Gottfried Wilhelm Leibniz Universität Hannover
Faculty of Natural Sciences
Institute of Technical Chemistry

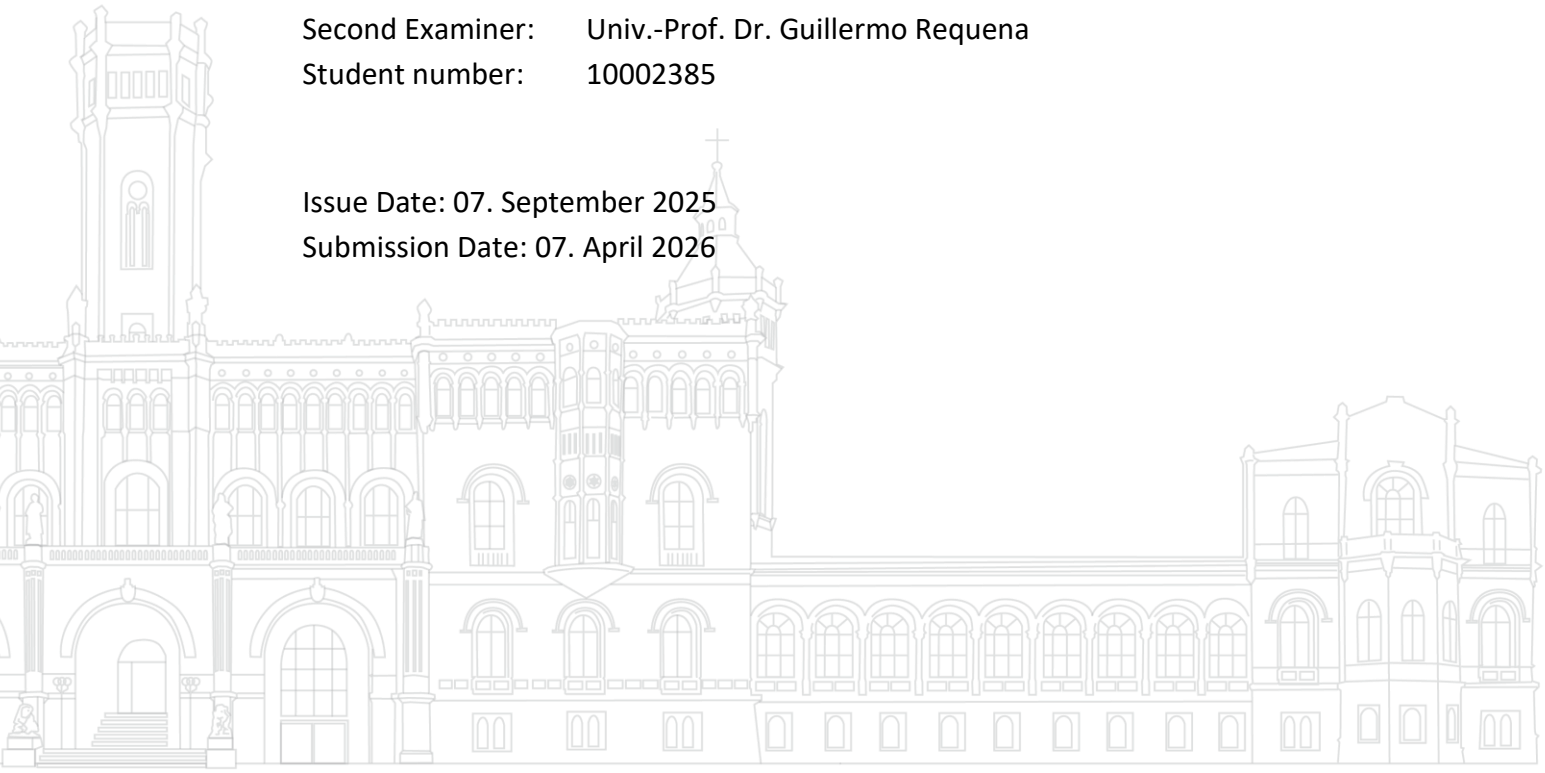
Calculation of Bulk Moduli using Many-Body Hamiltonians and Quantum Algorithms

Master's Thesis
in Chemistry
by Nico Hagemann

This thesis was conducted in collaboration with the German Aerospace Center (DLR), Institute for Frontier Materials on Earth and in Space.

First Examiner:	Prof. Dr.-Ing. Selin Kara
Second Examiner:	Univ.-Prof. Dr. Guillermo Requena
Student number:	10002385

Issue Date: 07. September 2025
Submission Date: 07. April 2026



Acknowledgments

At this point, I wish to thank everyone whose support and encouragement contributed to the success of this work. First and foremost, I would like to thank Prof. Dr.-Ing. Selin Kara and Dr. Patrick Lindner for the opportunity to conduct this research. I also wish to thank Univ.-Prof. Dr. Guillermo Requena for serving as the second examiner for this work.

This thesis was conducted in collaboration with the German Aerospace Center (DLR), specifically the Institute for Frontier Materials on Earth and in Space. I am deeply grateful for the kindness and the stimulating research environment provided by the DLR team. A special thank you goes to Erik Schultheis and Gabriel Breuil for their exceptional guidance and support.

Finally, I would like to thank my family and friends for their unwavering support throughout my studies.

Abstract

This Master's thesis investigates a many-body post-processing approach for calculating the ground-state properties of crystalline solids based on electron orbitals obtained from density functional theory (DFT) calculations. This approach aims to enable near-term quantum computers to generate useful results that cannot be computed classically while addressing DFT's weakness in calculations for strongly correlated systems. To validate this approach, bulk moduli for a variety of structures were calculated, revealing systematic overbinding, the origin of which could not be clearly determined. Our openly available computational framework, called Dopyqo, was used for these many-body calculations. Furthermore, in addition to the commonly used unitary coupled cluster ansatz with single and double excitations (UCCSD), the more hardware-efficient local unitary cluster Jastrow (LUCJ) ansatz was implemented into Dopyqo, and its improved scalability for the variational quantum eigensolver (VQE) on near-term quantum hardware was demonstrated.

Table of Contents

List of Abbreviations

BM	Birch-Murnaghan
DFA	density-functional approximation
DFT	density functional theory
EoS	equation of state
FCI	full configuration interaction
GGA	generalized gradient approximation
GUI	graphical user interface
HF	Hartree-Fock
JW	Jordan-Wigner
KS	Kohn-Sham
LDA	local-density approximation
LUCJ	local unitary cluster Jastrow
NISQ	noisy intermediate-scale quantum
PBE	Perdew-Burke-Ernzerhof
PP	pseudopotential
QE	Quantum ESPRESSO
qubits	quantum bits
regex	regular expressions
RMSE	root mean square error
SCF	self-consistent field
UCCSD	unitary coupled cluster with single and double excitations
UCJ	unitary cluster Jastrow
VQE	variational quantum eigensolver

List of Figures

List of Tables

List of Code Snippets

1 Introduction

Methods that can generate results without incorporating experimental references are generally referred to as *ab initio* (Latin: “from the beginning”)[1]. These methods have been advancing in recent years based on research and evolving computational hardware[1]. Especially in materials science, *ab initio* calculations hold great potential. The ability to predict material properties solely based on calculations could lead to the discovery of new materials, as well as a general speedup in materials research[2]. The combination of experimental and computational methods can drastically reduce development costs and time, as seen in the aerospace industry, where fluid dynamics calculations have mostly replaced expensive wind tunnel tests[3, 4]. In addition, advancements in the field of *ab initio* calculations play an essential role in understanding interactions at an atomic level and their influence on macroscopic quantities[2, 4, 5]. One of the main challenges of *ab initio* calculations lies in the electronic structure problem, i.e., finding the ground-state energy of chemical systems. Although classical computational chemistry is a well-established discipline, achieving precise and scalable results for strongly correlated systems (characterized by strong electron-electron interactions where mean-field approximations fail) continues to be a significant unresolved issue, as highly sophisticated calculations are only classically tractable for very small systems[2, 6]. Currently, density functional theory (DFT) is the most widely used framework for these tasks due to its computational efficiency. However, DFT approximations often struggle to describe strongly correlated systems. To address these limitations, many-body post-processing methods and the emerging field of quantum computing offer promising alternatives. This work contributes to the development of a method for calculating the ground state of crystalline solids using a many-body approach based on electron orbitals calculated by DFT. This approach was developed as a potential application for near-term quantum hardware. The primary objective of this thesis is to validate this many-body approach by performing *ab initio* calculations of the bulk modulus, which is a measure of a material's inherent resistance to uniform compression.

This thesis is organized as follows. Chapter 2 gives an overview of the chemical, physical, and computational bases of this work. This includes a review of crystal structures, the electronic structure problem, density functional theory, and the fundamentals of quantum computing and hardware. Then, in Chapter 3, a new Python module for the calculation of bulk moduli is implemented into the computational framework Dopyqo. Using this module, a comprehensive study of many-body post-processing based on density functional theory calculations is carried out. In Chapter 4, a more resource-efficient ansatz for the variational quantum eigensolver (VQE) is implemented into Dopyqo and its performance is evaluated. This work concludes with a discussion and interpretation of the research results, as well as an outlook on the future of *ab initio* calculations for crystalline solids in Chapter 5. Through these investigations, the thesis aims to bridge the gap between physical intuition and hardware efficiency, contributing to the development of robust quantum algorithms for near-term materials research.

2 Theoretical Background

This section establishes the theoretical foundation for the methods and concepts utilized throughout this work. First, Section 2.1 introduces the fundamental properties of crystal structures and their representation in reciprocal space. This is followed by a description of the electronic structure problem in Section 2.2, which forms the basis for finding ground-state energies. Section 2.3 discusses density functional theory as a computationally tractable framework for large systems, while also addressing its inherent limitations in accurately describing strongly correlated materials. To overcome these challenges, the concepts of second quantization and many-body post-processing are explained in Section 2.4. Next, Section 2.5 gives an introduction to quantum computing and the variational quantum eigensolver, which potentially serves as the future computational platform for the many-body calculations explored in this thesis. Finally, the chapter concludes in Section 2.6 with an overview of current quantum hardware. In this work, vectors are written in bold font ($\boldsymbol{v}$) and the Fourier transform of a function $f(\cdot)$ is defined as $\tilde{f}(\cdot)$.

2.1 Crystal Structures

The information contained in this chapter originates from reference [7]. A crystal is characterized by a periodic arrangement of atoms, held together by a bonding force. This system is described by a three-dimensional periodic array of identical building blocks called unit cells. Each unit cell contains an identical group of atoms called the basis. The vertices of the unit cell span a set of mathematical points to which the basis is attached, called the lattice. The mathematical basis of the lattice is defined by the lattice translation vectors, connecting the vertices of the unit cell. Figure 2.1.1 shows a unit cell within a periodic system with the lattice translation vectors $\mathbf{a}_1$, $\mathbf{a}_2$, $\mathbf{a}_3$.

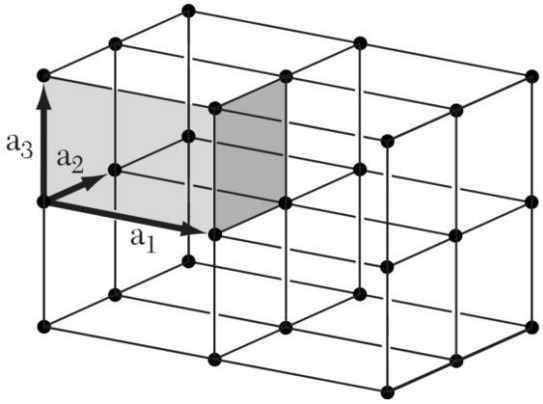


Figure 2.1.1: Primitive cell of a space lattice in three dimensions[7].

In theory, the unit cell can be chosen freely as long as it can fill the entire space completely without overlaps by translation, so that the periodic crystal structure is formed. A unit cell with the smallest possible volume is called a primitive unit cell. Since these tend to be less intuitive for visualizing symmetries, many structures are traditionally shown in the form of a larger conventional unit cell.

Much larger unit cells are called supercells, which are often constructed by incorporating multiple conventional cells.

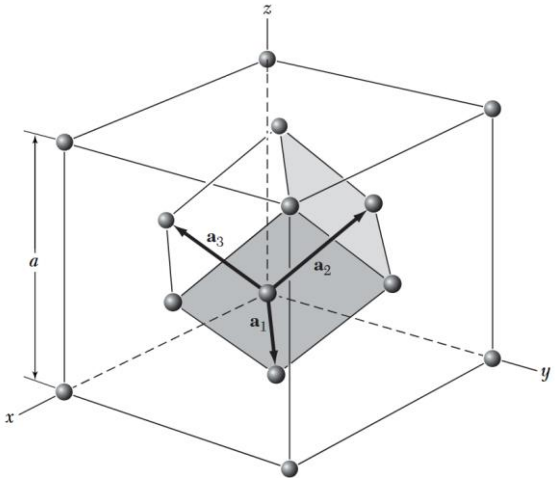


Figure 2.1.2: Rhombohedral primitive cell within the conventional face-centered cubic crystal structure cell. The primitive translation vectors $\mathbf{a}_1$, $\mathbf{a}_2$, $\mathbf{a}_3$ *are shown*[7].

Based on their symmetry, crystal structures are divided into seven different crystal systems, which can form different centerings, resulting in 14 structures called Bravais lattices. Figure 2.1.2 shows a face-centered cubic (fcc) structure, a subgroup of the cubic crystal system. The conventional cell clearly visualizes a cubic symmetry, unlike the primitive cell contained within it.

A crystal's lattice uniquely defines its reciprocal lattice. It is built using reciprocal lattice vectors that are derived from the real-space axes to describe the underlying periodicity of the electron density of a material. The permissible reciprocal lattice vectors $\boldsymbol{G}_i$ are defined by the lattice transitional vectors $\mathbf{a}_i$ via

$$\boldsymbol{G}_i \cdot \mathbf{a}_i = 2\pi m\,, \tag{2.1.1}$$

where m is a positive integer. The reciprocal space is a mathematical domain where dimensions are measured in units of inverse length. The primitive cell of the reciprocal lattice is called the Brillouin zone, which acts as the fundamental cell for wavevectors in the study of quantum mechanics within solids. For further explanation and derivation of this concept, see [8]. For a periodic system, all information about electrons and atoms is contained within the Brillouin zone[8]. Properties such as total energy or electron density are obtained by integration over all points of this volume, called k-points. Since a continuous integral over the Brillouin zone is numerically intractable, it is approximated by a weighted sum over a discrete grid of evenly distributed k-points.

2.2 Electronic Structure Problem

A quantum-mechanical system, such as a periodic crystal in this case can be described completely by a real-space wave function $\psi(\boldsymbol{r})$[9]. Each measurable quantity is defined according to an eigenvalue equation of the form

$$A\psi(\boldsymbol{r}) = a\psi(\boldsymbol{r}) , \tag{2.2.1}$$

where A is the linear operator of the corresponding measured quantity and $\psi(\boldsymbol{r})$ and a are an associated eigenfunction and eigenvalue, respectively. An exceptionally important measure and the focus of this work is the energy E of a system, which is defined by the Schrödinger equation

$$H\psi(\boldsymbol{r}) = E\psi(\boldsymbol{r}) . \tag{2.2.2}$$

Here, H is the Hamilton operator, called the Hamiltonian[5]. For a system consisting of I nuclei and i electrons, the Hamiltonian is defined as

$$H = -\sum_i \frac{\nabla_i^2}{2} - \sum_I \frac{\nabla_I^2}{2M_I'} - \sum_{i,I} \frac{Z_I}{|\boldsymbol{r}_i - \boldsymbol{R}_I|} + \frac{1}{2}\sum_{i\neq j} \frac{1}{|\boldsymbol{r}_i - \boldsymbol{r}_j|} + \frac{1}{2}\sum_{I\neq J} \frac{Z_I Z_J}{|\boldsymbol{R}_I - \boldsymbol{R}_J|} , \tag{2.2.3}$$

where $M_I' = \frac{M_I}{m_e}$ is the atomic mass divided by the mass of an electron and $\boldsymbol{R}_I$ and Z_I are the position and atomic number of the I-th nucleus. The position of the i-th electron is denoted by $\boldsymbol{r}_i$. Here, we work in atomic units, where lengths are in Bohr, masses are in electron masses, and energies are in Hartree. The first two sums of Equation (2.2.3) contain the kinetic energy of electrons and nuclei, respectively. The latter three sums contain the Coulomb interactions between the electrons and nuclei, the electrons themselves and the nuclei themselves. This equation can be simplified based on the Born-Oppenheimer approximation[10], which assumes the nuclei to be static point charges. This assumption is based on the fact that the nuclei are over 1,000 times heavier than the electrons and that their movement is accordingly slower. The Born-Oppenheimer approximation is remarkably accurate for the vast majority of problems in solid-state physics and allows for the coupling between nuclear and electronic motion to be neglected[1, 2, 11]. This leads to a simplification of Equation (2.2.3), where only the electronic Hamiltonian

$$H = -\sum_i \frac{\nabla_i^2}{2} - \sum_{i,I} \frac{Z_I}{|\boldsymbol{r}_i - \boldsymbol{R}_I|} + \frac{1}{2}\sum_{i\neq j} \frac{1}{|\boldsymbol{r}_i - \boldsymbol{r}_j|} + \frac{1}{2}\sum_{I\neq J} \frac{Z_I Z_J}{|\boldsymbol{R}_I - \boldsymbol{R}_J|} \tag{2.2.4}$$

needs to be addressed. Solving Equation (2.2.2) means finding the wave functions $\psi_i(r)$ (eigenstates) and the corresponding energy eigenvalues E_i, i.e., finding all possible states of the system and the total energy of all these states. This solution gives access to the electronic structure of the material, which allows for the calculation of a series of relevant real-world metrics[12-15]. These include, for example, band structures, excitation energies, bond strengths, and bond lengths. This work focuses on finding the ground-state energy, which is the basis for calculating the bulk modulus of a material. This energy is defined as

$$E = \min_{\psi} \int \psi^*(\boldsymbol{r}_1, \dots, \boldsymbol{r}_N) H \psi(\boldsymbol{r}_1, \dots, \boldsymbol{r}_N) d\boldsymbol{r} , \tag{2.2.5}$$

which defines the minimization over all antisymmetric N-particle wave functions $\psi(\boldsymbol{r}_1, \dots, \boldsymbol{r}_N)$ [2, 11, 16]. Solving this equation is exponentially costly with growing size of the system and can only be solved exactly for small molecules by classical computers. This is called the electronic structure problem, for which many different approximations exist. The historical and conceptual basis of many of these approximations is Hartree-Fock (HF) theory[16, 17]. In this approximation, each electron is described by an orbital and the total wave function $\psi(\boldsymbol{r}_1, \dots, \boldsymbol{r}_N)$ results from the product of all orbitals[1, 2, 16, 18]. Since fermions are indistinguishable, $\psi(\boldsymbol{r}_1, \dots, \boldsymbol{r}_N)$ must be antisymmetric upon interchanging two

electrons to obey the Pauli principle. This is achieved by assembling one-electron orbitals into a Slater determinant, forming the total wave function. Accordingly, the aim of HF theory is finding the Slater determinant, respectively the combination of one-electron orbitals with the lowest overall energy, i.e. the ground state. This depiction as one-electron orbitals is made possible by applying a mean-field approximation, where only the average electron-electron interaction is taken into account. As a result, correlation between electrons is neglected. Advanced methods including electron correlation require multideterminant wave functions, which are computationally much more demanding[16, 19]. A ground state described by a single Slater determinant is generally called an HF state. Although HF theory does not always generate accurate results, it is the starting point for many further approaches, such as Configuration Interaction[20] or Coupled Cluster[21].

