, X. Li, J. Cukras, M. Ringholm, E. D. Hedegård, R. Di Remigio, N. H. List, R. Faber, B. N. Cabral Tenorio, R. Bast, T. B. Pedersen, Z. Rinkevicius, S. P. A. Sauer, K. V. Mikkelsen, J. Kongsted, S. Coriani, K. Ruud, T. Helgaker, H. J. A. Jensen, and P. Norman, "Dalton project: A Python platform for molecular- and electronic-structure simulations of complex systems," J. Chem. Phys. **152**, 214115 (2020).

[83]E. Aprà, E. J. Bylaska, W. A. de Jong, N. Govind, K. Kowalski, T. P. Straatsma, M. Valiev, H. J. J. van Dam, Y. Alexeev, J. Anchell, V. Anisimov, F. W. Aquino, R. Atta-Fynn, J. Autschbach, N. P. Bauman, J. C. Becca, D. E. Bernholdt, K. Bhaskaran-Nair, S. Bogatko, P. Borowski, J. Boschen, J. Brabec, A. Bruner, E. Cauët, Y. Chen, G. N. Chuev, C. J. Cramer, J. Daily, M. J. O. Deegan, T. H. Dunning, M. Dupuis, K. G. Dyall, G. I. Fann, S. A. Fischer, A. Fonari, H. Früchtl, L. Gagliardi,

J. Garza, N. Gawande, S. Ghosh, K. Glaesemann, A. W. Götz, J. Hammond, V. Helms, E. D. Hermes, K. Hirao, S. Hirata, M. Jacquelin, L. Jensen, B. G. Johnson, H. Jónsson, R. A. Kendall, M. Klemm, R. Kobayashi, V. Konkov, S. Krishnamoorthy, M. Krishnan, Z. Lin, R. D. Lins, R. J. Littlefield, A. J. Logsdail, K. Lopata, W. Ma, A. V. Marenich, J. Martin del Campo, D. Mejia-Rodriguez, J. E. Moore, J. M. Mullin, T. Nakajima, D. R. Nascimento, J. A. Nichols, P. J. Nichols, J. Nieplocha, A. Otero-de-la Roza, B. Palmer, A. Panyala, T. Pirojsirikul, B. Peng, R. Peverati, J. Pittner, L. Pollack, R. M. Richard, P. Sadayappan, G. C. Schatz, W. A. Shelton, D. W. Silverstein, D. M. A. Smith, T. A. Soares, D. Song, M. Swart, H. L. Taylor, G. S. Thomas, V. Tipparaju, D. G. Truhlar, K. Tsemekhman, T. Van Voorhis, Á. Vázquez-Mayagoitia, P. Verma, O. Villa, A. Vishnu, K. D. Vogiatzis, D. Wang, J. H. Weare, M. J. Williamson, T. L. Windus, K. Woliński, A. T. Wong, Q. Wu, C. Yang, Q. Yu, M. Zacharias, Z. Zhang, Y. Zhao, and R. J. Harrison, "NWChem: Past, present, and future," J. Chem. Phys. **152**, 184102 (2020).

[84] S. Lehtola, C. Steigemann, M. J. T. Oliveira, and M. A. L. Marques, "Recent developments in LIBXC – a comprehensive library of functionals for density functional theory," SoftwareX **7**, 1–5 (2018).

[85] J. P. Perdew, K. Burke, and M. Ernzerhof, "Generalized gradient approximation made simple," Phys. Rev. Lett. **77**, 3865–3868 (1996).

[86] P. J. Stephens, F. J. Devlin, C. F. Chabalowski, and M. J. Frisch, "Ab initio calculation of vibrational absorption and circular dichroism spectra using density functional force fields," J. Phys. Chem. **98**, 11623–11627 (1994).

[87] J. Sun, A. Ruzsinszky, and J. Perdew, "Strongly constrained and appropriately normed semilocal density functional," Phys. Rev. Lett. **115**, 036402 (2015).

[88] A. H. Romero, D. C. Allan, B. Amadon, G. Antonius, T. Applencourt, L. Baguet, J. Bieder, F. Bottin, J. Bouchet, E. Bousquet, F. Bruneval, G. Brunin, D. Caliste, M. Côté, J. Denier, C. Dreyer, P. Ghosez, M. Giantomassi, Y. Gillet, O. Gingras, D. R. Hamann, G. Hautier, F. Jollet, G. Jomard, A. Martin, H. P. C. Miranda, F. Naccarato, G. Petretto, N. A. Pike, V. Planes, S. Prokhorenko, T. Rangel, F. Ricci, G.-M. Rignanese, M. Royo, M. Stengel, M. Torrent, M. J. van Setten, B. Van Troeye, M. J. Verstraete, J. Wiktor, J. W. Zwanziger, and X. Gonze, "ABINIT: Overview and focus on selected capabilities," J. Chem. Phys. **152**, 124102 (2020).

[89] X. Andrade, C. D. Pemmaraju, A. Kartsev, J. Xiao, A. Lindenberg, S. Rajpurohit, L. Z. Tan, T. Ogitsu, and A. A. Correa, "Inq, a modern GPU-accelerated computational framework for (time-dependent) density functional theory," J. Chem. Theory Comput. **17**, 7447–7467 (2021).

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

[91] S. Lehtola, "Fully numerical Hartree–Fock and density functional calculations. II. Diatomic molecules," Int. J. Quantum Chem. **119**, e25944 (2019), arXiv:1810.11653.

[92] S. Lehtola, "Fully numerical Hartree–Fock and density functional calculations. I. Atoms," Int. J. Quantum Chem. **119**, e25945 (2019), arXiv:1810.11651.

[93] S. Lehtola, M. Dimitrova, and D. Sundholm, "Fully numerical electronic structure calculations on diatomic molecules in weak to strong magnetic fields," Mol. Phys. **118**, e1597989 (2020), arXiv:1812.06274.