2.3 Density Functional Theory

Density functional theory (DFT) can be seen as an improvement upon HF theory[1]. The main difference compared to HF is that in DFT, the total electron density instead of the individual wave functions is considered. A wave function of an N-electron system is a function of $3N$ spatial and N spin coordinates. In contrast, the electron density, derived by integrating the squared wave function over the coordinates of $N-1$ electrons, depends only on three spatial coordinates, regardless of the total number of electrons. Consequently, while the complexity of the wave function (and the exact solution of the Schrödinger equation) scales exponentially with the system size, the electron density remains a function of a constant number of variables, providing a much more computationally tractable framework for large systems[1]. The fundamental challenge in DFT is that while it has been proven that every electron density uniquely corresponds to a specific ground-state energy, the exact functional linking these two remains unknown. Consequently, the primary focus of DFT methods is to engineer functionals that can accurately connect the electron density to a system's total energy.

The starting point of modern DFT is the aforementioned proof by Hohenberg and Kohn[22] that all properties determined by the Hamiltonian H are also determined by the ground-state electron density $n(\boldsymbol{r})$. They were able to show that the ground-state electron density uniquely determines the external potential $v(\boldsymbol{r})$ such that

$$V \equiv \sum_i v(\boldsymbol{r}_i) \tag{2.3.1}$$

and since the integration of the electron density $n(\boldsymbol{r})$ determines the number of electrons N, the kinetic energy

$$T \equiv -\sum_i \frac{\nabla_i^2}{2} \tag{2.3.2}$$

and the interelectronic repulsion

$$U = \frac{1}{2}\sum_{i \neq j} \frac{1}{|\boldsymbol{r}_i - \boldsymbol{r}_j|} \tag{2.3.3}$$

are also determined. Accordingly, the complete Hamiltonian is

$$H \equiv T + V + U\,. \tag{2.3.4}$$

Consequently, Hohenberg and Kohn's proof states that all necessary information about the system is contained in $n(\boldsymbol{r})$[1, 23]. The number of electrons is defined by the integrated density, the positions of the nuclei are defined by the cusps in the density, and the nuclear charges are contained in the heights of these cusps. Although it has been proven that each different density yields a different ground-state energy, the functional connecting these two quantities is unknown[1].

Using approximations, an equation can be derived which describes the ground-state energy as a functional of the electron density[16]. For this, first, equations (2.3.1), (2.3.2) and (2.3.3) need to be converted into a dependency of $n(\boldsymbol{r})$. This was first done by Thomas[24] and Fermi[25], who approximated the kinetic energy term by that of a uniform electron gas of noninteracting electrons. For a spin-unpolarized system

$$T[n] = a_s \int d^3\boldsymbol{r}\, n^{\frac{5}{3}}(\boldsymbol{r})\,, \qquad a_s = \frac{3(3\pi^2)^{\frac{2}{3}}}{10} \tag{2.3.5}$$

is the resulting equation. The interelectronic repulsion functional $U[n]$ is approximated by the Hartree energy

$$U[n] = \frac{1}{2}\int d^3\boldsymbol{r} \int d^3\boldsymbol{r}'\, \frac{n(\boldsymbol{r})n(\boldsymbol{r}')}{|\boldsymbol{r}-\boldsymbol{r}'|} \tag{2.3.6}$$

and the one-body potential is defined as

$$V[n] = \int d^3\boldsymbol{r}\; n(\boldsymbol{r})\, v(\boldsymbol{r})\,. \tag{2.3.7}$$

Even though DFT is, in itself, an exact theory, the approximations made to make it tractable result in inaccuracies, which led to a crucial step in developing modern DFT[16, 26]. By reintroducing orbitals to the theory, Kohn and Sham[27] made the exact density functional more accessible for numerical solutions[16]. The basis of their work is the consideration of a fictitious set of noninteracting electrons, which is defined to have the same electron density $n(\boldsymbol{r})$ as the underlying interacting problem. These Kohn-Sham (KS) electrons satisfy a noninteracting Schrödinger equation

$$\left\{-\frac{1}{2}\nabla^2 + v_{S,\sigma}(\boldsymbol{r})\right\}\phi_{i\sigma}(\boldsymbol{r}) = \varepsilon_{i\sigma}\phi_{i\sigma}(\boldsymbol{r})\,, \tag{2.3.8}$$

in which $\phi_{i\sigma}(\boldsymbol{r})$ is the KS orbital of the i-th electron with spin σ, $\varepsilon_{i\sigma}$ is the corresponding KS eigenvalue and $v_{S,\sigma}(\boldsymbol{r})$ is the KS potential, which accounts for all missing potentials. It is defined as

$$v_{S,\sigma}(\boldsymbol{r}) = v_\sigma(\boldsymbol{r}) + v_{\mathrm{H}}(\boldsymbol{r}) + v_{\mathrm{XC},\sigma}(\boldsymbol{r}), \tag{2.3.9}$$

where $v_\sigma(\boldsymbol{r})$ is the one-body potential, which consists of the external potential defined in Equation (2.3.1) and potentially an interaction with an external magnetic field[27, 28]. In the case of this thesis, no external magnetic fields are included. Therefore, $v_\sigma(\boldsymbol{r})$ contains only the external potential and is spin-independent[28]. $v_{\mathrm{H},\sigma}(\boldsymbol{r})$ is the Hartree potential, written as a functional derivative of $U[n]$

$$v_{\mathrm{H},\sigma}(\boldsymbol{r}) = \frac{\delta U[n]}{\delta n_\sigma(\boldsymbol{r})} = \int d^3\boldsymbol{r}'\, \frac{n_\sigma(\boldsymbol{r}')}{|\boldsymbol{r}-\boldsymbol{r}'|} \tag{2.3.10}$$

and $v_{\mathrm{XC},\sigma}(\boldsymbol{r})$ is the exchange-correlation potential

$$v_{XC,\sigma}(\boldsymbol{r}) = \frac{\delta E_{\mathrm{XC}}[n]}{\delta n_{\sigma}(\boldsymbol{r})}. \tag{2.3.11}$$

The exchange-correlation energy $E_{XC}[n]$ acts as a correction, accounting for three key physical phenomena[16, 23, 27]. First, it eliminates self-interaction, which is a mathematical artifact where an electron repels its own charge density due to the formulation of the Hartree energy. Second, it models the Fermi hole (exchange), which arises from the Pauli exclusion principle. This principle enforces the antisymmetry of the wave function, resulting in a statistical depletion of probability density (the Fermi hole) for electrons of the same spin being near each other. Third, it describes the Coulomb hole (correlation), which is the dynamic avoidance between all electrons due to their electrostatic repulsion[1]. Up to this point, this approach would yield the exact ground-state energy.

Two types of orbitals must be distinguished here[16, 28]. Depending on whether a spin-polarized or unpolarized system is present, either spin-orbitals or spatial orbitals and densities are used. Spatial orbitals are used for unpolarized systems, where each orbital may be occupied by two electrons of opposite spin. For spin-polarized systems, each orbital can contain only one electron and two electron densities, $n_{\uparrow}(\boldsymbol{r})$ and $n_{\downarrow}(\boldsymbol{r})$, are used. This is more accurate for systems with an odd number of electrons and allows for the treatment of collinear magnetic fields. The equations shown in (2.3.8), (2.3.9), and (2.3.11) onwards belong to a spin-polarized system. For an unpolarized system, these calculations are simplified due to the elimination of spin differentiation. It should be noted that the Hartree potential in (2.3.10) is spin-independent[28], but is depicted as a function of spin specific electron densities according to the underlying spin-polarized system.

The main advantage of using KS orbitals is that the kinetic energy can be accounted for without approximations via[16, 27]

$$T(\boldsymbol{r}) = -\frac{1}{2}\sum_{\sigma}\sum_{i}^{N_{\sigma}} \int d^3\boldsymbol{r}\, \phi_{i\sigma}^{*}(\boldsymbol{r})\, \nabla^2\, \phi_{i\sigma}(\boldsymbol{r})\,. \tag{2.3.12}$$

In terms of KS quantities, the total energy is expressed as

$$E[n] = T[n] + U[n] + V[n] + E_{\mathrm{XC}}[n]\,. \tag{2.3.13}$$

Since the exchange-correlation functional is unknown, it must be approximated[29]. The accuracy of KS-DFT is therefore solely based on the quality of the density-functional approximation (DFA), accounting for the exchange-correlation potential. For that reason, a wide range of DFAs have been constructed and improved over the years. The DFAs are classified into five rungs of Jacob's ladder, a conceptual hierarchy of exchange–correlation approximations, ordered by increasing physical complexity and accuracy, but also by increasing computational cost[30, 31]. The first rung is represented by the local-density approximation (LDA), which considers only the density itself[32]. For that reason, LDA is successful for systems with slowly varying electron densities, such as nearly free-electron metals. In addition to its limited accuracy compared to other DFAs, it generally tends to struggle with overbinding, i.e., overestimating the bond strength, which has a great influence on multiple quantities, including the bulk modulus. The next rung on the ladder is occupied by functionals based on the generalized gradient approximation (GGA), which extends the approach of LDA by introducing the

gradient of the electron density[33]. In contrast to LDAs, GGAs tend toward underbinding. One specific GGA, the Perdew-Burke-Ernzerhof (PBE) functional, is one of the most commonly used DFAs and also the DFA of choice for this work. The meta-GGAs on the third rung additionally introduce the kinetic energy density to the functional, further improving accuracy. Since the weaknesses of the first three rungs mainly lie in self-interaction errors and the lack of long-range van der Waals interactions, the DFAs of the fourth and fifth rungs specifically target these effects via hybrid functionals and the random-phase approximation[29, 34, 35]. Rungs 3 to 5 are not relevant to this work and they serve only as a brief mention for the sake of completeness.

In solids, it is common to make another approximation in addition to the DFAs, by treating the core electrons as "frozen"[1, 36]. This is achieved using pseudopotentials (PPs), which replace the nucleus and core electrons with an effective potential that substitutes oscillating valence orbitals with smooth, nodeless functions. Because these potentials are fitted to reference data, they are method-dependent and optimized for specific DFAs to ensure physical consistency. Ultimately, PPs significantly reduce computational costs and incorporate relativistic effects by focusing only on the chemically active valence electrons[1].

With a chosen PP, the KS-DFT calculation begins with an initial guess for the electron density[16]. Since the PP is typically generated using a specific DFA, it must be chosen consistently with the functional used in the main calculation[28]. This setup allows for the calculation of $v_{\mathrm{XC},\sigma}(\boldsymbol{r})$ via (2.3.11), which then allows Equation (2.3.8) to be solved[1, 16, 23]. With the resulting KS orbitals, new electron densities are found using

$$n_\sigma(\boldsymbol{r}) = \sum_i^{N_\sigma} |\phi_{i\sigma}(\boldsymbol{r})|^2 \,. \tag{2.3.14}$$

Based on these, new values for $v_{XC,\sigma}(\boldsymbol{r})$ can be calculated. The resulting cycle is continued until densities and orbitals converge. This type of calculation is called a self-consistent field (SCF) calculation. When the change in orbitals and densities becomes negligible, the field is self-consistent. The ground-state energy is then calculated using (2.3.13) with (2.3.12), (2.3.6), (2.3.7), and the chosen DFA.

In this thesis, the KS orbital functions $\phi_{i\sigma}(\boldsymbol{r})$ are represented using a plane wave basis set. When describing an infinite, periodic system, like a crystal structure, it is sensible to use a function that corresponds to these characteristics. Using a linear combination of plane waves is therefore common practice for approximating orbital functions in solids[1, 2, 37]. The underlying equation is defined as

$$\phi_{i\sigma}^{(\boldsymbol{k})}(\boldsymbol{r}) = \sum_{\boldsymbol{G}} c_{\boldsymbol{G},i\sigma}^{(\boldsymbol{k})} e^{i(\boldsymbol{k}+\boldsymbol{G})\cdot\boldsymbol{r}} \,, \tag{2.3.15}$$

where $\phi_{i\sigma}^{(\boldsymbol{k})}(\boldsymbol{r})$ is the i-th KS orbital with spin σ at k-point $\boldsymbol{k}$. The sum runs over all reciprocal lattice vectors $\boldsymbol{G}$ for which

$$E_{\mathrm{cut}} \geq \frac{1}{2}\boldsymbol{G}^2 \tag{2.3.16}$$

is satisfied with the defined cutoff energy E_{cut}. The higher E_{cut} is chosen, the higher the accuracy of the orbital and the greater the computational effort. $c_{\boldsymbol{G},i}^{(\boldsymbol{k})}$ is the complex plane-wave coefficient, which is optimized within KS-DFT to yield the lowest total energy. As stated in Section 2.1, the continuous

integral over the Brillouin zone is approximated by a weighted sum over a discrete grid of evenly distributed k-points[8]. These k-points $\boldsymbol{k}$ are chosen for each reciprocal dimension by setting a calculation parameter called the k-mesh. The influence of neighboring unit cells is also covered by the k-mesh[8]. A calculation with an N×N×N k-mesh in a simple unit cell is mathematically equivalent to a calculation of an N×N×N supercell, in which only the Γ-point ($k = 0$, i.e., the center of the Brillouin zone) is calculated[8].

In comparison to HF theory, both HF and KS-DFT consider an independent-particle model and have a similar approach in terms of computation[1]. KS-DFT, however, has the potential to provide significantly better results, but fails to accurately describe strongly correlated systems[29]. This correlation consists of two distinct contributions, called dynamic and static correlation[2, 38]. Dynamic correlation refers to the instantaneous, short-range avoidance between electrons caused by their mutual Coulombic repulsion. It describes how electrons move to stay apart in real-time, an effect that is approximated by the correlation part of the exchange-correlation functional. Since the ground state is a state of superposition, more than one Slater determinant is necessary to characterize the state. Most systems are dominated by a single Slater determinant, for which KS-DFT is decently accurate. Strongly correlated systems, on the other hand, are defined by a ground state containing relevant contributions from different states. This phenomenon is called static correlation and is most common in transition metals. The additional contributions from a potentially large number of excited state determinants to the dynamic correlation are relatively small, while even a small number of states can have a great impact on static correlation[29, 38]. Additionally, the exact exchange-correlation functional is unknown and standard approximations (like LDA or GGA) often fail to capture long-range and localized electronic effects[29]. As a consequence of these limitations of KS-DFT, a more explicit treatment of the electronic structure problem becomes necessary.

2.4 Many-Body Post-Processing

From this point onwards, KS-DFT will be implicitly referred to as DFT, since basic DFT is not relevant to this work. To improve upon DFT's exchange-correlation functional and to include static correlation, Equation (2.2.4) must first be rewritten in accordance with second quantization[2]. To satisfy the Pauli principle in DFT, a Slater determinant is constructed from the wave functions. In contrast, second quantization maintains the correct exchange statistics through the properties of the operators that are applied to the wave function, rather than through the total wave function itself. This allows for a more efficient treatment of the electronic structure problem.