[94] S. Lehtola, "Fully numerical calculations on atoms with fractional occupations and range-separated exchange functionals," Phys. Rev. A **101**, 012516 (2020), arXiv:1908.02528.

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

[96] "MADNESS – Multiresolution Adaptive Numerical Environment for Scientific Simulation," .

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

[98] G. M. J. Barca, C. Bertoni, L. Carrington, D. Datta, N. De Silva, J. E. Deustua, D. G. Fedorov, J. R. Gour, A. O. Gunina, E. Guidez, T. Harville, S. Irle, J. Ivanic, K. Kowalski, S. S. Leang, H. Li, W. Li, J. J. Lutz, I. Magoulas, J. Mato, V. Mironov, H. Nakata, B. Q. Pham, P. Piecuch, D. Poole, S. R. Pruitt, A. P. Rendell, L. B. Roskop, K. Ruedenberg, T. Sattasathuchana, M. W. Schmidt, J. Shen, L. Slipchenko, M. Sosonkina, V. Sundriyal, A. Tiwari, J. L. Galvez Vallejo, B. Westheimer, M. Włoch, P. Xu, F. Zahariev, and M. S. Gordon, "Recent developments in the general atomic and molecular electronic structure system," J. Chem. Phys. **152**, 154102 (2020).

[99] H.-J. Werner, P. J. Knowles, F. R. Manby, J. A. Black, K. Doll, A. Heßelmann, D. Kats, A. Köhn, T. Korona, D. A. Kreplin, Q. Ma, T. F. Miller, A. Mitrushchenkov, K. A. Peterson, I. Polyak, G. Rauhut, and M. Sibaev, "The Molpro quantum chemistry package," J. Chem. Phys. **152**, 144107 (2020).

[100] M. Kállay, P. R. Nagy, D. Mester, Z. Rolik, G. Samu, J. Csontos, J. Csóka, P. B. Szabó, L. Gyevi-Nagy, B. Hégely, I. Ladjánszki, L. Szegedy, B. Ladóczki, K. Petrov, M. Farkas, P. D. Mezei, and Á. Ganyecz, "The MRCC program system: Accurate quantum chemistry from water to proteins," J. Chem. Phys. **152**, 074107 (2020).

[101] F. Neese, F. Wennmohs, U. Becker, and C. Riplinger, "The ORCA quantum chemistry program package," J. Chem. Phys. **152**, 224108 (2020).

[102] S. G. Balasubramani, G. P. Chen, S. Coriani, M. Diedenhofen, M. S. Frank, Y. J. Franzke, F. Furche, R. Grotjahn, M. E. Harding, C. Hättig, A. Hellweg, B. Helmich-Paris, C. Holzer, U. Huniar, M. Kaupp, A. Marefat Khah, S. Karbalaei Khani, T. Müller, F. Mack, B. D. Nguyen, S. M. Parker, E. Perlt, D. Rappoport, K. Reiter, S. Roy, M. Rückert, G. Schmitz, M. Sierka, E. Tapavicza, D. P. Tew, C. van Wüllen, V. K. Voora, F. Weigend, A. Wodyński, and J. M. Yu, "TURBOMOLE: Modular program suite for ab initio quantum-chemical and condensed-matter simulations," J. Chem. Phys. **152**, 184107 (2020).

[103] K. Lejaeghere, G. Bihlmayer, T. Bjorkman, P. Blaha, S. Blugel, V. Blum, D. Caliste, I. E. Castelli, S. J. Clark, A. D. Corso, S. de Gironcoli, T. Deutsch, J. K. Dewhurst, I. D. Marco, C. Draxl, M. D. ak, O. Eriksson, J. A. Flores-Livas, K. F. Garrity, L. Genovese, P. Giannozzi, M. Giantomassi, S. Goedecker, X. Gonze, O. Granas, E. K. U. Gross, A. Gulans, F. Gygi, D. R. Hamann, P. J. Hasnip, N. A. W. Holzwarth, D. I. an, D. B. Jochym, F. Jollet, D. Jones, G. Kresse, K. Koepernik, E. Kucukbenli, Y. O. Kvashnin, I. L. M. Locht, S. Lubeck, M. Marsman, N. Marzari, U. Nitzsche, L. Nordstrom, T. Ozaki, L. Paulatto, C. J. Pickard, W. Poelmans, M. I. J. Probert, K. Refson, M. Richter, G.-M. Rignanese, S. Saha, M. Scheffler, M. Schlipf, K. Schwarz, S. Sharma, F. Tavazza, P. Thunstrom, A. Tkatchenko, M. Torrent, D. Vanderbilt, M. J. van Setten, V. V. Speybroeck, J. M. Wills, J. R. Yates, G.-X. Zhang, and S. Cottenier, "Reproducibility in density functional theory calculations of solids," Science **351**, aad3000 (2016).

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

[105] M. J. T. Oliveira, N. Papior, Y. Pouillon, V. Blum, E. Artacho, D. Caliste, F. Corsetti, S. de Gironcoli, A. M. Elena, A. García, V. M. García-Suárez, L. Genovese, W. P. Huhn, G. Huhs, S. Kokott, E. Küçükbenli, A. H. Larsen, A. Lazzaro, I. V. Lebedeva, Y. Li, D. López-Durán, P. López-Tarifa, M. Lüders, M. A. L. Marques, J. Minar, S. Mohr, A. A. Mostofi, A. O'Cais, M. C. Payne, T. Ruh, D. G. A. Smith, J. M. Soler,

D. A. Strubbe, N. Tancogne-Dejean, D. Tildesley, M. Torrent, and V. W.-z. Yu, "The CECAM electronic structure library and the modular software development paradigm," J. Chem. Phys. **153**, 024117 (2020), arXiv:2005.05756.

[106] E. Caldeweyher, C. Bannwarth, and S. Grimme, "Extension of the D3 dispersion coefficient model," J. Chem. Phys. **147**, 034112 (2017).

[107] E. Caldeweyher, S. Ehlert, A. Hansen, H. Neugebauer, S. Spicher, C. Bannwarth, and S. Grimme, "A generally applicable atomic-charge dependent London dispersion correction," J. Chem. Phys. **150**, 154122 (2019).

[108] E. Caldeweyher, J.-M. Mewes, S. Ehlert, and S. Grimme, "Extension and evaluation of the D4 London-dispersion model for periodic systems," Phys. Chem. Chem. Phys. **22**, 8499–8512 (2020).

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

[110] D. G. A. Smith, L. A. Burns, D. A. Sirianni, D. R. Nascimento, A. Kumar, A. M. James, J. B. Schriber, T. Zhang, B. Zhang, A. S. Abbott, E. J. Berquist, M. H. Lechner, L. A. Cunha, A. G. Heide, J. M. Waldrop, T. Y. Takeshita, A. Alenaizan, D. Neuhauser, R. A. King, A. C. Simmonett, J. M. Turney, H. F. Schaefer, F. A. Evangelista, A. E. DePrince, T. D. Crawford, K. Patkowski, and C. D. Sherrill, "PSI4NumPy: An interactive quantum chemistry programming environment for reference implementations and rapid development," J. Chem. Theory Comput. **14**, 3504–3511 (2018).

[111] M. F. Herbst, A. Levitt, and E. Cancès, "DFTK: A Julian approach for simulating electrons in solids," JuliaCon Proceedings **3**, 69 (2021).

[112] S. Lehtola, F. Blockhuys, and C. Van Alsenoy, "An overview of self-consistent field calculations within finite basis sets," Molecules **25**, 1218 (2020), arXiv:1912.12029.

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

[114] L. Talirz, S. Kumbhar, E. Passaro, A. V. Yakutovich, V. Granata, F. Gargiulo, M. Borelli, M. Uhrin, S. P. Huber, S. Zoupanos, C. S. Adorf, C. W. Andersen, O. Schütt, C. A. Pignedoli, D. Passerone, J. VandeVondele, T. C. Schulthess, B. Smit, G. Pizzi, and N. Marzari, "Materials cloud, a platform for open computational science," Sci. Data **7**, 299 (2020).

[115] S. P. Huber, S. Zoupanos, M. Uhrin, L. Talirz, L. Kahle, R. Häuselmann, D. Gresch, T. Müller, A. V. Yakutovich, C. W. Andersen, F. F. Ramirez, C. S. Adorf, F. Gargiulo, S. Kumbhar, E. Passaro, C. Johnston, A. Merkys, A. Cepellotti, N. Mounet, N. Marzari, B. Kozinsky, and G. Pizzi, "AiiDA 1.0, a scalable computational infrastructure for automated reproducible workflows and data provenance," Sci. Data **7**, 300 (2020).

[116] M. Gjerding, T. Skovhus, A. Rasmussen, F. Bertoldo, A. H. Larsen, J. J. Mortensen, and K. S. Thygesen, "Atomic Simulation Recipes: A Python framework and library for automated workflows," Comp. Mater. Sci. **199**, 110731 (2021).

[117] D. G. A. Smith, A. T. Lolinco, Z. L. Glick, J. Lee, A. Alenaizan, T. A. Barnes, C. H. Borca, R. D. Remigio, D. L. Dotson, S. Ehlert, A. G. Heide, M. F. Herbst, J. Hermann, C. B. Hicks, J. T. Horton, A. G. Hurtado, P. Kraus, H. Kruse, S. J. R. Lee, J. P. Misiewicz, L. N. Naden, F. Ramezanghorbani, M. Scheurer, J. B. Schriber, A. C. Simmonett, J. Steinmetzer, J. R. Wagner, L. Ward, M. Welborn, D. Altarawy, J. Anwar, J. D. Chodera, A. Dreuw, H. J. Kulik, F. Liu, T. J. Martínez, D. A. Matthews, H. F. Schaefer, J. Šponer, J. M. Turney, L.-P. Wang, N. D. Silva, R. A. King, J. F. Stanton, M. S. Gordon, T. L. Windus, C. D. Sherrill, and L. A. Burns, "Quantum chemistry common driver and databases (QCDB) and quantum chemistry engine (QCEngine): Automation and interoperability among computational chemistry programs," J. Chem. Phys. **155**, 204801 (2021).

[118] G. Samsonidze and B. Kozinsky, "Half-heusler compounds for use in thermoelectric generators," (U.S. Patent 20170141282, May 2017).

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

[120] E. Strohmaier, H. W. Meuer, J. Dongarra, and H. D. Simon, "The TOP500 list and progress in high-performance computing," Computer **48**, 42–49 (2015).

[121] H. W. Meuer, E. Strohmaier, J. Dongarra, H. Simon, and M. Meuer, "Top500," https://top500.org/, accessed Thu May 20 2021.

[122] A. Szabo and N. S. Ostlund, *Modern quantum chemistry: introduction to advanced electronic structure theory* (Dover Pubns, 1996).

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

[124] S. Pirhadi, J. Sunseri, and D. R. Koes, "Open source molecular modeling," J. Mol. Graph. Model. **69**, 127–143 (2016).

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

[126] L. Talirz, L. M. Ghiringhelli, and B. Smit, "Trends in atomistic simulation software usage," (2021), arXiv:2108.12350 [cond-mat.mtrl-sci].

[127] "Python package index – pypi," https://pypi.org/, accessed 7 July 2021.

[128] Continuum Analytics, "Conda package manager," https://conda.io/, accessed 26 May 2021.

[129] P. Hohenberg and W. Kohn, "Inhomogeneous electron gas," Phys. Rev. **136**, B864–B871 (1964).