In this section, spatial orbitals are used implicitly. The derivation and explanation of second quantization originate from references [2], [3] and [18]. The Hamiltonian is first projected onto the M basis wave functions $\phi_t(\boldsymbol{r}_i)$ and many-electron wave functions are considered, which must be antisymmetric upon exchange of two electrons. A convenient shorthand for Slater determinants is established, where N is the total number of electrons and $\boldsymbol{r}_i$ and σ_i are the spatial and spin coordinates of the i-th electron:

$$\psi(\boldsymbol{r}_0, \sigma_0 \dots, \boldsymbol{r}_{N-1}, \sigma_{N-1}) = |f_{M-1}, \dots, f_t, \dots, f_0\rangle = |f\rangle\,. \tag{2.4.1}$$

Here, $f_t = 1$ if ϕ_t is occupied and $f_t = 0$ if ϕ_t is unoccupied. Second quantization deals with the manipulation of these occupation number vectors $|f\rangle$. To describe the state of the system, we use Dirac notation (bra-ket notation). A state is represented by a so-called "ket" $|\dots\rangle$. Mathematically, this ket is a vector in a high-dimensional space. These vectors form an orthonormal basis of this system-specific space called the Fock space. Electrons are excited into higher-energy orbitals by the fermionic creation operators $a_t^\dagger$ and deexcited by the annihilation operators a_t. Their effects on $|f\rangle$ are defined by

$$\begin{aligned} a_t|f_{M-1},\dots,f_t,\dots,f_0\rangle &= \delta_{f_t,1}(-1)^{\sum_{i=0}^{t-1} f_i}|f_{M-1},\dots,f_t \oplus 1,\dots,f_0\rangle\,, \\ a_t^\dagger|f_{M-1},\dots,f_t,\dots,f_0\rangle &= \delta_{f_t,0}(-1)^{\sum_{i=0}^{t-1} f_i}|f_{M-1},\dots,f_t \oplus 1,\dots,f_0\rangle\,, \end{aligned} \tag{2.4.2}$$

where the Kronecker delta ensures that the annihilation only happens if $f_t = 1$ and equally that the creation only happens if $f_t = 0$. The phase term $(-1)^{\sum_{i=0}^{t-1} f_i}$ enforces the exchange antisymmetry of fermions and $\oplus$ denotes the addition modulo 2, which defines $0 \oplus 1 = 1$ and $1 \oplus 1 = 0$. Since the total number of electrons must remain constant, operators in second quantization must contain equal numbers of creation and annihilation operators. Based on this, the Hamiltonian equals

$$H = \sum_{t,u} h_{tu} a_t^\dagger a_u + \frac{1}{2}\sum_{t,u,v,w} h_{tuvw} a_t^\dagger a_u^\dagger a_v a_w\,, \tag{2.4.3}$$

where

$$h_{tu} = \int d\boldsymbol{r}\phi_t^*(\boldsymbol{r})\left(-\frac{\nabla^2}{2} - \sum_I \frac{Z_I}{|\boldsymbol{r}-\boldsymbol{R}_I|}\right)\phi_u(\boldsymbol{r})\,, \tag{2.4.4}$$

$$h_{tuvw} = \int d\boldsymbol{r}_1 d\boldsymbol{r}_2 \frac{\phi_t^*(\boldsymbol{r}_1)\phi_u^*(\boldsymbol{r}_2)\phi_v(\boldsymbol{r}_2)\phi_w(\boldsymbol{r}_1)}{|\boldsymbol{r}_1 - \boldsymbol{r}_2|}\,. \tag{2.4.5}$$

This Hamiltonian includes the same interactions as in Equation (2.2.4), with the exception of the internuclear repulsion, which will be accounted for separately. The indices t, u, v, w correspond to different orbitals. The first term in (2.4.3) represents the kinetic energy of the electrons, as well as their Coulomb interaction with the nuclei. Since only one electron is involved, the term only contains one creation and one annihilation operator. In contrast to that, the second term represents the electron-electron Coulomb repulsion and contains two creation and two annihilation operators, since it involves two electrons. The eigenstates of the Hamiltonian, namely the many-body wavefunctions of the system, can be expressed as

$$|\Psi\rangle = \sum_f \alpha_f |f\rangle\,, \tag{2.4.6}$$

where α_f are the complex coefficients of the Slater determinants $|f\rangle$, which define the contributions of different occupations to the state. The wave function obtained by considering all possible $\binom{M}{N}$ determinants for the ground state, where $\binom{M}{N}$ is the binomial coefficient for M orbitals and N electrons, is referred to as the full configuration interaction (FCI) wave function. Note that this does not mean all values of α_f are non-zero. To reduce the computational effort, the considered states are typically restricted to a small number of excitations above the reference state, most commonly single

excitations and double excitations. This allowed number of excited electrons is referred to as the excitation pool. Since low energy excitations dominate the ground-state wave function in many systems, this approximation can produce highly accurate results.

To improve upon DFT, the single-electron matrix elements h_{tu} and the two-electron matrix elements h_{tuvw} are calculated using the KS orbitals from DFT, which are then used to construct the many-body Hamiltonian[12]

$$H^{(\boldsymbol{k})} = \sum_{t,u} h_{tu}^{(\boldsymbol{k})} a_t^\dagger a_u + \frac{1}{2} \sum_{t,u,v,w} h_{tuvw}^{(\boldsymbol{k})} a_t^\dagger a_u^\dagger a_v a_w + E_{\mathrm{n-n}} + E_{\mathrm{e-self}} \,. \tag{2.4.7}$$

The nuclear interaction $E_{\mathrm{n-n}}$ and the repulsion between each electron and its periodic image $E_{\mathrm{e-self}}$ are added as constant energies. Here, $\boldsymbol{k}$ indicates that the Hamiltonian, as well as $h_{tu}^{(\boldsymbol{k})}$ and $h_{tuvw}^{(\boldsymbol{k})}$ are k-point specific. This is relevant for calculations featuring a k-mesh. For the sake of clarity, the k-point will be omitted in the following, meaning

$$\begin{aligned} \psi_t^{(\boldsymbol{k})} &\to \psi_t \,, \\ \boldsymbol{k} + \boldsymbol{G} &\to \boldsymbol{G} \,, \\ h_{tu}^{(\boldsymbol{k})} &\to h_{tu} \,, \\ h_{tuvw}^{(\boldsymbol{k})} &\to h_{tuvw} \,. \end{aligned} \tag{2.4.8}$$

The single-electron matrix elements h_{tu} are defined as

$$h_{tu} = \int \psi_u^* (\boldsymbol{r}) (T + V_{\mathrm{PP}}) \psi_u(\boldsymbol{r}) d\boldsymbol{r} \,, \tag{2.4.9}$$

where T is the kinetic energy operator and V_{pp} is the pseudopotential operator. For details on the definition and derivation of the operators, see reference [12]. The two-electron matrix elements, which define the periodic electron-electron interaction, are given as

$$h_{tuvw} = \frac{4\pi}{V} \sum_{\boldsymbol{G},\boldsymbol{G}\neq 0} \frac{\tilde{\rho}_{tw}^*(\boldsymbol{G}) \tilde{\rho}_{uv}(\boldsymbol{G})}{\boldsymbol{G}^2} \,, \tag{2.4.10}$$

where the reciprocal space pair density $\tilde{\rho}_{uv}(\boldsymbol{G})$ is defined by the convolution of the Fourier-transformed orbitals $\tilde{\psi}_u^*(\boldsymbol{G})$ and $\tilde{\psi}_v^*(\boldsymbol{G})$ via $\tilde{\rho}_{uv}(\boldsymbol{G}) = \tilde{\psi}_u^*(\boldsymbol{G}) * \tilde{\psi}_v(\boldsymbol{G})$, where $*$ represents the convolution operation[12, 39]. This definition in the reciprocal space is equivalent to the Fourier transform of the real-space pair densities, which are given by $\rho_{uv}(\boldsymbol{r}) = \psi_u^*(\boldsymbol{r}) \psi_v(\boldsymbol{r})$. By starting with the reciprocal space representation, the calculation of the two-electron matrix elements h_{tuvw} becomes a direct summation over $\boldsymbol{G}$-vectors as shown in Equation (2.4.10). The interaction between each electron and its periodic images has been removed from this term and is treated separately in[39]

$$E_{\mathrm{e-self}} = N \sum_{\boldsymbol{L}\neq \boldsymbol{0}} \frac{1}{|\boldsymbol{L}|} \,. \tag{2.4.11}$$

The lattice translation vector $\boldsymbol{L}$ is excluded for $\boldsymbol{L} = 0$, since it is unphysical. Finally, the internuclear repulsion can be calculated via[40]

$$E_{\mathrm{n-n}} = \frac{1}{2}\sum_{I,J}^{N}{\sum_{\boldsymbol{L}}}' \frac{Z_I Z_J}{\left|\boldsymbol{R}_I - \boldsymbol{R}_J - \boldsymbol{L}\right|} \,. \tag{2.4.12}$$

Since $E_{\mathrm{e-self}}$ and $E_{\mathrm{n-n}}$ converge slowly if calculated with equations (2.4.11) and (2.4.12), they are calculated using the Ewald summation technique[41], which improves convergence speed drastically. The Ewald summation splits the equations into short- and long-range terms, which are calculated in real and reciprocal space, respectively. The symmetries of h_{tu} and h_{tuvw} reduce the number of matrix elements that have to be calculated explicitly. h_{tuvw} has additional symmetries, if the wave functions are real[12, 42].

To simplify the calculation, the frozen core approximation[36, 43] is used, which fixes all electrons and orbitals that are not part of a chosen set of relevant electrons and orbitals, called active space. Since orbitals with low-lying energies tend to have an occupation number very close to 1, these orbitals and the corresponding number of electrons can be assumed to be fixed. Following the same logic, orbitals of very high energy have an occupation number close to 0 and can be fixed as well[2, 3]. According to this, the system is split into the active space and the environment. Only electrons in the active space are simulated explicitly and can, therefore, be excited within the active space, while all other electrons in the environment are modeled by modifying the matrix elements of the Hamiltonian. Since the number of electronic configurations only depends on the active space, the computational effort is reduced[12]. This approximation results in an energy offset, the frozen core energy

$$E_{\mathrm{frozen}} = 2\sum_{a} h_{aa} + \sum_{ab}(2h_{abba} - h_{abab}) \tag{2.4.13}$$

and an effective single-electron potential with matrix elements

$$g_{tu} = \sum_{a}(2h_{taau} - h_{taua}) \,. \tag{2.4.14}$$

Orbitals within the active space are defined by the indices t, u, v, w and orbitals in the environment are indexed by a, b. It is assumed that the spin-orbitals are identical for both spins and a and b run over all fully-occupied spatial orbitals in the environment. With this approximation, the Hamiltonian in Equation (2.4.7) is extended to[12]

$$\begin{aligned} H^{(\boldsymbol{k})} = \sum_{t,u}\left(h_{tu}^{(\boldsymbol{k})} + g_{tu}^{(\boldsymbol{k})}\right) a_t^\dagger a_u + \frac{1}{2}\sum_{t,u,v,w} h_{tuvw}^{(\boldsymbol{k})} a_t^\dagger a_u^\dagger a_v a_w \\ + E_{\mathrm{n-n}} + E_{\mathrm{e-self}} + E_{\mathrm{frozen}} \,. \end{aligned} \tag{2.4.15}$$

In contrast to DFT, Equation (2.4.15) includes static correlation and explicitly calculates electron-electron interaction instead of approximating it. Calculations based on Equation (2.4.15) have the potential to improve upon DFT results, in particular for strongly correlated systems[12]. Since materials of high commercial and scientific interest, such as catalysts and high-temperature superconductors, are believed to be strongly correlated, better modeling of these systems holds great value for research and development[2, 44]. The computational framework used for the many-body calculations in this work is called Dopyqo and is openly available as a software package[45].

2.5 Quantum Computing

To calculate the ground-state energy of Equation (2.4.15), a classical computer must be able to store the system's FCI wave function[2, 46]. However, the number of Slater determinants contained in this wave function scales exponentially with $\binom{M}{N}$, where M is the number of orbitals and N is the number of electrons. Consequently, storing the FCI wave function becomes computationally intractable for large systems on classical hardware. In contrast, a quantum computer can store the wave function using only M quantum bits (qubits), enabling both efficient storage and the subsequent calculation of the FCI energy.

As early as 1982, Feynman predicted that the solution for calculating large quantum systems lies in the use of quantum hardware[47]. He remarked, "If you want to make a simulation of nature, you'd better make it quantum mechanical, and by golly it's a wonderful problem, because it doesn't look so easy." The development of quantum hardware and quantum algorithms are believed to tackle classically intractable problems in fields as diverse as chemistry, physics, biology, medicine, materials science, and optimization[2, 37, 48-50]. As quantum hardware develops, we may see a paradigm shift where simulations move from confirming experimental results to providing a higher level of precision than is physically measurable in a laboratory[51]. To date, various efficient quantum algorithms have been developed to address chemical problems[2]. The computational overhead, in terms of both runtime and physical hardware, is expected to scale polynomially (instead of the exponential scaling on classical hardware) with respect to system size and the desired level of precision.

A common misconception is that quantum computers derive their advantages from a generally higher computing capacity[51]. In fact, quantum computers are only superior to classical hardware in very specific applications. This is called quantum advantage and results from the properties of the smallest unit of information in a quantum computer, the qubit, which is realized as a two-level quantum-mechanical system, such as the spin of an electron or the polarization of a photon[51, 52]. Just as with a classical bit, which has a state of either 0 or 1, the qubit has the possible states $|0\rangle$ and $|1\rangle$, defined by

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \qquad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}. \tag{2.5.1}$$

The qubit lives in the two-dimensional space defined by the computational basis states in Equation (2.5.1), called the Hilbert space, where the main difference from the classical bit lies in the qubit's ability to exist in a superposition of both states[2, 51]. A general single-qubit state is described by

$$\begin{gathered} |\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}, \\ \alpha, \beta \in \mathbb{C}, \\ |\alpha|^2 + |\beta|^2 = 1. \end{gathered} \tag{2.5.2}$$

Unlike a classical bit, whose state of 0 or 1 can always be read, we cannot examine a qubit to determine its quantum state, that is, the values of α and β. According to the laws of quantum mechanics, the information acquired is restricted. A measurement of the qubit yields either the result 0, with probability $|\alpha|^2$, or the result 1, with probability $|\beta|^2$. An example of a qubit state with superposition is the $|+\rangle$ state

$$|+\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\,, \tag{2.5.3}$$

defined by a 50/50 chance of measuring either 0 or 1. This state can, for example, be achieved by applying a microwave pulse of appropriate wavelength and duration to an immobilized atom. By expressing Equation (2.5.2) as

$$|\psi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\varphi}\sin\frac{\theta}{2}|1\rangle\,, \tag{2.5.4}$$

the qubit can be represented geometrically. Here, θ and φ define a point on a three-dimensional unit sphere, as shown in Figure 2.5.1.

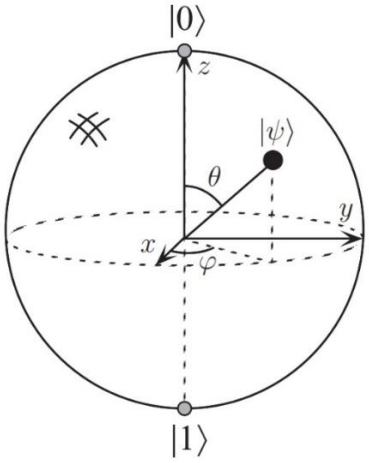


Figure 2.5.1: Representation of a qubit on a Bloch sphere[51].

This representation of a two-level quantum-mechanical system is called the Bloch sphere. It provides a useful means of visualizing the states and many of the operations performed on single qubits[51]. Just as logic gates act on bits of information in classical circuit models, quantum gates are used to manipulate qubits. These quantum gates manipulate both α and β simultaneously. Mathematically, they can be described as unitary matrices. Typical gates include the Pauli gates

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \tag{2.5.5}$$

and the Hadamard and T gates

$$\mathrm{Had} = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad \mathrm{T} = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}. \tag{2.5.6}$$

The assortment of available gates is dependent on the specific quantum computer used, i.e., on the physical implementation of the qubit[2]. Note that unavailable gates can be synthesized from a combination of existing gates. Figure 2.5.2 shows the influence of the Hadamard gate on the state described in Equation (2.5.3).

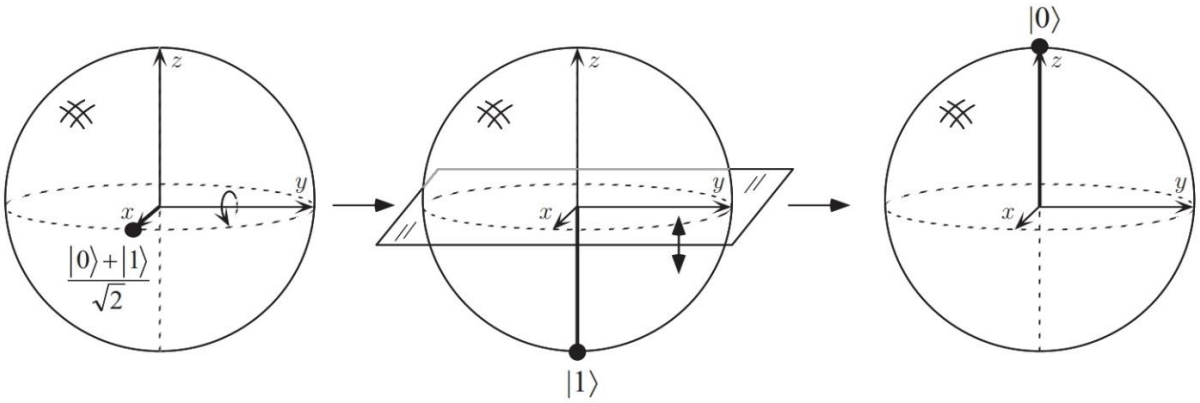


Figure 2.5.2: Visualization of the Hadamard gate on the Bloch sphere, acting on the input $|+\rangle$ state[51].

The Hadamard gate performs a 90° rotation around the y-axis, followed by a 180° rotation around the x-axis. In addition to single-qubit gates, gates acting on more than one qubit exist, which are called multi-qubit gates[2, 51]. The prototypical multi-qubit quantum logic gate is the controlled-NOT gate, also referred to as CNOT gate. For a two-qubit system defined by the tensor product of the qubit's states

$$(\alpha|0\rangle + \beta|1\rangle) \otimes (\gamma|0\rangle + \delta|1\rangle) = \alpha\gamma|00\rangle + \alpha\delta|01\rangle + \beta\gamma|10\rangle + \beta\delta|11\rangle\,, \qquad (2.5.7)$$

the CNOT gate acts upon one qubit as the control qubit and another qubit as the target qubit. If the control qubit is 1, the target qubit is flipped, otherwise, nothing happens. The operation is expressed as

$$|00\rangle \to |00\rangle\,, \quad |01\rangle \to |01\rangle\,, \quad |10\rangle \to |11\rangle\,, \quad |11\rangle \to |10\rangle\,. \qquad (2.5.8)$$

The control qubit is not measured here. This is important, since measuring a qubit leads to its collapse into either 0 or 1, while the information regarding α and β is irretrievably lost. Additionally, the state of a qubit cannot be copied, in accordance with the no-cloning theorem of quantum information. The Hadamard and CNOT gates allow for the preparation of the state

$$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) = \mathrm{CNOT}^{0,1}\mathrm{Had}^{0}|00\rangle\,, \qquad (2.5.9)$$

where the measured value of each qubit is not independent of the other. A state of dependent qubits is called "entangled." By first applying a Hadamard gate to qubit 0 of the state $|00\rangle$, followed by a CNOT gate with qubit 0 as the control and qubit 1 as the target, we obtain an entangled state where measuring one qubit determines the value of the other. While a classical register of n bits can represent 2^n distinct configurations, it can only occupy one of those states at any given moment. In contrast, quantum superposition allows a system of n qubits to exist in a linear combination of all 2^n basis states simultaneously. Quantum parallelism refers to the ability of a quantum algorithm to perform a single physical operation that acts on the entire state vector, thereby influencing all 2^n amplitudes at once[51]. Equation (2.5.7) shows how a system of two qubits exists in a superposition of four states in accordance with the aforementioned 2^n states. By applying quantum gates, these states are transformed and entangled, establishing the logical framework necessary to execute quantum algorithms and derive computational results.

To extract information from a quantum circuit, a measurement of an observable O is performed[2]. Typically, an average value,

$$\bar{O} = \langle\psi|O|\psi\rangle\,, \tag{2.5.10}$$

called the expectation value, is determined based on many measurements. In Dirac notation, $\langle\psi|$ is defined as the complex conjugate transpose of $|\psi\rangle$. A quantum state,

$$|\psi\rangle = \prod_k U_k^{i_k,j_k}(\mathbf{\Theta}_k)|0\rangle^{\otimes n}\,, \tag{2.5.11}$$

is prepared by a quantum circuit defined by the product of all gates acting on n qubits initially in the state $|0\rangle$. Each gate is designated by a unitary operator $U_k^{i_k,j_k}(\mathbf{\Theta}_k)$, where k denotes the k-th gate in the circuit, i and j are the indices of the qubits the gates act upon ($i = j$ for single-qubit gates), and $\mathbf{\Theta}$ are gate parameters. Practically, we perform an iterative preparation of the state $|\psi\rangle$ followed by a measurement, where outcomes for each qubit are labeled +1 (for $|0\rangle$) and −1 (for $|1\rangle$). The expectation value is then derived by averaging these results. To measure qubits in the X or Y basis, i.e., to measure their phase, single-qubit rotations are applied to change the basis of the relevant qubits, which can then be measured in the Z basis. For multi-qubit operators, the outcome can be obtained by taking the product of the measurement of single-qubit operators within the same circuit. Since the Pauli operators (Equation (2.5.5)) and the identity matrix form a complete basis for any Hermitian single- or multi-qubit operator, any multi-qubit observable can be described as a linear combination of Pauli operators, the expectation value of which can be measured efficiently with a quantum computer.

Broadly speaking, quantum parallelism allows quantum algorithms to evaluate a function for many different input values simultaneously[51]. This offers an advantage over classical algorithms for complex problems characterized by exponentially large search spaces. Quantum algorithms providing a significant advantage over classical methods can be categorized into three primary classes. The first class is based on the Quantum Fourier Transform. This group includes the Deutsch-Jozsa algorithm[53] and Shor's algorithm[54] for integer factoring, both of which offer an exponential speedup over their classical counterparts, with profound implications for cryptography and number theory. The second class consists of quantum search algorithms led by Grover's algorithm[55]. By providing a quadratic speedup for searching unstructured databases, these algorithms serve as a versatile tool for a wide array of heuristic and combinatorial problems. The third major class is quantum simulation, where a quantum computer is used to directly and efficiently simulate quantum-mechanical systems. On a classical computer, storing the quantum state requires approximately c^n bits of memory, where c is a constant that depends upon the details of the system being simulated and the desired accuracy of the simulation. In contrast, a quantum computer needs only kn qubits, where k is again a constant that depends upon the details of the system being simulated. One specific quantum simulation, solving the electronic structure problem, is widely recognized as one of the first major applications of quantum computing[2, 46]. It serves as a fundamental prerequisite for a broad range of advanced chemical simulations, including the determination of molecular geometries, the calculation of reaction kinetics, and the characterization of thermodynamic phases and optical properties. Note that for all three classes, quantum algorithms are not only faster but hold the potential to solve problems intractable for even the largest classical supercomputers. In the following, the basic functionality and implementation of the encoding and the algorithm used by Dopyqo are explained.

Simulating chemical systems in second quantization necessitates an encoding scheme that translates fermionic operators into qubit-based counterparts[2]. This mapping bridges the gap between the fermionic Fock space and the qubit Hilbert space, ensuring that the entire range of fermionic states can be accurately captured by the quantum hardware. Dopyqo uses the Jordan-Wigner (JW) encoding, the primary advantage of which lies in its simplicity, as each orbital corresponds directly to a qubit. The description of a system's electronic state in Equation (2.4.1) can be mapped directly onto the qubits. In this manner, the occupation number of a spin orbital is stored in the $|0\rangle$ or $|1\rangle$ state according to

$$\begin{gathered} |f_{M-1}, \dots, f_t, \dots, f_0\rangle \rightarrow |q_{M-1}, \dots, q_t, \dots, q_0\rangle\,, \\ q_t = f_t \in \{0,1\}\,. \end{gathered} \tag{2.5.12}$$

For an example quantum system of four spin orbitals and two electrons, the FCI wave function is given by

$$|\Psi\rangle = \alpha|0011\rangle + \beta|1100\rangle + \gamma|1001\rangle + \delta|0110\rangle\,, \tag{2.5.13}$$

using only four qubits to describe all possible states. With the JW encoding, the second quantized Hamiltonian is mapped onto a linear combination of single-qubit Pauli operators

$$H = \sum_j h_j \prod_i \sigma_i^j\,, \tag{2.5.14}$$

where h_j is a real scalar coefficient, σ_i^j represents one Pauli operator or the identity operator, i denotes which qubit the operator acts upon, and j denotes the term of the Hamiltonian. While other encodings exist, which tend to use fewer gates and are, therefore, more efficient, the JW encoding is preferred due to its simple implementation and ease of interpretability.

The algorithm used by Dopyqo for calculating the ground-state energy based on the many-body Hamiltonian is the variational quantum eigensolver (VQE). The VQE is a hybrid quantum-classical algorithm, meaning it uses a quantum computer for state-preparation and measurement subroutines and a classical computer to process the measurements and update the quantum computer for the next iteration of the calculation according to an update rule[2, 56, 57]. The VQE relies upon the Rayleigh-Ritz variational principle[58]

$$\langle\Psi(\boldsymbol{\Theta})|H|\Psi(\boldsymbol{\Theta})\rangle \geq E_0\,, \tag{2.5.15}$$

stating that the energy expectation value of any trial wave function $|\Psi(\boldsymbol{\Theta})\rangle$ is always greater than or equal to the true ground-state energy E_0, i.e., the lowest energy eigenvalue of the Hamiltonian H. The ground-state wave function and energy can, therefore, be determined by finding the parameter vector $\boldsymbol{\Theta}$ that minimizes the energy expectation value. Since classical computers are unable to efficiently prepare, store, and measure the wave function for large systems, the quantum computer is used for this subroutine. The classical computer is then used to update the parameter vector $\boldsymbol{\Theta}$ according to an optimization algorithm. A scheme of this procedure is shown in Figure 2.5.3.

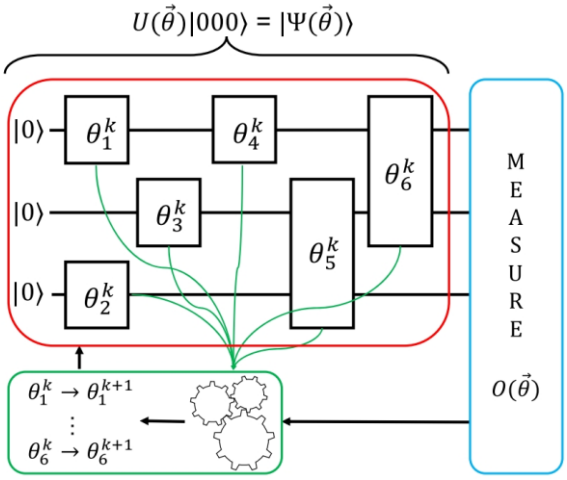


Figure 2.5.3: Scheme of the VQE. The quantum computer's operations are marked in red, the measurement is marked in blue, and the classical computer's optimization is marked in green[2].

Here, all qubits are initialized in the $|0\rangle$ state, after which a series of parameterized gates $U(\mathbf{\Theta}_k) = U_{k,q}(\Theta_q^k) \cdots U_{k,1}(\Theta_1^k)$ is applied to the qubits, with the parameters being determined by the classical computer. The index k denotes the k-th iteration of the cycle. The resulting trial wave function,

$$|\Psi(\mathbf{\Theta}_k)\rangle = U(\mathbf{\Theta}_k)|0\rangle^{\otimes n}, \tag{2.5.16}$$

is prepared and measured many times in order to determine the expectation value. With Equation (2.5.14) and the linearity of expectation values, the energy of the trial wave function is defined as

$$E(\mathbf{\Theta}_k) = \sum_j h_j \left\langle \Psi(\mathbf{\Theta}_k) \middle| \prod_i \sigma_i^j \middle| \Psi(\mathbf{\Theta}_k) \right\rangle. \tag{2.5.17}$$

The resulting energy (calculated by the classical computer from the qubit output), as well as the current parameters $\mathbf{\Theta}_k$, and, optionally, additional values like the gradient, are then used by the optimization algorithm on the classical computer. This algorithm generates new parameters $\mathbf{\Theta}_{k+1}$ used to prepare a new trial wave function $|\Psi(\mathbf{\Theta}_{k+1})\rangle$, which ideally has a lower energy. This procedure continues until the energy converges. The trial state $|\Psi(\mathbf{\Theta})\rangle$ as well as the circuit $U(\mathbf{\Theta})$ generating it are both referred to as the ansatz. An ansatz must be selected appropriately for the available hardware and the chemical system being simulated. A detailed explanation of this matter will follow in Section 4.1.

2.6 Quantum Hardware

Various approaches exist for the physical implementation of a qubit, since it does not have to be an ideal two-state system but only needs to behave as one in a well-controlled environment[59]. Each of these approaches comes with its own set of advantages and challenges. A selection of particularly significant approaches is discussed below.

Trapped-ion quantum computing architectures utilize individual ions confined within electromagnetic traps, where state manipulation is achieved through precisely controlled laser or microwave pulses [60].

These systems are characterized by the highest accuracy in two-qubit gates, which often exceeds the 99.9% threshold. The limitation of this precision, however, is the significantly slower operating speed compared to other approaches.

Photonics-based quantum computers process information by manipulating the optical properties of light[61, 62]. Their main advantages lie in their operation at room temperature and in the extremely high speed of the quantum gates, which can operate on the order of a few nanoseconds. However, since the targeted interaction between photons for computational operations is difficult to implement, universal photonic systems are still limited to a very small number of qubits.

Superconducting quantum computers use superconducting resonant circuits as physical qubits, which must be operated at extremely low temperatures to maintain their quantum properties[63, 64]. This technology is currently widely used because it offers a favorable combination of fast computational operations and high process accuracy for two-qubit gates and is favored by leading companies in the field of quantum computing, such as IBM[65] and Google[66]. A critical drawback, however, is the comparatively short coherence time, i.e., the time during which a qubit can maintain its quantum properties, such as superposition and entanglement, which limits the duration of stable computations. In commercial systems, such as those from IBM, this hardware is already applied with over a hundred qubits. Since connectivity is usually limited to immediate neighbors, algorithms often need to be adapted with additional operations to entangle qubits that are far apart. This constraint, known as gate topology, necessitates the use of SWAP gates to move logical information across the physical hardware[6]. This process significantly increases the total gate count and circuit depth, making the algorithm more susceptible to errors.

A major challenge in the advancements of quantum computation is the development of error-resistant hardware and algorithms[2, 59]. Due to their interaction with the environment and imperfect control, qubits accumulate errors during computation. These errors must be avoided, corrected, or mitigated, otherwise, the results of an algorithm will be distorted and rendered meaningless[67]. Consequently, most quantum computers must be cooled to temperatures close to absolute zero. Avoiding interference from external energy or unrelated particles is an essential step in error prevention. Multiple methods have been developed to mitigate errors during calculations, two of which are explained briefly. By intentionally increasing the error rate by different factors, an extrapolation to the error-free value can be made (Zero Noise Extrapolation)[68, 69]. Symmetry-based methods mitigate VQE errors by verifying that the quantum state respects conserved physical properties, such as particle number and discarding unphysical measurements[70, 71]. These techniques can be implemented through stabilizer circuits, post-processing, or specialized fermion-to-qubit mappings that introduce constraints used to detect and correct errors. The creation of fully error-corrected qubits is crucial for the development of future high-performance quantum computers[72]. Implementing quantum error correction is challenging due to the no-cloning theorem, which prevents the duplication of quantum states, and the fact that measurements can erase encoded information. To overcome this, quantum information is redundantly encoded by entangling it across an expanded Hilbert space of multiple physical qubits to form a single logical qubit. This framework allows for the detection of errors through projective stabilizer measurements, which extract error information without destroying the underlying quantum data. These protocols can theoretically suppress logical errors to arbitrary levels. To realize practical quantum error correction, physical qubit error rates must be suppressed below a specific

technology-dependent threshold while scaling hardware to millions of physical qubits to provide the necessary redundancy for fault-tolerant logical operations. Since current quantum hardware is still far from meeting these requirements, the use of quantum computers remains limited to specialized scientific experiments and feasibility studies.

Dopyqo aims to make near-term quantum computers usable for solving the electronic structure problem for periodic systems. It is currently unclear whether near-term quantum hardware is capable of providing useful results for electronic structure calculations due to the occurrence of errors. Since the error rate increases with higher gate counts, i.e., with longer quantum circuits, developments that enable problems to be tackled with fewer quantum resources are considered important[2]. The VQE is regarded as a promising algorithm for near-term quantum hardware, since it appears to be robust against some errors and is capable of finding the ground state with a shorter circuit than competing algorithms[2, 57]. The VQE, therefore, holds the potential to run calculations without error correction, generating classically intractable results using just error mitigation. Dopyqo aims to provide a framework that enables scientifically meaningful calculations to be performed on near-term quantum computers. To achieve this, already optimized KS orbitals are used instead of performing a plane wave optimization cycle based on the VQE energy, which significantly reduces computation time. Ideally, this approach allows for the accurate calculation of strongly correlated materials using quantum algorithms and making these calculations ready for near-term quantum hardware. To validate that the combination of KS orbitals and the VQE leads to improved results compared to DFT (especially for strongly correlated systems), Dopyqo implements the capability to compute Equation (2.4.15) using classical computers. This is achieved by utilizing the TenCirChem[73] Python library with the PySCF[74] backend to estimate the energy of the ansatz using a state-vector simulation. Using this framework, this work investigates the calculation of bulk moduli with the KS-based many-body Hamiltonian. By comparing bulk moduli for weakly and strongly correlated systems based on DFT and Dopyqo, Dopyqo's calculation method should be validated. Dopyqo implements the L-BFGS optimizer[75] to optimize the parameters in the VQE. Additionally, based on the Python library Qiskit[76], the option is provided to perform the VQE calculation directly on a quantum computer, which enables the computation of larger systems with a larger active space in the near future.

3 Bulk Modulus

Calculations of bulk ground-state properties, such as lattice constants, atomic positions, and bulk moduli, play an important role in the physics of condensed matter[77, 78]. Bulk calculations help in understanding and predicting the mechanical properties of materials. Since the calculation of bulk moduli is a common tool for testing *ab initio* electronic structure techniques such as DFT[79, 80], this section focuses on integrating methods for bulk modulus calculation into Dopyqo. These methods are then used to compare bulk moduli calculated from pure DFT results with those including post-processing calculations. First, Section 3.1 establishes the theoretical foundation of *ab initio* bulk modulus calculations, followed by the computational implementation for bulk modulus calculations in Dopyqo in Section 3.2. Section 3.3 presents a comprehensive analysis of the results obtained for various systems, including convergence tests and an investigation into a systematic error observed in many-body results. Section 3.4 concludes the chapter with an assessment of these findings and a discussion on potential sources of the observed error.

3.1 Theoretical Background of the Bulk Modulus

The bulk modulus represents a material's "three-dimensional stiffness," quantifying its inherent resistance to uniform compression across all classes of solids[81]. Intuitively, it describes the intensity of electron overlap interactions that occur when attempting to force atoms closer together than their equilibrium state. A higher bulk modulus, therefore, signifies a steeper energy penalty for volume reduction, reflecting the strength of the material's interatomic bonds under pressure. The isothermal bulk modulus $B(V)$ is defined as the negative derivative of the pressure P with respect to the unit cell volume V, or the second derivative of the ground-state energy E with respect to the volume[79, 81]

$$B(V) = -V\frac{\partial^2 E}{\partial V^2} = V\frac{\delta P}{\delta V}. \tag{3.1.1}$$

The bulk modulus equals the curvature of the E-V curve and is usually evaluated at the equilibrium volume. To characterize materials, the zero-pressure bulk modulus B_0 and its pressure derivative

$$B_0' = \frac{dB_0}{dP} \tag{3.1.2}$$

are used[79, 81-83]. For simplicity, from here on, B_0 will be implicitly referred to as the bulk modulus and B_0' as the bulk modulus pressure derivative.