[130] W. Kohn and L. J. Sham, "Self-consistent equations including exchange and correlation effects," Phys. Rev. **140**, A1133–A1138 (1965).

[131] S. F. Boys, "Electronic wave functions. I. A general method of calculation for the stationary states of any molecular system," Proc. R. Soc. A Math. Phys. Eng. Sci. **200**, 542–554 (1950).

[132] L. E. McMurchie and E. R. Davidson, "One- and two-electron integrals over cartesian Gaussian functions," J. Comput. Phys. **26**, 218–231 (1978).

[133] S. Obara and A. Saika, "Efficient recursive computation of molecular integrals over cartesian Gaussian functions," J. Chem. Phys. **84**, 3963 (1986).

[134] E. R. Davidson and D. Feller, "Basis set selection for molecular calculations," Chem. Rev. **86**, 681–696 (1986).

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

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

[137] T. Shiozaki, "BAGEL: Brilliantly advanced general electronic-structure library," Wiley Interdiscip. Rev. Comput. Mol. Sci. **8**, e1331 (2018), arXiv:1707.03771.

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

[139] K. Aidas, C. Angeli, K. L. Bak, V. Bakken, R. Bast, L. Boman, O. Christiansen, R. Cimiraglia, S. Coriani, P. Dahle, E. K. Dalskov, U. Ekström, T. Enevoldsen, J. J. Eriksen, P. Ettenhuber, B. Fernández, L. Ferrighi, H. Fliegl, L. Frediani, K. Hald, A. Halkier, C. Hättig, H. Heiberg, T. Helgaker, A. C. Hennum, H. Hettema, E. Hjertenaes, S. Høst, I.-M. Høyvik, M. F. Iozzi, B. Jansík, H. J. A. Jensen, D. Jonsson, P. Jørgensen,

J. Kauczor, S. Kirpekar, T. Kjaergaard, W. Klopper, S. Knecht, R. Kobayashi, H. Koch, J. Kongsted, A. Krapp, K. Kristensen, A. Ligabue, O. B. Lutnaes, J. I. Melo, K. V. Mikkelsen, R. H. Myhre, C. Neiss, C. B. Nielsen, P. Norman, J. Olsen, J. M. H. Olsen, A. Osted, M. J. Packer, F. Pawlowski, T. B. Pedersen, P. F. Provasi, S. Reine, Z. Rinkevicius, T. A. Ruden, K. Ruud, V. V. Rybkin, P. Sałek, C. C. M. Samson, A. S. de Merás, T. Saue, S. P. A. Sauer, B. Schimmelpfennig, K. Sneskov, A. H. Steindal, K. O. Sylvester-Hvid, P. R. Taylor, A. M. Teale, E. I. Tellgren, D. P. Tew, A. J. Thorvaldsen, L. Thøgersen, O. Vahtras, M. A. Watson, D. J. D. Wilson, M. Ziolkowski, and H. Ågren, "The Dalton quantum chemistry program system," Wiley Interdiscip. Rev. Comput. Mol. Sci. **4**, 269–284 (2014).

140E. Rudberg, E. H. Rubensson, P. Sałek, and A. Kruchinina, "Ergo: An open-source program for linear-scaling electronic structure calculations," SoftwareX **7**, 107–111 (2018).

141S. D. Folkestad, E. F. Kjønstad, R. H. Myhre, J. H. Andersen, A. Balbi, S. Coriani, T. Giovannini, L. Goletto, T. S. Haugland, A. Hutcheson, I.-M. Høyvik, T. Moitra, A. C. Paul, M. Scavino, A. S. Skeidsvoll, Å. H. Tveten, and H. Koch, "e^T 1.0: An open source electronic structure program with emphasis on coupled cluster and multilevel methods," J. Chem. Phys. **152**, 184103 (2020), arXiv:2002.05631.

142G. J. R. Aroeira, M. M. Davis, J. M. Turney, and H. F. Schaefer, "Fermi.jl: A modern design for quantum chemistry," J. Chem. Theory Comput. (2022), 10.1021/acs.jctc.1c00719.

143D. Poole, J. L. G. Vallejo, and M. S. Gordon, "A new kid on the block: Application of Julia to Hartree–Fock calculations," J. Chem. Theory Comput. **16**, 5006–5013 (2020).

144F. Bruneval, T. Rangel, S. M. Hamed, M. Shao, C. Yang, and J. B. Neaton, "MOLGW 1: Many-body perturbation theory software for atoms, molecules, and clusters," Comput. Phys. Commun. **208**, 149–161 (2016).

145C. Peng, C. A. Lewis, X. Wang, M. C. Clement, K. Pierce, V. Rishi, F. Pavošević, S. Slattery, J. Zhang, N. Teke, A. Kumar, C. Masteran, A. Asadchev, J. A. Calvin, and E. F. Valeev, "Massively Parallel Quantum Chemistry: A high-performance research platform for electronic structure," J. Chem. Phys. **153**, 044120 (2020).

146R. P. Mueller, "PyQuante: Python quantum chemistry," http://pyquante.sourceforge.net/, accessed 6 July 2021.

147J. P. Unsleber, T. Dresselhaus, K. Klahr, D. Schnieders, M. Böckers, D. Barton, and J. Neugebauer, "Serenity: A subsystem quantum chemistry program," J. Comput. Chem. **39**, 788–798 (2018).

148E. Kjellgren, "SlowQuant," https://github.com/erikkjellgren/SlowQuant, accessed 6 July 2021.

149Z. Rinkevicius, X. Li, O. Vahtras, K. Ahmadzadeh, M. Brand, M. Ringholm, N. H. List, M. Scheurer, M. Scott, A. Dreuw, and P. Norman, "VeloxChem: A Python-driven density-functional theory program for spectroscopy simulations in high-performance computing environments," Wiley Interdiscip. Rev. Comput. Mol. Sci. **10**, e1457 (2019).

150P. Souvatzis, "Uquantchem: A versatile and easy to use quantum chemistry computational software," Comput. Phys. Commun. **185**, 415–421 (2014).

151F. Bloch, "Bemerkung zur Elektronentheorie des Ferromagnetismus und der elektrischen Leitfähigkeit," Z. Phys. **57**, 545–555 (1929).

152P. Kratzer and J. Neugebauer, "The basics of electronic structure theory for periodic systems," Front. Chem. **7**, 1–18 (2019).

153P. Schwerdtfeger, "The pseudopotential approximation in electronic structure theory," ChemPhysChem **12**, 3143–3155 (2011).

154S. Kang, J. Woo, J. Kim, H. Kim, Y. Kim, J. Lim, S. Choi, and W. Y. Kim, "ACE-Molecule: An open-source real-space quantum chemistry package," J. Chem. Phys. **152**, 124110 (2020).

155L. E. Ratcliff, W. Dawson, G. Fisicaro, D. Caliste, S. Mohr, A. Degomme, B. Videau, V. Cristiglio, M. Stella, M. D'Alessandro, S. Goedecker, T. Nakajima, T. Deutsch, and L. Genovese, "Flexibilities of wavelets as a computational basis set for large-scale electronic structure calculations," J. Chem. Phys. **152**, 194110 (2020).

156A. Nakata, J. S. Baker, S. Y. Mujahed, J. T. L. Poulton, S. Arapan, J. Lin, Z. Raza, S. Yadav, L. Truflandier, T. Miyazaki, and D. R. Bowler, "Large scale and linear scaling DFT with the CONQUEST code," J. Chem. Phys. **152**, 164112 (2020), arXiv:2002.07704.

157T. D. Kühne, M. Iannuzzi, M. Del Ben, V. V. Rybkin, P. Seewald, F. Stein, T. Laino, R. Z. Khaliullin, O. Schütt, F. Schiffmann, D. Golze, J. Wilhelm, S. Chulkov, M. H. Bani-Hashemian, V. Weber, U. Borštnik, M. Taillefumier, A. S. Jakobovits, A. Lazzaro, H. Pabst, T. Müller, R. Schade, M. Guidon, S. Andermatt, N. Holmberg, G. K. Schenter, A. Hehn, A. Bussy, F. Belleflamme, G. Tabacchi, A. Glöß, M. Lass, I. Bethune, C. J. Mundy, C. Plessl, M. Watkins, J. VandeVondele, M. Krack, and J. Hutter, "CP2K: An electronic structure and molecular dynamics software package - Quickstep: Efficient and accurate electronic structure calculations," J. Chem. Phys. **152**, 194103 (2020), arXiv:2003.03868.

158"The Elk Code," http://elk.sourceforge.net/.

159A. Gulans, S. Kontur, C. Meisenbichler, D. Nabok, P. Pavone, S. Rigamonti, S. Sagmeister, U. Werner, and C. Draxl, "exciting: a full-potential all-electron package implementing density-functional theory and many-body perturbation theory," J. Phys. Condens. Matter **26**, 363202 (2014).

160"FLEUR," http://www.flapw.de.

161J. Enkovaara, C. Rostgaard, J. J. Mortensen, J. Chen, M. Dułak, L. Ferrighi, J. Gavnholt, C. Glinsvad, V. Haikola, H. A. Hansen, H. H. Kristoffersen, M. Kuisma, A. H. Larsen, L. Lehtovaara, M. Ljungberg, O. Lopez-Acevedo, P. G. Moses, J. Ojanen, T. Olsen, V. Petzold, N. A. Romero, J. Stausholm-Møller, M. Strange, G. A. Tritsaris, M. Vanin, M. Walter, B. Hammer, H. Häkkinen, G. K. H. Madsen, R. M. Nieminen, J. K. Nørskov, M. Puska, T. T. Rantala, J. Schiøtz, K. S. Thygesen, and K. W. Jacobsen, "Electronic structure calculations with GPAW: a real-space implementation of the projector augmented-wave method." J. Phys. Condens. Matter **22**, 253202 (2010).

162R. Sundararaman, K. Letchworth-Weaver, K. A. Schwarz, D. Gunceler, Y. Ozhabes, and T. A. Arias, "JDFTx: Software for joint density-functional theory," SoftwareX **6**, 278–284 (2017).

163Q. Xu, A. Sharma, and P. Suryanarayana, "M-SPARC: Matlab-simulation package for ab-initio real-space calculations," SoftwareX **11**, 100423 (2020).

164N. Tancogne-Dejean, M. J. T. Oliveira, X. Andrade, H. Appel, C. H. Borca, G. Le Breton, F. Buchholz, A. Castro, S. Corni, A. A. Correa, U. De Giovannini, A. Delgado, F. G. Eich, J. Flick, G. Gil, A. Gomez, N. Helbig, H. Hübener, R. Jestädt, J. Jornet-Somoza, A. H. Larsen, I. V. Lebedeva, M. Lüders, M. A. L. Marques, S. T. Ohlmann, S. Pipolo, M. Rampp, C. A. Rozzi, D. A. Strubbe, S. A. Sato, C. Schäfer, I. Theophilou, A. Welden, and A. Rubio, "Octopus, a computational framework for exploring light-driven phenomena and quantum dynamics in extended and finite systems," J. Chem. Phys. **152**, 124119 (2020), arXiv:1912.07921.

165T. Ozaki and H. Kino, "Numerical atomic basis orbitals from H to Kr," Phys. Rev. B **69**, 195113 (2004).

166Y. Saad, J. R. Chelikowsky, and S. M. Shontz, "Numerical Methods for Electronic Structure Calculations of Materials," SIAM Rev. **52**, 3–54 (2010).

167F. Fathurrahman, M. K. Agusta, A. G. Saputro, and H. K. Dipojono, "PWDFT.jl: A Julia package for electronic structure calculation using density functional theory and plane wave basis," Comput. Phys. Commun. **256**, 107372 (2020).