The general approach for calculating these values is to fit an equation of state (EoS) to a number of calculated energies for different unit cell volumes[77]. Different EoSs exist, of which three are of relevance to this work.

The Murnaghan EoS

$$E(V) = E_0 + \frac{B_0 V}{B_0'}\left[\frac{(V_0/V)^{B_0'}}{B_0' - 1} + 1\right] - \frac{V_0 B_0}{B_0' - 1} \tag{3.1.3}$$

is based on the assumption that the bulk modulus is a linear function of the pressure[77, 84, 85]. It is mathematically very simple and sufficient if the volume changes are relatively small.

The Birch-Murnaghan (BM) EoS

$$E(V) = E_0 + \frac{9V_0B_0}{8}\left[\left(\frac{V_0}{V}\right)^{2/3} - 1\right]^2 + \frac{9V_0B_0(B_0' - 4)}{16}\left[\left(\frac{V_0}{V}\right)^{2/3} - 1\right]^3 \tag{3.1.4}$$

is the most commonly used EoS for bulk modulus determination[86-88]. It describes the energy as a Taylor series depending on the Eulerian strain. The form shown in (3.1.4) is the most commonly used variant, in which the series is truncated after the third term. It is the "gold standard" for most crystalline solids. It is physically more accurate than the Murnaghan EoS because it directly takes the deformation of the lattice into account.

The Vinet EoS

$$E(V) = E_0 + \frac{2B_0V_0}{(B_0' - 1)^2} \cdot \left\{2 - \left[5 + 3B_0'\left(\left(\frac{V}{V_0}\right)^{\frac{1}{3}} - 1\right) - 3\left(\frac{V}{V_0}\right)^{1/3}\right] e^{-3\left[B_0' - 1\right]\left[\left(\frac{V}{V_0}\right)^{\frac{1}{3}} - 1\right]/2}\right\} \tag{3.1.5}$$

is not based on a Taylor series, but on an empirical interatomic potential[81, 86, 87]. It is ideal for highly compressible solids and often provides better results than the BM EoS at extremely high pressures.

All of these EoSs are commonly used[86] and will therefore be included in the code. Since the BM EoS is the most common choice, it will be used implicitly in this work, unless otherwise stated.

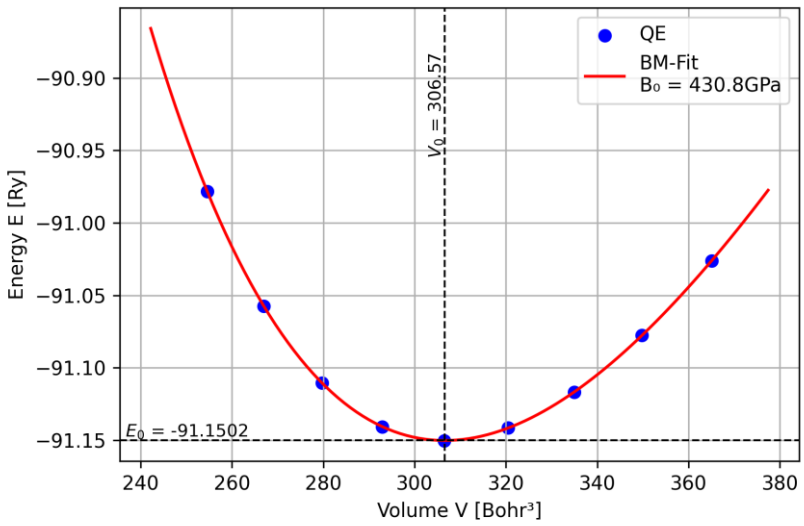


Figure 3.1.1: Data fit to the Birch-Murnaghan (BM) EoS for cubic diamond. The data points were calculated in Quantum ESPRESSO.

Figure 3.1.1 shows a fit of nine data points of cubic diamond, which were calculated using the Quantum ESPRESSO (QE) software package. The plot also shows the equilibrium volume V_0 and equilibrium energy E_0.

The standard procedure for computational investigations in solid-state chemistry starts with a crystal structure obtained from either diffraction studies or through chemical analogy[89]. Lattice shape, volume, and atomic positions are then optimized through the minimization of all forces. This procedure is called relaxation and may require hundreds of self-consistent field iterations across a series of ionic configurations. A corresponding algorithm is part of the QE software package used in this work for DFT calculations[89, 90]. Based on this optimized structure, the volume is scaled to calculate the data points for the EoS fit. This procedure is correct for cubic lattices with high internal symmetry, e.g. rock salt, where the only degree of freedom is the volume[89, 91, 92]. For a structure with more degrees of freedom, each data point requires relaxation. QE allows for the relaxation of different volumes by setting the parameter `cell_dofree = "shape"`, which fixes the cell's volume while allowing all cell parameters to change[93]. If this variable is not set to "shape", all data points would relax to the equilibrium volume. For example, in a triclinic cell, the lengths, angles, and internal positions, in principle, all require optimization. Since these relaxations require a significant amount of computation time, this work focuses on structures with only one degree of freedom.

3.2 Computational Implementation

In this thesis, objects of computational implementation, e.g. method names, variable names, and input file parameters, are written in their original formatting (e.g., `input_path`) to further clarify their origin. To maintain focus on the underlying logic and the relevant parameters, the code snippets in this thesis follow specific conventions. Only significant parts of the code are shown. Full source code, including standard library imports or extensive docstrings, may be omitted to enhance readability. Code examples are occasionally shortened to avoid redundancy and to emphasize the essential scientific content of the implementation.

As a basis for this work, the software package Dopyqo[45] was extended by a dedicated Python module designed to streamline the calculation and analysis of bulk moduli based on both DFT data and post-processed results. Dopyqo is based on the results generated by QE, which is a modern, open-source program suite for modeling materials at the atomic level. It is based on DFT and uses plane waves and pseudopotentials to calculate the electronic structure of solids[90]. QE is Fortran-based and relies mostly on text files for input and output. For that reason, the Dopyqo module for bulk modulus calculation focuses heavily on reading and manipulating text files for managing QE. The basic workflow of the module (shown in Figure 3.2.1) starts with the creation of a QE input file, which is used to call the method `generate_datapoints()`.

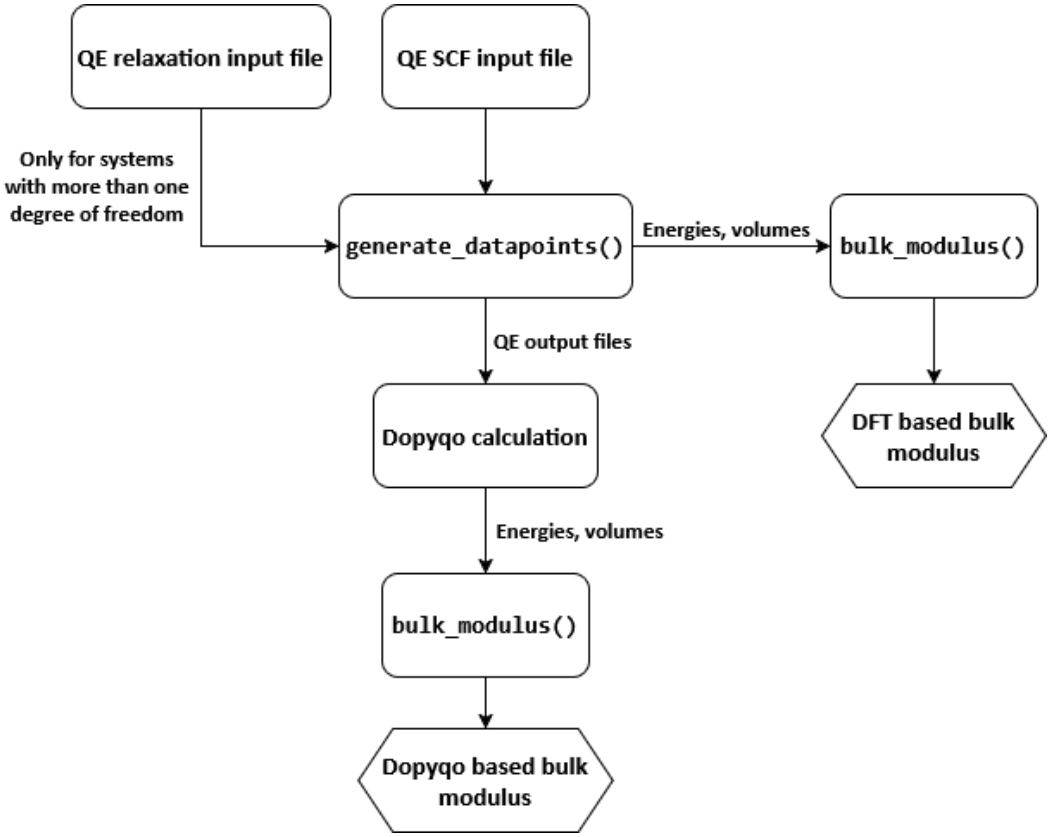


Figure 3.2.1: Workflow of the bulk modulus Dopyqo module.

Based on the input file, the method creates copies with scaled unit cell volumes and starts a QE-SCF calculation for each of these. As an optional argument, `generate_datapoints()` receives the path to a QE relaxation input file. In this case, each volume is previously relaxed and the resulting relaxed structure is transferred to the SCF input file. Finally, the method extracts the energies and volumes from the QE output files and returns them. These can now be used to call the method `bulk_modulus()`, which performs a fit of the data to the chosen EoS. In addition to the returned fit parameters (including the bulk modulus) and their standard deviations, the method has options to save a plot of the $E(V)$-curve and a text file containing all relevant values. For the Dopyqo-based bulk modulus, the QE output files from `generate_datapoints()` are used to perform the post-processing calculations. This calculation is based on methods of the existing Dopyqo software, which is also Python-based. Again, `bulk_modulus()` is used to compute the Dopyqo-based bulk modulus from fitting the energies and volumes. The arguments for calling `generate_datapoints()` are shown in Code Snippet 3.2.1.

```python
def generate_datapoints(
    input_path: str,
    target_dir: str,
    output_dir_name: str,
    relax_path: str = "",
    n_datapoints: int = 4,
    n_cores: int = 4,
    scaling: float = 10,
) -> tuple[list[float], list[float]]:
```

Code Snippet 3.2.1: Arguments of `generate_datapoints()`.

Here, `input_path` defines the path to the underlying SCF input file. With `target_dir`, the user sets a path to a target directory, where the method creates a directory with the name specified in `output_dir_name` to save all results in. As stated in Section 3.1, a relaxation is necessary if the structure has more than one degree of freedom, which is executed if a corresponding path is provided in `relax_path`. The number of data points is controlled by setting `n_datapoints`, where the number defines how many additional larger (and smaller) volumes are calculated. This ensures that the volumes are evenly spread around the equilibrium by enforcing an odd number of total data points. The value in `scaling` works the same way, so it defines the maximum percentage by which the volume is enlarged and reduced. The argument `n_cores` defines the number of CPU cores used for the QE calculation.

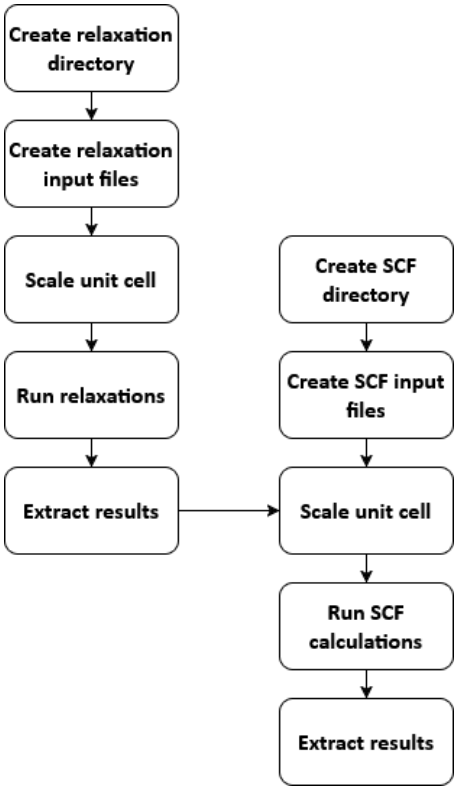


Figure 3.2.2: Concept of `generate_datapoints()`.

The method first validates the passed values and creates the directories for the input and output files. In case of a relaxation, another directory is created as shown in Figure 3.2.2. Both of these contain two subdirectories, “QE inputs” and “QE outputs”. The input file is now copied `n_datapoints` $\times 2 + 1$

times, with each copy differentiated by appending an integer to the file's name. To scale the volume, the given input file needs to be analyzed, since QE possesses multiple parameters for defining the unit cell. The Bravais lattice index `ibrav`, as well as the crystallographic constants `celldm(i)` are read from the file to determine which value needs to be scaled[93]. For a value of `ibrav` $= 0$ and `celldm(1)` not set, the `CELL_PARAMETERS` block needs to be scaled, otherwise, `celldm(1)` is scaled. This implementation of which parameter is used for defining the unit cell is due to the framework provided by QE. For a better understanding of the input files, see Appendix A and [93]. The values are extracted by the method `_read_celldm_ibrav()`, the implementation of which is shown in Code Snippet 3.2.2.

```
    with open(input_path, "r") as f:
        content = f.read()

    # Patterns to search for
    number_rx = r"[+-]?\d+(?:\.\d*)?(?:[eE][+-]?\d+)?"
    pattern = rf"(celldm\s*\(\s*([1-3])\s*\)\s*=\s*)({number_rx})(\s*!.*)?$"

    # Find values
    matches = re.findall(pattern, content, flags=re.IGNORECASE)

    # Fill values into dictionary
    celldm_values = {}
    for i, val in matches:
        celldm_values[f"celldm({i})"] = float(val)

    ibrav = None
    match_ibrav = re.search(r"\bibrav\s*=\s*([-+]?\d+)", content, flags=re.IGNORECASE)
    if match_ibrav:
        ibrav = int(match_ibrav.group(1))

    return celldm_values, ibrav
```

Code Snippet 3.2.2: Implementation of `_read_celldm_ibrav()`.

The code employs regular expressions (regex) to search the file for the `celldm(i)` and `ibrav` values. Regex is a formal syntax used to define search patterns for efficiently matching, extracting, or manipulating specific character sequences within a text. The file is searched for pattern matches, which are written to the variables `celldm_values` and `ibrav` and then returned. As stated, `generate_datapoints()` now differentiates based on which parameter needs to be scaled. The procedure for both parameters is essentially the same, where regex is used to find, read, scale, and replace the values.

```
        # Scale values
        celldm_scaled = np.linspace(
            celldm["celldm(1)"] * (1 - scaling / 100) ** (1 / 3),
            celldm["celldm(1)"] * (1 + scaling / 100) ** (1 / 3),
            2 * n_datapoints + 1,
        )

        # Change celldm in input files
        for i, file in enumerate(input_file_paths, start=0):
            _change_celldm(file, 1, celldm_scaled[i])
            _change_outdir_prefix(file, output_path_out, f"{base_name}_{i+1}")
            _scale_atomic_positions(
                file, scaling_values[i]
            )  # Scales only if QE option is bohr or angstrom

    # Run scf files
    output_files = _run_inputs(input_file_paths, nprocs=n_cores)
```

Code Snippet 3.2.3: Unit cell scaling of `generate_datapoints()`.

As an example, Code Snippet 3.2.3 shows the scaling of the `celldm(1)` value. First, a list of evenly distributed scaling factors is created, based on the arguments `scaling` and `n_datapoints`. This list, as well as the input file paths, are given to the methods `_change_celldm()` and `_scale_atomic_positions()` to scale the corresponding values. `_scale_atomic_positions()` only scales the atomic positions if they are given in Bohr or Angstroms instead of coordinates relative to the cell translation vectors. For this distinction, the selected unit is read using a regex pattern. The method `_change_outdir_prefix()` sets the QE parameters directing the output file paths and names. The QE calculation is now started by the method `_run_inputs()`, which is shown partially in Code Snippet 3.2.4.

```
    # Build command
    cmd = [
        "mpirun",
        "-np",
        str(nprocs),
        qe_cmd,
        "-inp",
        input_file,
    ]
    print(f"Running QE: {' '.join(cmd)} > {output_file}")
    # Start the process and redirect stdout to the file.
    try:
        with open(output_file, "w") as f_out:
            result = subprocess.run(
                cmd,
                check=False,  # Doesn‘t trigger an error if QE fails.
                stdout=f_out,  # Redirects standard output to the file 'f_out'.
                stderr=subprocess.PIPE,  # Captures error messages in 'result.stderr'
            )
            if result.returncode != 0:
                print(f"⚠  Error with {input_file}, Exit-Code {result.returncode}")
    except subprocess.CalledProcessError as e:
        print(f"Error executing {input_file}:")
        print(e.stderr.decode())  # Displays the error message from QE
        raise  # Aborts script
```

Code Snippet 3.2.4: QE calculation start of `_run_inputs()`.

The execution script automates the workflow by dynamically generating output paths and constructing parallel execution commands via `mpirun`, which is a command-line utility used to launch and manage parallel applications by starting multiple instances of a program across one or more processors, enabling them to communicate via the Message Passing Interface (MPI) protocol. `_run_inputs()` utilizes the Python `subprocess` module to manage the external calls to QE, redirecting the standard output to log files for subsequent data extraction. This process loops over all input files. The output file paths are returned and used as an argument to call the method `read_qe_results()`, which returns the total energy and volume from each file by again using a regex search pattern. Finally, the energies and volumes are returned. Note that the method name is not preceded by an underscore "_". In Python, a leading underscore is a conventional signal indicating that a method is intended for internal use only, distinguishing helper functions from the public application programming interface.