168E. L. Briggs, D. J. Sullivan, and J. Bernholc, "Real-space multigrid-based approach to large-scale electronic structure calculations," Phys. Rev. B **54**, 14362–14375 (1996).

169A. García, N. Papior, A. Akhtar, E. Artacho, V. Blum, E. Bosoni, P. Brandimarte, M. Brandbyge, J. I. Cerdá, F. Corsetti, R. Cuadrado, V. Dikan, J. Ferrer, J. Gale, P. García-

Fernández, V. M. García-Suárez, S. García, G. Huhs, S. Illera, R. Korytár, P. Koval, I. Lebedeva, L. Lin, P. López-Tarifa, S. G. Mayo, S. Mohr, P. Ordejón, A. Postnikov, Y. Pouillon, M. Pruneda, R. Robles, D. Sánchez-Portal, J. M. Soler, R. Ullah, V. W.-z. Yu, and J. Junquera, "Siesta: Recent developments and applications," J. Chem. Phys. **152**, 204108 (2020).

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

[171]Q. Xu, A. Sharma, B. Comer, H. Huang, E. Chow, A. J. Medford, J. E. Pask, and P. Suryanarayana, "SPARC: Simulation package for ab-initio real-space calculations," SoftwareX **15**, 100709 (2021).

[172]S. Lehtola, "A review on non-relativistic, fully numerical electronic structure calculations on atoms and diatomic molecules," Int. J. Quantum Chem. **119**, e25968 (2019), arXiv:1902.01431.

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

[174]R. J. Harrison, G. Beylkin, F. A. Bischoff, J. A. Calvin, G. I. Fann, J. Fosso-Tande, D. Galindo, J. R. Hammond, R. Hartman-Baker, J. C. Hill, J. Jia, J. S. Kottmann, M.-J. Y. Ou, J. Pei, L. E. Ratcliff, M. G. Reuter, A. C. Richie-Halford, N. A. Romero, H. Sekino, W. A. Shelton, B. E. Sundahl, W. S. Thornton, E. F. Valeev, Á. Vázquez-Mayagoitia, N. Vence, T. Yanai, and Y. Yokoi, "MADNESS: A multiresolution, adaptive numerical environment for scientific simulation," SIAM J. Sci. Comput. **38**, S123–S142 (2016).

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

[176]P. Koskinen and V. Mäkinen, "Density-functional tight-binding for beginners," Comput. Mater. Sci. **47**, 237–253 (2009).

[177]G. Seifert and J.-O. Joswig, "Density-functional tight binding—an approximate density-functional theory method," Wiley Interdiscip. Rev.: Comput. Mol. Sci. **2**, 456–465 (2012).

[178]M. Gaus, Q. Cui, and M. Elstner, "Density functional tight binding: application to organic and biological molecules," Wiley Interdiscip. Rev.: Comput. Mol. Sci. **4**, 49–61 (2013).

[179]W. Thiel, "Semiempirical quantum-chemical methods," Wiley Interdiscip. Rev. Comput. Mol. Sci. **4**, 145–157 (2014).

[180]C. Bannwarth, E. Caldeweyher, S. Ehlert, A. Hansen, P. Pracht, J. Seibert, S. Spicher, and S. Grimme, "Extended tight-binding quantum chemistry methods," Wiley Interdiscip. Rev. Comput. Mol. Sci. **11**, e1493 (2021).

[181]B. Hourahine, B. Aradi, V. Blum, F. Bonafé, A. Buccheri, C. Camacho, C. Cevallos, M. Y. Deshaye, T. Dumitrică, A. Dominguez, S. Ehlert, M. Elstner, T. van der Heide, J. Hermann, S. Irle, J. J. Kranz, C. Köhler, T. Kowalczyk, T. Kubař, I. S. Lee, V. Lutsker, R. J. Maurer, S. K. Min, I. Mitchell, C. Negre, T. A. Niehaus, A. M. N. Niklasson, A. J. Page, A. Pecchia, G. Penazzi, M. P. Persson, J. Řezáč, C. G. Sánchez, M. Sternberg, M. Stöhr, F. Stuckenberg, A. Tkatchenko, V. W. Yu, and T. Frauenheim, "DFTB+, a software package for efficient approximate density functional theory based atomistic simulations," J. Chem. Phys. **152**, 124101 (2020).

[182]N. Bock, M. J. Cawkwell, J. D. Coe, A. Krishnapriyan, M. P. Kroonblawd, A. Lang, , C. Liu, E. M. Saez, S. M. Mniszewski, C. F. A. Negre, A. M. N. Niklasson, E. Sanville, M. A. Wood, and P. Yang, "Latte," https://github.com/lanl/LATTE (2008), accessed 12 July 2021.

[183]T. Husch and M. Reiher, "Comprehensive analysis of the neglect of diatomic differential overlap approximation," J. Chem. Theory Comput. **14**, 5169–5179 (2018), arXiv:1806.05615.

[184]I. Cabezas, R. Segovia, P. Caratozzolo, and E. Webb, "Using software engineering design principles as tools for freshman students learning," in *2020 IEEE Frontiers in Education Conference (FIE)* (IEEE, 2020).

[185]P. Lam, J. Dietrich, and D. J. Pearce, "Putting the semantics into semantic versioning," in *Proceedings of the 2020 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software* (ACM, 2020).

[186]E. F. Valeev, "Libint: A library for the evaluation of molecular integrals of many-body operators over gaussian functions," http://libint.valeyev.net/ (2021).

[187]C. L. Lawson, R. J. Hanson, F. T. Krogh, and D. R. Kincaid, "Algorithm 539: Basic linear algebra subprograms for fortran usage [f1]," ACM Trans. Math. Software **5**, 324–325 (1979).

[188]F. G. V. Zee and R. A. van de Geijn, "BLIS: A framework for rapidly instantiating BLAS functionality," ACM Trans. Math. Software **41**, 1–33 (2015).

[189]R. C. Whaley and J. J. Dongarra, "Automatically tuned linear algebra software," in *Proceedings of the IEEE/ACM SC98 Conference* (IEEE, 1998).

[190]N. Flocke and V. Lotrich, "Efficient electronic integrals and their generalized derivatives for object oriented implementations of electronic structure calculations," J. Comput. Chem. **29**, 2722–2736 (2008).

[191]Q. Sun, "Libcint: An efficient general integral library for Gaussian basis functions," J. Comput. Chem. **36**, 1664–1671 (2015), arXiv:1412.0649.

[192]B. P. Pritchard and E. Chow, "Horizontal vectorization of electron repulsion integrals," J. Comput. Chem. **37**, 2537–2546 (2016).

[193]F. Peng, M.-S. Wu, M. Sosonkina, T. Windus, J. Bentz, M. Gordon, J. Kenny, and C. Janssen, "Tackling component interoperability in quantum chemistry software," in *Proceedings of the 2007 symposium on Component and framework technology in high-performance and scientific computing - CompFrame '07* (ACM Press, 2007).

[194]J. P. Kenny, C. L. Janssen, E. F. Valeev, and T. L. Windus, "Components for integral evaluation in quantum chemistry," J. Comput. Chem. **29**, 562–577 (2008).

[195]U. Ekström, L. Visscher, R. Bast, A. J. Thorvaldsen, and K. Ruud, "Arbitrary-Order Density Functional Response Theory from Automatic Differentiation," J. Chem. Theory Comput. **6**, 1971–1980 (2010).

[196]S. Wouters, W. Poelmans, P. W. Ayers, and D. Van Neck, "CheMPS2: A free open-source spin-adapted implementation of the density matrix renormalization group for ab initio quantum chemistry," Comput. Phys. Commun. **185**, 1501–1514 (2014), arXiv:1312.2415.

[197]M. Scheurer, P. Reinholdt, E. R. Kjellgren, J. M. H. Olsen, A. Dreuw, and J. Kongsted, "CPPE: An open-source C++ and Python library for polarizable embedding," J. Chem. Theory Comput. **15**, 6154–6163 (2019).

[198]S. Grimme, J. Antony, S. Ehrlich, and H. Krieg, "A consistent and accurate ab initio parametrization of density functional dispersion correction (DFT-D) for the 94 elements H-Pu," J. Chem. Phys. **132**, 154104 (2010).

[199]I. A. Kaliman and L. V. Slipchenko, "LIBEFP: A new parallel implementation of the effective fragment potential method as a portable software library," J. Comput. Chem. **34**, 2284–2292 (2013).

[200]R. D. Remigio, L. Frediani, A. H. Steindal, R. Bast, L. A. Burns, T. D. Crawford, V. Weijo, and Wpoely86, "PCMSolver, an open-source library for the polarizable continuum model electrostatic problem," https://github.com/PCMSolver/pcmsolver (2020), accessed Feb 2 2021.

[201]B. P. Pritchard, D. Altarawy, B. Didier, T. D. Gibson, and T. L. Windus, "New Basis Set Exchange: An open, up-to-date resource for the molecular sciences community," J. Chem. Inf. Model. **59**, 4814–4820 (2019).

[202]R. Shaw and J. Hill, "libecpint: A c++ library for the efficient evaluation of integrals over effective core potentials," Journal of Open Source Software **6**, 3039 (2021).

[203]"Jmol: an open-source Java viewer for chemical structures in 3D. http://www.jmol.org,".

[204]M. D. Hanwell, D. E. Curtis, D. C. Lonie, T. Vandermeersch, E. Zurek, and G. R. Hutchison, "Avogadro: an advanced semantic chemical editor, visualization, and analysis platform," J. Cheminf. **4**, 17 (2012).

[205]A. Gilbert, "Iqmol, a free open-source molecular editor and visualization package." http://iqmol.org, accessed June 26 2021.

[206]Schrödinger Inc., "Pymol, a molecular visualization system," https://github.com/schrodinger/pymol-open-source, accessed 6 July 2021.

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

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

[209]A. H. Larsen, J. J. Mortensen, J. Blomqvist, I. E. Castelli, R. Christensen, M. Dułak, J. Friis, M. N. Groves, B. Hammer, C. Hargus, E. D. Hermes, P. C. Jennings, P. B. Jensen, J. Kermode, J. R. Kitchin, E. L. Kolsbjerg, J. Kubal, K. Kaasbjerg, S. Lysgaard, J. B. Maronsson, T. Maxson, T. Olsen, L. Pastewka, A. Peterson, C. Rostgaard, J. Schiøtz, O. Schütt, M. Strange, K. S. Thygesen, T. Vegge, L. Vilhelmsen, M. Walter, Z. Zeng, and K. W. Jacobsen, "The atomic simulation environment—a Python library for working with atoms," J. Phys.: Condens. Matter **29**, 273002 (2017).

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

[211]T. Lu and F. Chen, "Multiwfn: A multifunctional wavefunction analyzer," J. Comput. Chem. **33**, 580–592 (2011).

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

[213]S. Lehtola and A. J. Karttunen, "git repository containing a copy of the supporting information," https://github.com/susilehtola/fosschemistry, accessed 8 August 2021.

[214]C. Bannwarth, S. Ehlert, and S. Grimme, "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. **15**, 1652–1671 (2019).

[215]J. P. Menzel, M. Kloppenburg, J. Belić, H. J. M. Groot, L. Visscher, and F. Buda, "Efficient workflow for the investigation of the catalytic cycle of water oxidation catalysts: Combining GFN-xTB and density functional theory," J. Comput. Chem. **42**, 1885–1894 (2021).