This process is slightly modified if a structure relaxation is included. As shown in Figure 3.2.2, the relaxation process works equivalently to the SCF calculation. Here, the unit cells are scaled inside the relaxation input files. Accordingly, the resulting structures from the relaxation output files are transferred into the SCF input files without further scaling their parameters. The energies and volumes can be used directly to call the method `bulk_modulus()`, as visible in Code Snippet 3.2.5.

```
    def bulk_modulus(
        E: NDArray[np.float64] | list[float],
        V: NDArray[np.float64] | list[float],
        eos: Literal[1, 2, 3],
        init_params: list[float] = None,  # type: ignore
        print_results: bool = False,
        plot_results: bool = False,
        save_results: bool = False,
        output_path: str = os.getcwd(),
        output_name: str = "bulk",
    ) -> tuple[NDArray[np.float64], NDArray[np.float64]]:
```

Code Snippet 3.2.5: Arguments of `bulk_modulus()`.

The EoS used for the fitting function is selected via the marker `eos`. Initial fitting parameters can be set via `init_params` and will otherwise be set to realistic default values. The method offers the option to print the results in the console, save them in a text file, and save a PNG file of the $E(V)$-plot. The storage location and name prefix of these files can be set via `output_path` and `output_name`. After validating all arguments, the method `curve_fit()` from the Python library SciPy is used to fit the given data to the EoS of choice. The marker `eos` addresses one of three methods defining the Murnaghan EoS (1), the Birch-Murnaghan EoS (2) and the Vinet EoS (3). Code Snippet 3.2.6 shows the calculation of the BM EoS equivalent to Equation (3.1.4).

```
    def _birch_murnaghan_eos(
        V: float | NDArray[np.float64] | list[float],
        E0: float,
        V0: float,
        B0: float,
        B0_prime: float,
    ) -> float | NDArray[np.float64]:

        # Convert type
        V = np.array(V)

        # EoS-Equation:
        f = (V0 / V) ** (2 / 3) - 1
        coeff_quad = 9.0 * V0 * B0 / 8.0
        coeff_cubic = 9.0 * V0 * B0 * (B0_prime - 4.0) / 16.0

        E = E0 + coeff_quad * f**2 + coeff_cubic * f**3

        return E
```

Code Snippet 3.2.6: Method for the BM EoS calculation.

Note that the docstring has been omitted in Code Snippet 3.2.6. After fitting, the results are printed, plotted by `plot_bulk()`, and saved by `save_bulk()` if specified in the arguments. Finally, the fitting parameters and their standard deviations are returned.

The creation of the text file by `save_results()` is rather simple and is not discussed any further. An exemplary result file can be found in Appendix B. The method `plot_bulk()` generates plot points based on the fitted parameters and the corresponding EoS method. It relies on the matplotlib Python library to create the plot itself. Since the computational environments differ from system to system, the method checks if a graphical user interface (GUI) for visualization is available and saves the plot as a PNG file otherwise. Code Snippet 3.2.7 shows the test for the availability of a GUI-backend, setting the flag `gui_available` accordingly.

```
    # Try using GUI-Backend to plot, otherwise save plot file
    try:
        matplotlib.use("TkAgg")  # Tkinter-Backend
        gui_available = True
    except ImportError:
        try:
            matplotlib.use("Qt5Agg")  # Qt5-Backend
            gui_available = True
        except ImportError:
            matplotlib.use("Agg")  # headless-Backend
            gui_available = False
```

Code Snippet 3.2.7: GUI search of `plot_bulk()`.

The code attempts to initialize an interactive GUI backend for plotting, prioritizing TkAgg and falling back to Qt5Agg if necessary. If neither is available, it defaults to the headless Agg backend to allow saving plots to files without displaying them on screen. These specific backends were chosen to maximize compatibility by prioritizing the standard Python GUI (TkAgg) and a common high-performance alternative (Qt5Agg) for interactive use, while guaranteeing a fail-safe (Agg) for headless environments like servers[94]. `plot_bulk()` shows or saves the plot according to `gui_available`.

Note that `save_results()` and `plot_bulk()` are not preceded by an underscore "_". These methods are intended to support users with deviating workflows. Additionally, the method `read_qe_results()`, which is also used in `generate_datapoints()` to extract energies and volumes, is meant to allow users to easily import data from their own QE calculations. Lastly, `read_results_dat()` allows the user to import the data from previously saved files by `save_results()`. This method also uses regex for this purpose. To increase user-friendliness, a range of different methods for converting volumes, energies, and bulk moduli between common units are offered.

An exemplary start of a workflow according to Figure 3.2.1 begins with the QE calculation by `generate_datapoints()` as shown in Code Snippet 3.2.8.

```
E, V = generate_datapoints(
    "/home/hage_nc/Masterarbeit/BulkModulus/NaCl/NaCl_scf.in",
    "/home/hage_nc/Masterarbeit/BulkModulus",
    "NaCl",
)

bulk_modulus(
    E,
    V,
    2,
    plot_results=True,
    save_results=True,
    output_name="NaCl",
    output_path="/home/hage_nc/Masterarbeit/BulkModulus/NaCl",
)
```

Code Snippet 3.2.8: Calculation of the DFT based bulk modulus for sodium chloride (NaCl).

Since `generate_datapoints()` offers reliable default values for most of its arguments, only the input and output paths, as well as the output directory name need to be set. The energies and volumes returned by the method are used directly to calculate the bulk modulus. Results and a plot are saved just by setting four additional arguments of `bulk_modulus()`.

Following Code Snippet 3.2.8, Dopyqo calculations can be started based on the QE outputs of `generate_datapoints()`. Code Snippet 3.2.9 shows a customized Dopyqo workflow, followed by the bulk modulus calculation based on it.

```
fci_energy = []
for x in range(1, 10):

    vqe_optimizer = dopyqo.VQEOptimizers.L_BFGS_B
    config = dopyqo.DopyqoConfig(
        base_folder="/home/hage_nc/Masterarbeit/BulkModulus/NaCl/QE_outputs",
        prefix=f"NaCl_scf_{x}",
        active_electrons=8,
        active_orbitals=8,
        run_fci=True,  # Run FCI calculation
        run_vqe=False,  # Run VQE calculation
        vqe_optimizer=vqe_optimizer,
        vqe_excitations=dopyqo.ExcitationPools.SINGLES_DOUBLES,
        n_threads=10,
        kpoint_idx="all",
        use_qiskit=False,
    )
    energy_dict, wfc_obj, h_ks, mats = dopyqo.run(config)
    fci_energy.append(Hartree_Rydberg(energy_dict["fci_energy"])
)

bulk_modulus(
    fci_energy,
    V,
    2,
    plot_results=True,
    save_results=True,
    output_path="/home/hage_nc/Masterarbeit/BulkModulus/NaCl",
    output_name="NaCl_dopyqo",
)
```

Code Snippet 3.2.9: Dopyqo-based bulk modulus calculation for NaCl.

Since all QE output files are in a single directory, only distinguished by an integer at the end of each file's name, the Dopyqo calculation can be looped over these integers. All calculation parameters are set by creating a `dopyqo.DopyqoConfig` object. The path to the QE_outputs directory and the name prefix of the files allow Dopyqo to gather all relevant data from the QE calculation. Additionally, calculation parameters like the active space and excitation pool are set. The calculation is started via the method `dopyqo.run()` and the individual total ground-state energies are saved in `fci_energy`. To match the output of QE, the energies from Dopyqo are converted from Hartree (Ha) to Rydberg (Ry). The energies `fci_energy` and the volumes `V` from Code Snippet 3.2.8 are then used to calculate the Dopyqo-based bulk modulus.

3.3 Analysis of the Bulk Modulus Calculations

All calculations were performed on a high-performance Linux workstation server. The system is equipped with two Intel Xeon Gold 6240 processors, which correspond to a total of 36 physical cores (72 logical threads). The working memory comprises 512 GB of RAM. Two NVIDIA Quadro RTX 8000 GPUs, each with 48 GB of VRAM (CUDA version 12.8), were used for GPU-accelerated computing operations. A 1.9 TB NVMe SSD served as the primary storage for data processing and an 11 TB storage volume was used for long-term archiving. All Dopyqo-based results calculated in this section are based

on FCI calculations unless stated otherwise. Norm-conserving PPs from the SG15 optimized norm-conserving Vanderbilt PPs are used, which are generated with the optimized norm-conserving Vanderbilt pseudopotential code in a scalar-relativistic version[95]. All calculations performed use the PBE functional.

Before calculating the data points for the bulk modulus, the material's structure needs to be relaxed as described in Section 3.1. The relaxation is preceded by a convergence test of the value `ecutwfc`, which is defined as the kinetic energy cutoff for wave functions, as defined in Equation (2.3.16), in Rydberg[93]. Since this value scales with the computation time, while also determining the accuracy of the calculation, it is of great importance to find a value where the result is converged and the computation time is minimal. For this purpose, multiple SCF calculations for the non-relaxed structure are performed covering a wide range of `ecutwfc` values. Convergence is considered to be achieved below an energy difference of 10^{-3} Ry per atom to the previous value. Accordingly, for the example of NaCl, as shown in Figure 3.3.1, `ecutwfc` = 50 Ry is selected.

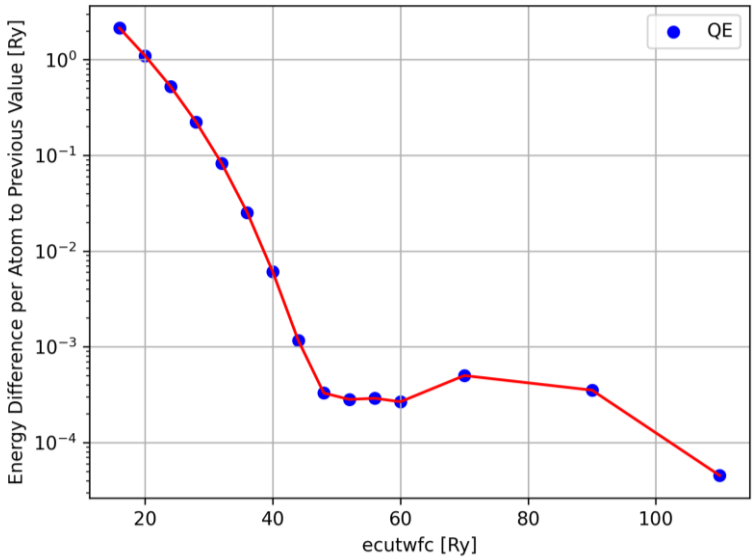


Figure 3.3.1: ecutwfc convergence test for NaCl.

Using this value, the equilibrium structure is calculated via the QE mode `vc-relax` (variable-cell relaxation). The resulting optimized cell parameters and atomic positions are transferred into an SCF input file. Following this, another `ecutwfc` convergence test is conducted. In most cases, this leads to the same result, but it assures that the value is also correct for the new geometry. Lastly, the convergence of the k-mesh needs to be analyzed, since its size also scales with accuracy and computation time. The k-mesh is set via the input file parameter K_POINTS, consisting of six values[93]. The first three describe the number of k-points along each reciprocal lattice direction forming a grid inside the first Brillouin zone. The latter three digits define the shift of the k-points. The shift refers to a displacement of the k-point grid, often by half a grid spacing, so that the sampling points are no longer centered at the origin (Γ-point)[8]. This procedure is often used to achieve better numerical convergence with fewer points by avoiding high-symmetry positions and providing a more representative sampling of the Brillouin zone volume. Figure 3.3.2 displays the k-mesh convergence test for NaCl, containing a curve for each shifted and unshifted k-point grid.

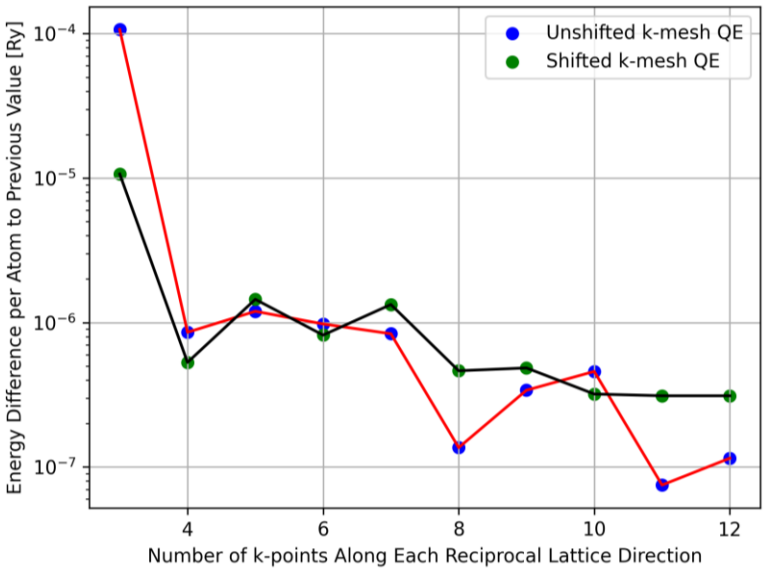


Figure 3.3.2: k-mesh convergence test for NaCl.

In this specific case, convergence is already achieved for relatively small k-meshes. Here, a symmetric 444111 k-mesh is used. A shifted 444000 k-mesh would also be a valid choice, but has no visible advantages. Since NaCl has a cubic structure, a uniform k-point mesh (e.g., 4×4×4) is used to sample all reciprocal directions equally[8]. For other crystal systems with unequal lattice parameters, the mesh density would need to be adjusted accordingly to maintain a consistent sampling of the Brillouin zone. The described procedure of relaxation, `ecutwfc` convergence tests, and k-mesh convergence test determines all values necessary to construct an SCF input file. This file can be used by the method `generate_datapoints()`. The procedure was performed for all discussed structures in this work before proceeding with further calculations.

The chosen volume range, set in `generate_datapoints()`, has a non-negligible influence on the resulting bulk modulus and its pressure derivative. Its effect is visualized exemplarily in Figure 3.3.3 for NaCl based on both QE and Dopyqo calculations.

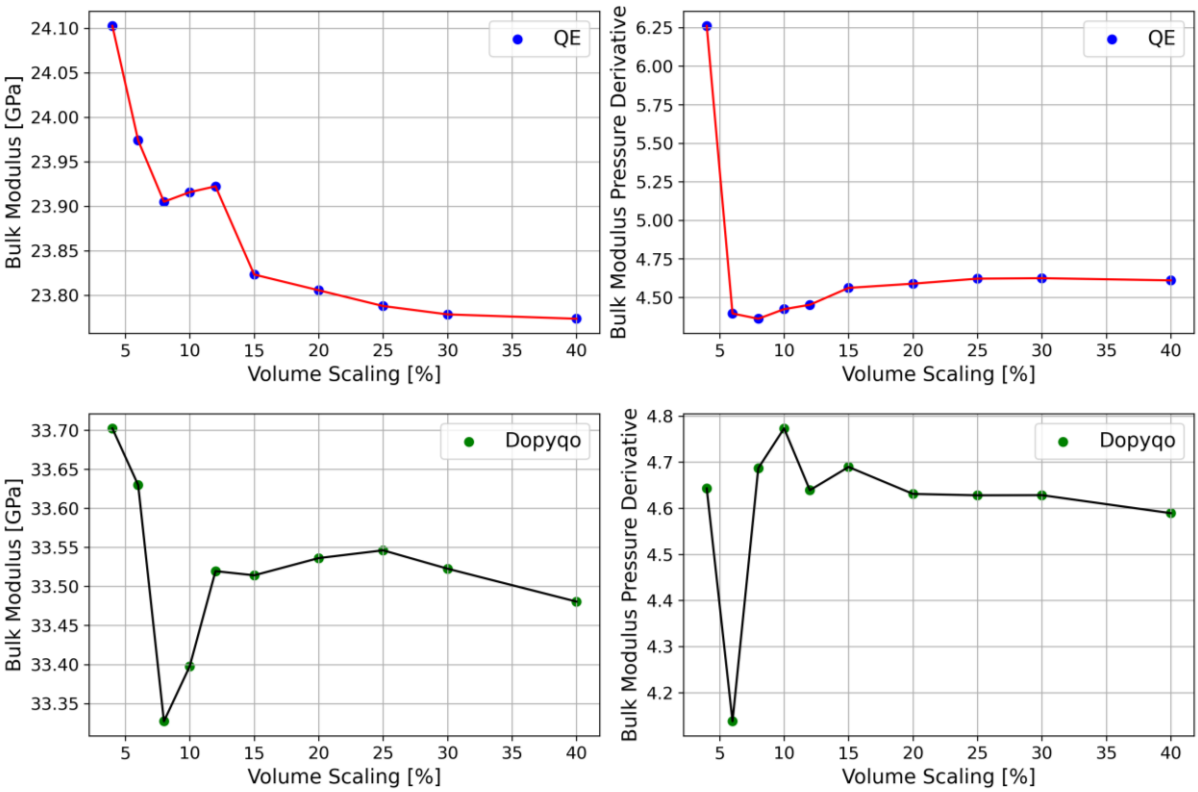


Figure 3.3.3: Influence of the volume scaling interval on the bulk modulus and its pressure derivative for NaCl with k-mesh 444111. Top-left: DFT-based bulk modulus. Top-right: DFT-based bulk modulus pressure derivative. Bottom-left: Dopyqo-based bulk modulus. Bottom-right: Dopyqo-based bulk modulus pressure derivative.

Both, the bulk modulus and its pressure derivative, converge with increased volume scaling for both DFT- and Dopyqo-based calculations. Furthermore, the values towards which the calculations converge remain identical, regardless of whether the data points are centered around the minimum of the DFT or Dopyqo calculation. This insight also allows covering the curves of the DFT-based fits and the volume-shifted Dopyqo-based fits with one data set by selecting larger volume intervals (~20 %) without distorting the results.

The k-mesh is an essential parameter for the bulk modulus calculation. While quantities like Bader charges can be calculated by only including the Γ-point[12], the bulk modulus is dependent on the use of a converged k-mesh, as visible for NaCl and Calcium (Ca) in Figure 3.3.4.