[216]N. Tasinato, C. Puzzarini, and V. Barone, "Correct modeling of cisplatin: a paradigmatic case," Angew. Chem. - Int. Ed. **56**, 13838–13841 (2017).

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

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

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

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

[221]M. Ernzerhof and G. E. Scuseria, "Assessment of the Perdew–Burke–Ernzerhof exchange-correlation functional," J. Chem. Phys. **110**, 5029–5036 (1999).

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

[223]M. Bühl, C. Reimann, D. A. Pantazis, T. Bredow, and F. Neese, "Geometries of third-row transition-metal complexes from density-functional theory," J. Chem. Theory Comput. **4**, 1449–1459 (2008).

[224]W. J. Hehre, R. F. Stewart, and J. A. Pople, "Self-consistent molecular-orbital methods. I. Use of Gaussian expansions of Slater-type atomic orbitals," J. Chem. Phys. **51**, 2657 (1969).

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

[226]W. J. Hehre, R. Ditchfield, and J. A. Pople, "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. **56**, 2257–2261 (1972).

[227]F. Weigend and R. Ahlrichs, "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. **7**, 3297–305 (2005).

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

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

[230]M. Dolg, "Chapter 14 relativistic effective core potentials," in *Theoretical and Computational Chemistry* (Elsevier, 2002) pp. 793–862.

[231]D. Rappoport and F. Furche, "Property-optimized gaussian basis sets for molecular response calculations." J. Chem. Phys. **133**, 134105 (2010).

[232]S. Ehlert, U. Huniar, J. Ning, J. W. Furness, J. Sun, A. D. Kaplan, J. P. Perdew, and J. G. Brandenburg, "r^2SCAN-D4: Dispersion corrected meta-generalized gradient approximation for general chemical applications," J. Chem. Phys. **154**, 061101 (2021).

[233]C. Møller and M. S. M. Plesset, "Note on an approximation treatment for many-electron systems," Phys. Rev. **46**, 618–622 (1934).

[234]J. Čížek, "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. **45**, 4256–4266 (1966).

[235]S. Grimme, "Exploration of chemical compound, conformer, and reaction space with meta-dynamics simulations based on tight-binding quantum chemical calculations," J. Chem. Theory Comput. **15**, 2847–2862 (2019).

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

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

[238]J. L. Whitten, "Coulombic potential energy integrals and approximations," J. Chem. Phys. **58**, 4496 (1973).

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

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

[241]B. I. Dunlap, J. W. D. Connolly, and J. R. Sabin, "On some approximations in applications of Xα theory," J. Chem. Phys. **71**, 3396 (1979).

[242]B. I. Dunlap, N. Rösch, and S. B. Trickey, "Variational fitting methods for electronic structure calculations," Mol. Phys. **108**, 3167–3180 (2010).

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

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

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

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

[247] M. Bühl and H. Kabrede, "Geometries of transition-metal complexes from density-functional theory," J. Chem. Theory Comput. **2**, 1282–1290 (2006).

[248] S. Grimme, C. Bannwarth, and P. Shushkov, "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. **13**, 1989–2009 (2017).

[249] P. A. M. Dirac, "Note on exchange phenomena in the Thomas atom," Math. Proc. Cambridge Philos. Soc. **26**, 376–385 (1930).

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

[251] J. W. Furness, A. D. Kaplan, J. Ning, J. P. Perdew, and J. Sun, "Accurate and numerically efficient r^2SCAN meta-generalized gradient approximation," J. Phys. Chem. Lett. **11**, 8208–8215 (2020).

[252] J. W. Furness, A. D. Kaplan, J. Ning, J. P. Perdew, and J. Sun, "Correction to "Accurate and numerically efficient r^2SCAN meta-generalized gradient approximation"," J. Phys. Chem. Lett. **11**, 9248–9248 (2020).

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

[254] W. J. Pietro and W. J. Hehre, "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. **4**, 241–251 (1983).

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

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

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

[258] P. J. Gardner and P. Pang, "Thermodynamics of the zinc sulphide transformation, sphalerite $\rightarrow$ wurtzite, by modified entrainment," J. Chem. Soc., Faraday Trans. 1 **84**, 1879 (1988).

[259] K. F. Garrity, J. W. Bennett, K. M. Rabe, and D. Vanderbilt, "Pseudopotentials for high-throughput DFT calculations," Comput. Mater. Sci. **81**, 446–452 (2014), arXiv:1305.5973.

[260] C. Team, "Crystallography open database," http://www.crystallography.net/cod/, accessed 20 July 2021.

[261] Materials Cloud, "Quantum espresso input generator and structure visualizer," https://www.materialscloud.org/work/tools/qeinputgenerator (), accessed 20 July 2021.

[262] H. J. Monkhorst and J. D. Pack, "Special points for Brillouin-zone integrations," Phys. Rev. B **13**, 5188–5192 (1976).

[263] M. Cardona, R. K. Kremer, R. Lauck, G. Siegle, A. Muñoz, A. H. Romero, and A. Schindler, "Electronic, vibrational, and thermodynamic properties of ZnS with zinc-blende and rocksalt structure," Phys. Rev. B **81**, 075207 (2010).

[264] Materials Cloud, "Seek-path: the k-path finder and visualizer," https://www.materialscloud.org/work/tools/seekpath (), accessed 20 July 2021.

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

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

[267] A. R. McDonald, J. A. Nash, P. S. Nerenberg, K. A. Ball, O. Sode, J. J. Foley, T. L. Windus, and T. D. Crawford, "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. **120**, e26359 (2020).

[268] D. B. Magers, V. H. Chávez, B. G. Peyton, D. A. Sirianni, R. C. Fortenberry, and A. Ringer McDonald, "PSI4EDUCATION: Free and open-source programing activities for chemical education with free and open-source software," in *Teaching Programming across the Chemistry Curriculum* (2021) Chap. 8, pp. 107–122.