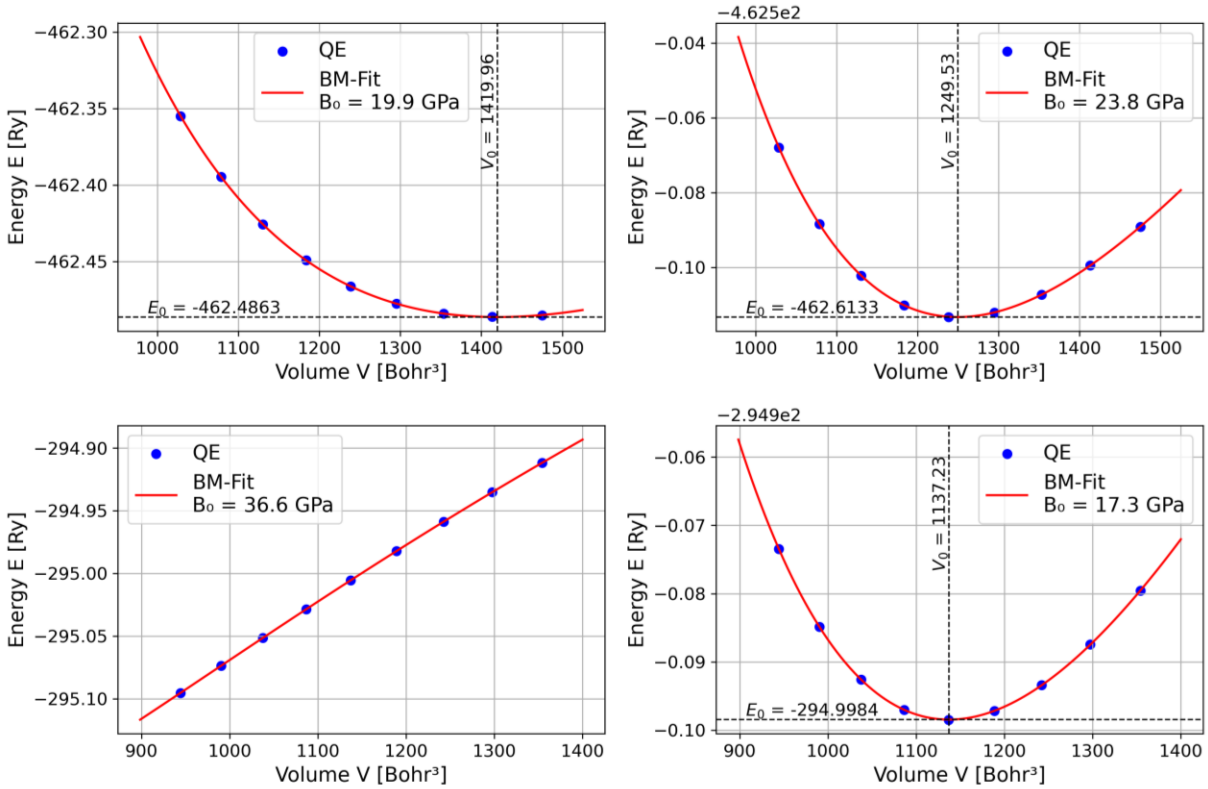


Figure 3.3.4: BM fit comparison at the Γ-point and with a k-mesh. Top-left: NaCl at the Γ-point. Top-right: NaCl with 444111 k-mesh. Bottom-left: Ca at the Γ-point. Bottom-right: Ca with an 888111 k-mesh.

The k-mesh-based calculations on the right side correspond much more closely to the expected parabolic shape. Additionally, the resulting bulk moduli are closer to the experimental values of 23.9 GPa[96] for NaCl and 17.4 GPa[97] for Ca. Using only the Γ-point fails because it represents an isolated sample of the Brillouin zone that cannot capture the necessary electronic properties across the entire crystal. To obtain an accurate bulk modulus, a dense k-mesh is needed to properly integrate the total energy and its second derivative with respect to the volume change. From this point onwards, converged k-meshes, as shown in Figure 3.3.2, are used implicitly for all calculations unless stated otherwise.

The influence of the chosen active space for the Dopyqo calculation is shown in Figure 3.3.5 for diamond.

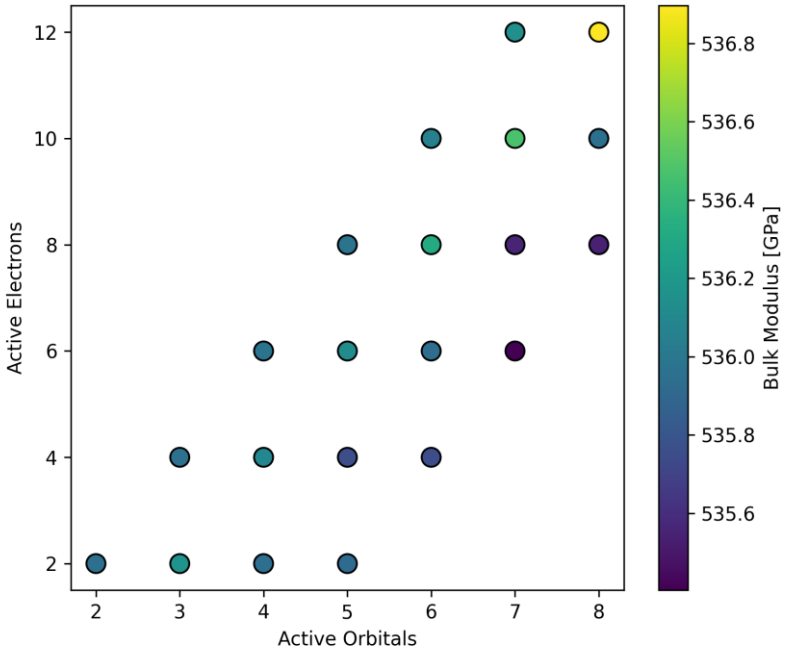


Figure 3.3.5: Influence of the active space on the bulk modulus for diamond.

Here, all possible active spaces within a reasonable computation time have been included. As visible, the active space has only a small influence on the resulting bulk modulus. This is true for all discussed structures in this section, for all of which an equivalent active space variation was calculated. As a compromise between accuracy and computation time, an active space of six electrons and six orbitals (6e6o) is used in the following unless stated otherwise. While larger active spaces like 12e8o change the resulting bulk modulus slightly, these calculations take several hours and up to several days depending on the structure calculated.

The absolute value of the ground-state energy cannot be used as a comparison measure for DFT and Dopyqo, since this value is not experimentally determinable and depends heavily on the underlying theory physical basis. The bulk moduli and equilibrium volumes, on the other hand, are measurable quantities, which can be compared directly. Figure 3.3.6 shows the BM fit of a DFT- and Dopyqo-based bulk modulus calculation for aluminum (Al).

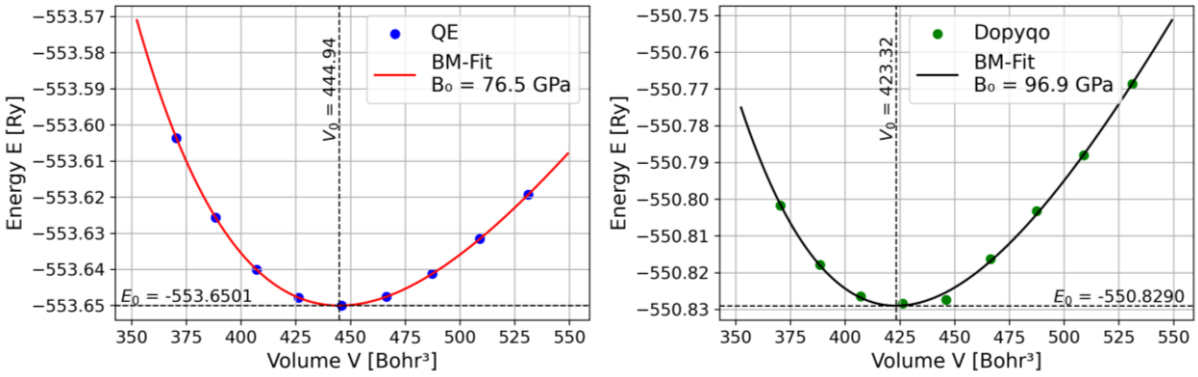


Figure 3.3.6: Comparison of DFT- and Dopyqo-based BM fit for Al.

Figure 3.3.6 shows a strong shift of Dopyqo's equilibrium volume V_0 towards smaller values. Additionally, the bulk modulus is significantly greater compared to DFT. Since the experimental bulk modulus is 74.2 GPa[98] and the experimental unit cell volume is 447.6 Bohr^3[99], it is clear that Dopyqo worsens the results. Figure 3.3.7 visualizes both plots with normalized energies to allow for a better comparison.

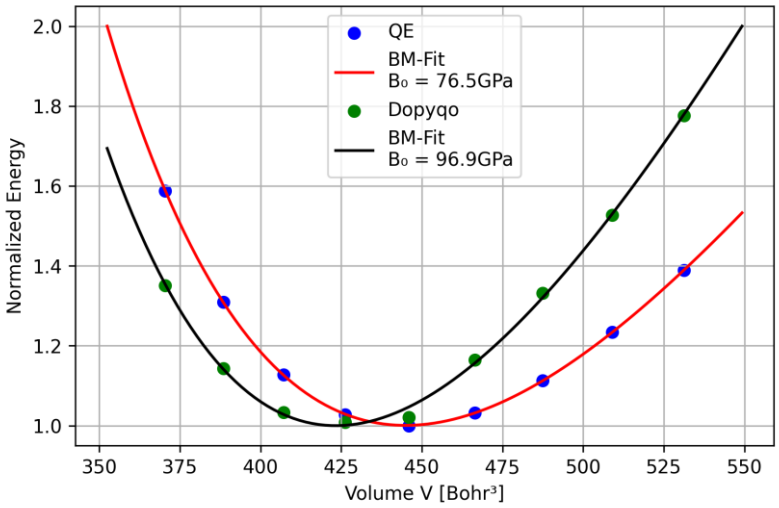


Figure 3.3.7: Normalized DFT- and Dopyqo-based BM fits for Al.

This representation also shows that the energy-volume curve produced by the Dopyqo calculation exhibits a steeper curvature, meaning the second derivative is larger than the DFT-based reference. Consequently, the model predicts a larger energy penalty for volume changes, leading to an overestimation of the bulk modulus in accordance with Equation (3.1.1). This equation also indicates that the smaller equilibrium volume reduces the bulk modulus. This is mathematically correct but ignores the physical correlation between the equilibrium volume and the second derivative of energy with respect to volume. The overestimation of the bulk modulus in connection with an underestimation of the equilibrium volume is a phenomenon common for LDA-based DTF calculations, called overbinding[29]. Estimating stronger bonding for a system results in the aforementioned greater energy penalty for volume changes while also rewarding a smaller equilibrium interatomic distance, i.e., a smaller equilibrium volume. This is a very unexpected behavior since the underlying DFT calculation is PBE-based, which typically exhibits a tendency toward underbinding instead of

overbinding. Additionally, the deviation from experiment is large compared to typical deviations of DFT calculations[35]. This overbinding has been a consistent phenomenon for all examined structures. Table 3.3.1 shows the calculated values for these structures, as well as the corresponding experimental values and the resulting deviations in percent.

Table 3.3.1: Calculated equilibrium volumes and bulk moduli, including their deviations from the experimental values.

	Equilibrium Volume				
	DFT		Dopyqo		Experiment
	[Bohr^3]	Deviation [%]	[Bohr^3]	Deviation [%]	[Bohr^3]
Ca	1137.2	- 2.0	1103.1	- 4.9	1159.9[100]
Al	444.9	- 0.6	423.3	- 5.4	447.6[99]
Si	1109.0	2.6	987.1	- 8.7	1081.4[101]
Diamond	306.6	0.1	288.5	- 5.8	306.3[102]
NaCl	1249.5	5.4	1135.5	-4.2	1185.1[103]
	Bulk Modulus				
	DFT		Dopyqo		Experiment
	[GPa]	Deviation [%]	[GPa]	Deviation [%]	[GPa]
Ca	17.3	- 0.6	19.7	13.2	17.4[97]
Al	76.5	3.1	96.9	30.6	74.2[98]
Si	87.2	- 11.7	123.4	24.9	98.8[104]
Diamond	430.8	- 3.2	536.5	20.6	445.0[105]
NaCl	23.8	- 0.4	33.5	40.2	23.9[96]

For all structures, Dopyqo-based bulk moduli are larger and the equilibrium volumes are lower than the corresponding values based on DFT. This phenomenon occurs independently of the chosen EoS for the fit. The deviations from experimental values are much higher for Dopyqo-based calculations, with the sole exception of the NaCl equilibrium volume. For lead (Pb) and sodium (Na), these deviations are so great that comparing the fit parameters no longer makes sense. The normalized-energy plots for these structures are shown in Figure 3.3.8.

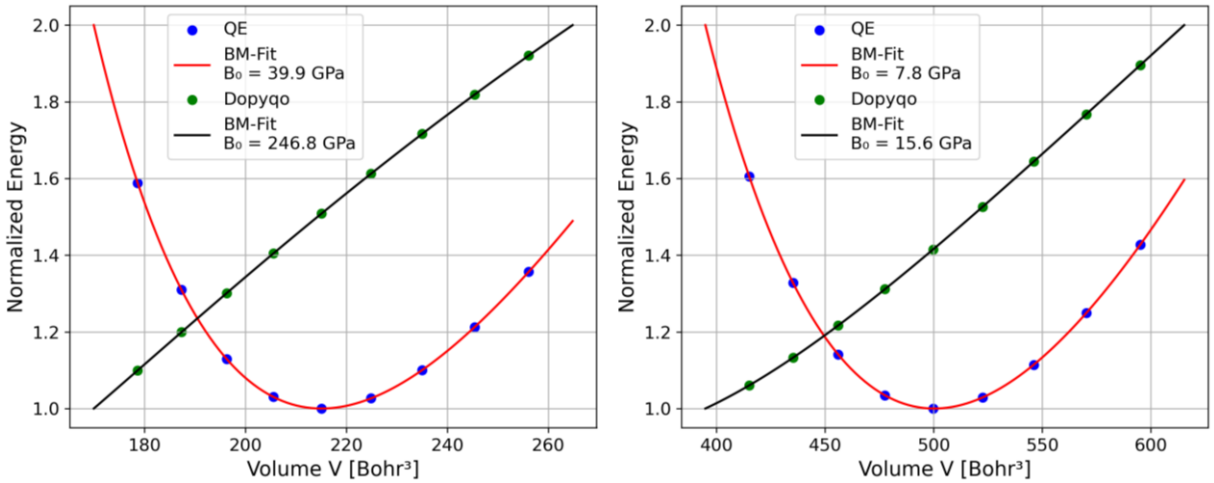


Figure 3.3.8: Normalized energy DFT- and Dopyqo-based BM fits for Pb (left) and Na (right).

These structures were chosen to verify this systematic error over different structure and material types. While all chosen structures are cubic for simplified calculations, face-centered cubic (fcc), hexagonal close-packing (hcp), body-centered cubic (bcc), and diamond structures are included to preclude structure dependency. Also, metals, insulators, and a semiconductor are included since these require different treatments of the electronic occupation near the Fermi level[90, 106]. For insulators and semiconductors, the presence of a band gap allows for the use of fixed occupations, as bands are either entirely full or empty. In contrast, metals require smearing techniques to broaden the sharp discontinuity at the Fermi level, which ensures numerical stability during the SCF cycle. In smearing, the sharp step function is replaced by a smooth curve (e.g., a Gaussian curve or the Fermi-Dirac distribution). This simulates the electrons having an artificially elevated temperature. As a result, states near the Fermi edge are no longer strictly 0 or 1, but take on values in between instead.

In order to find the source of this systematic error, calculation parameters were examined first. The used k-meshes and `ecutwfc` values are converged in terms of the QE calculation, but their influence on Dopyqo's results has not been studied yet. Figure 3.3.9 shows the comparison of DFT's and Dopyqo's energy convergence at different `ecutwfc` values for the representative structures silicon (Si) and Pb.

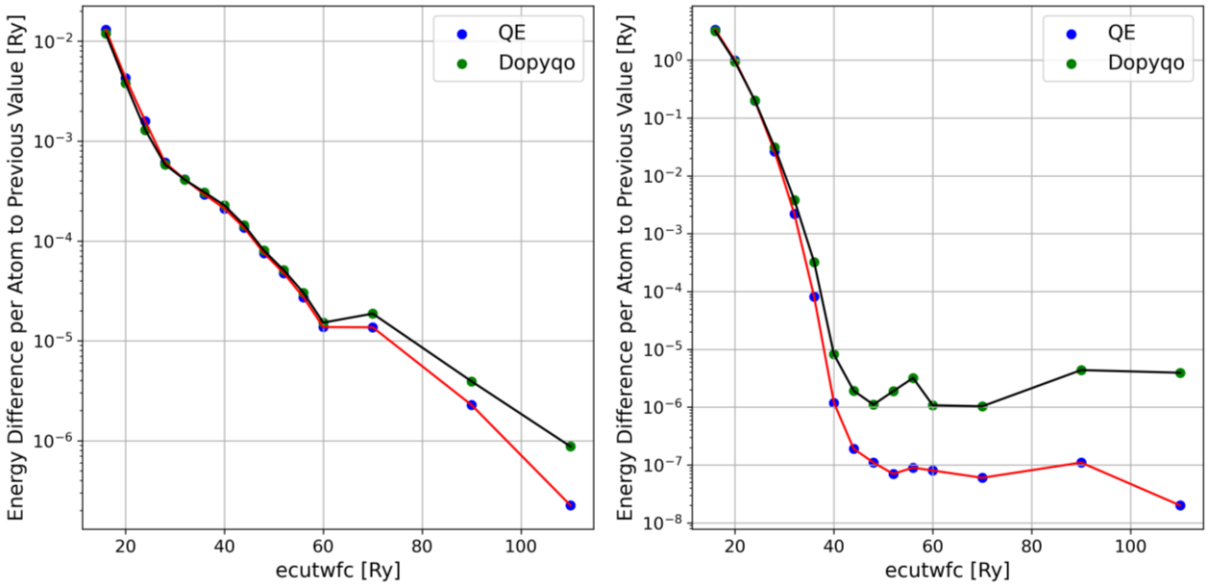


Figure 3.3.9: Comparison of the energy convergence of DFT and Dopyqo calculations for different `ecutwfc` values in Si (left) and Pb (right).

The basic trend of the DFT- and Dopyqo-based curves is identical. Accordingly, no significant difference in the influence of `ecutwfc` between DFT and Dopyqo is to be expected. Figure 3.3.10 visualizes the influence of the k-mesh on Dopyqo's results for the representative structure of diamond.

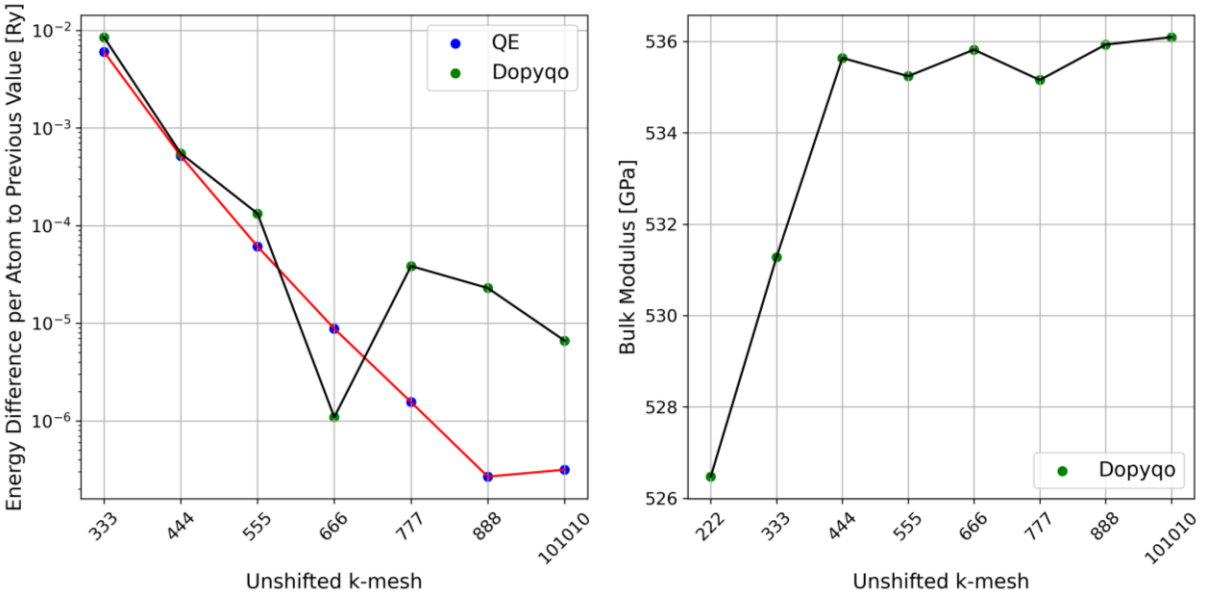


Figure 3.3.10: Left: Comparison of the convergence based on the unshifted k-mesh for DFT and Dopyqo for diamond. Right: Dopyqo-based bulk modulus depending on the unshifted k-mesh for diamond.

It is visible that the energy calculated by Dopyqo converges similarly to the underlying DFT-based energy. Additionally, the bulk modulus converges starting from a k-mesh of 444000 onwards. This k-mesh variation, as well as the influence of `ecutwfc`, has been tested for all previously mentioned structures. It is clear that Dopyqo's results are fully converged and the error originates from a more fundamental problem.

To evaluate whether the concept of using the k-points and corresponding weights from QE for Dopyqo might cause the problem, different supercells were calculated at the Γ-point. These calculations did not yield unambiguously interpretable results. For further information, see Appendix C.

Another potential source of error lies in how Dopyqo handles overlapping bands. As illustrated in Figure 3.3.11 for NaCl and Al, the energetic ordering of bands can shift between different k-points.

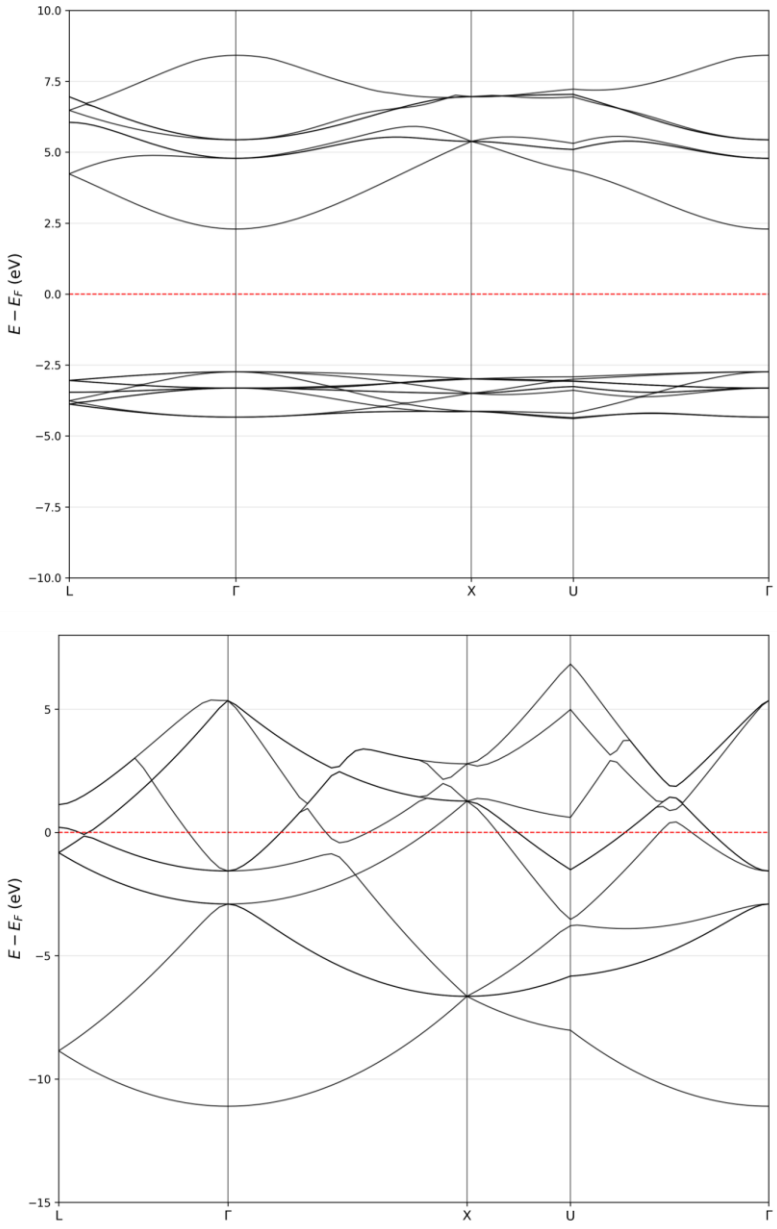


Figure 3.3.11: Band structures for NaCl (top) and Al (bottom) calculated with QE. The energies are centered around the Fermi energy E_F.

Consequently, a specific orbital might reside within the active space at one k-point but fall out of it at another. However, the convergence of the active space, as shown in Figure 3.3.5, suggests that this is not the source of the error. Additionally, the difference in energetic order between k-points can introduce inconsistencies where an orbital's occupancy state varies across the Brillouin zone within a single calculation. It is unclear whether this has any consequences for the resulting energy. To examine

this influence, bulk modulus calculations were performed for a 2×2×2 supercell of magnesium hydride (MgH_2) with a central iron substituent[12]. This structure was selected because of its band structure (Figure 3.3.12), where the frontier bands, i.e., those closest to the Fermi energy, are well-isolated.

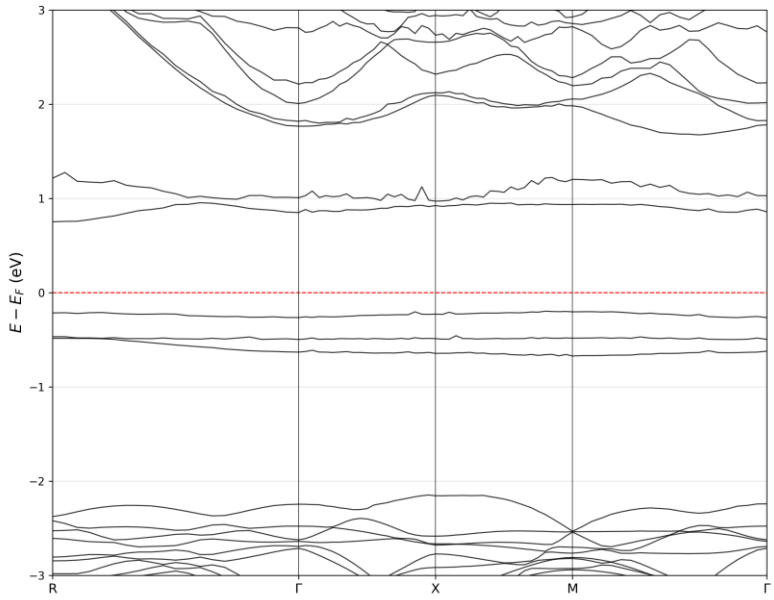


Figure 3.3.12: Band structure of a 2×2×2 MgH_2 supercell with a central substituted iron atom calculated with QE.

For this system, the DFT-based equilibrium volume was 3186.8 $Bohr^3$ compared to 3241.0 $Bohr^3$ for Dopyqo, with corresponding bulk moduli of 51.23 GPa and 55.66 GPa, respectively. Experimental comparative data are not available for this structure. In accordance with the isolated bands shown in Figure 3.3.12, the active space for this calculation was set to 6e5o. These results show significantly better agreement between DFT and Dopyqo than previous cases. Furthermore, the systematic overbinding observed in other structures is absent here. However, the active space variation for this structure shown in Figure 3.3.13 implies that the isolation of the bands is not the reason.

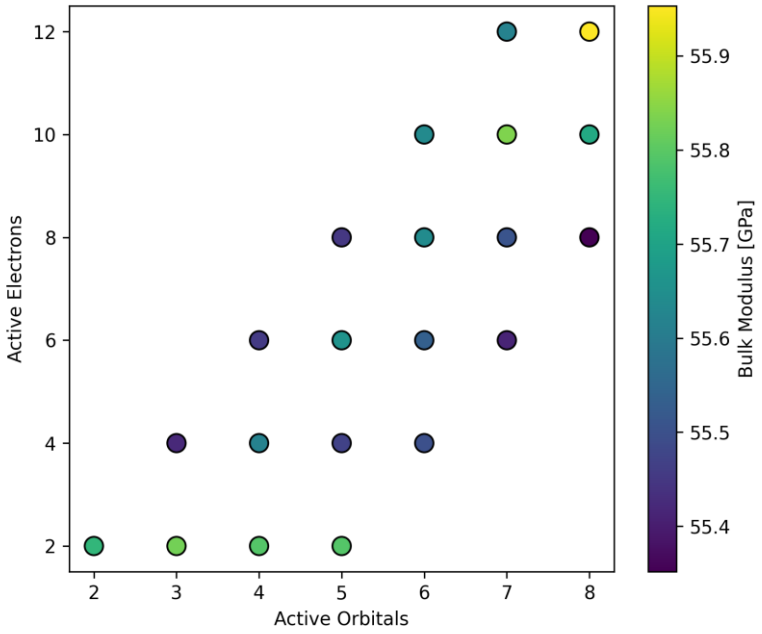


Figure 3.3.13: Influence of the active space on the bulk modulus of a 2×2×2 MgH_2 supercell with a central substituted iron atom.

Since the bulk modulus (and the equilibrium volume) remains largely insensitive to the active space size, the inclusion of overlapping bands does not seem to dictate the presence or absence of overbinding. This could imply that the overlapping levels are too far from the Fermi level to exert a meaningful influence. On the other hand, the difference from the results of previous structures can originate from the structure itself, as this is the first structure featuring a potentially strongly correlated transition metal atom (Fe). Ultimately, while this structure serves as a counterexample to previous deviations, it does not clearly pinpoint the source of the overbinding.

Despite all these investigations into the variable parameters of the Dopyqo and DFT calculations and the wide variety of materials calculated, no source of the error could be found. This suggests that the error originates in the implementation of Dopyqo itself. In addition to the FCI energy, Dopyqo also calculates the HF energy, i.e., the ground-state energy based on a single Slater determinant. If the HF energy is used to perform EoS fits, the same overbinding occurs as seen in FCI-based calculations. This proves that the error is not related to static correlation, since the HF energy ignores it. Another indication of the origin of the overbinding is provided by the calculations of Bader charges by Schultheis et al.[12]. Bader charges are used to quantify the distribution of electrons within a molecule or a crystal by dividing the total electronic charge density into regions assigned to individual atoms. Because Bader charges are calculated directly from the real-space electronic density, which in turn is derived from the many-body wave function Ψ, they serve as a sensitive probe for the accuracy of the electronic part of the Hamiltonian. The paper demonstrates that Dopyqo produces accurate Bader charges that match or improve upon reference data. The electronic Hamiltonian used within Dopyqo (Equation (2.4.15)) consists of two distinct parts, which are the electronic operators (the sums involving creation and annihilation operators) and scalar constants $E_{\mathrm{n-n}}$, $E_{\mathrm{e-self}}$ and E_{frozen}. The shape of the many-body wave function Ψ is determined solely by the operators within the Hamiltonian, as the variational solver optimizes the state to minimize the expectation value of these interactions. Adding scalar constants to

a Hamiltonian shifts the energy eigenvalues (the total energy) but mathematically does not change the eigenvectors (the wave functions). Now, since the calculated Bader charges are correct, this proves that the electronic operators are correctly describing the distribution of electrons. This means the error causing overbinding may have its origin in the terms that affect the total energy without changing the wave function, which are the added scalar energies. It is possible that one of the existing energy terms causes the error, or that additional energy input is required, which has been overlooked until now.

To investigate the energy deviation of Dopyqo-based calculations, the energy difference from the corresponding DFT-based calculation is determined for each data point. For the best possible comparability with DFT, the HF energy was used since it neglects static correlation, as DFT does. These calculations were performed for diamond, Al, and NaCl. For increased accuracy, 21 data points were calculated instead of the default nine. The representative energy differences for NaCl are shown in Figure 3.3.14.

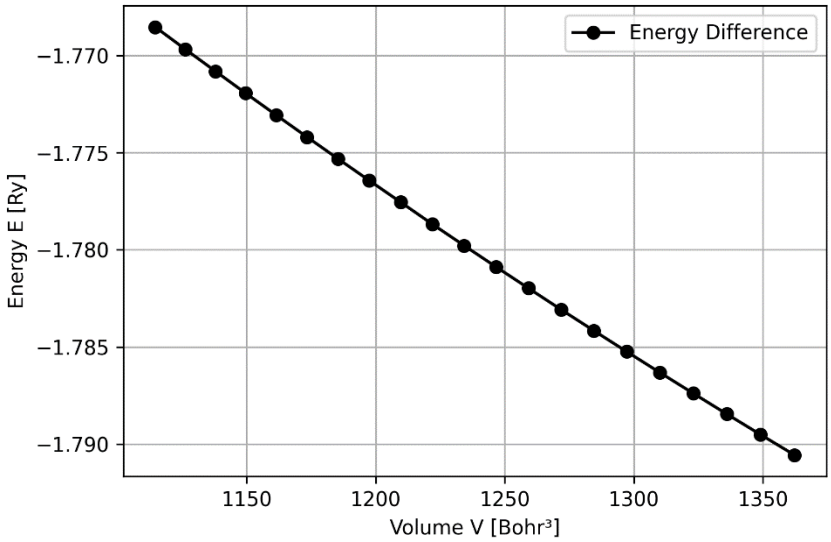


Figure 3.3.14: Energy differences between DFT-based total energies and Dopyqo-based HF energies for NaCl.

Since the absolute value of the total energy is not relevant for the quantities of interest (the bulk modulus and the equilibrium volume), the relevant variables are the slope and curvature of the energy difference curve in Figure 3.3.14. The slope causes a shift in the curve in relation to the volume. Therefore, the equilibrium volume is adjusted. The curvature, i.e., the second derivative of the energy with respect to the volume of the energy difference, directly influences the bulk modulus via Equation (3.1.1). This energy correction could be achieved by adding an energy term of the shown slope and curvature or by removing an energy term of the negative slope and curvature. It is also theoretically possible to perform this correction by introducing a factor c, such that

$$E_{\mathrm{cor}} = E_{\mathrm{total}} + c \cdot E_{\mathrm{sc}} \,, \tag{3.3.1}$$

where E_{total} is the total energy calculated by Dopyqo, E_{sc} is one of Dopyqo's scalar energy terms, and E_{cor} is the corrected total energy. It was investigated whether one of the existing scalar energy terms ($E_{\mathrm{n-n}}$, $E_{\mathrm{e-self}}$ and E_{frozen}) matches the required energy correction. This could provide indications of a possible coefficient error in Dopyqo's existing calculations. The Ewald energy $E_{\mathrm{n-n}}$, the electron self-

energy $E_{\mathrm{e-self}}$, and the frozen core energy E_{frozen} were extracted from Dopyqo. The comparison of these energies with the energy difference is shown in Figure 3.3.15 for NaCl.

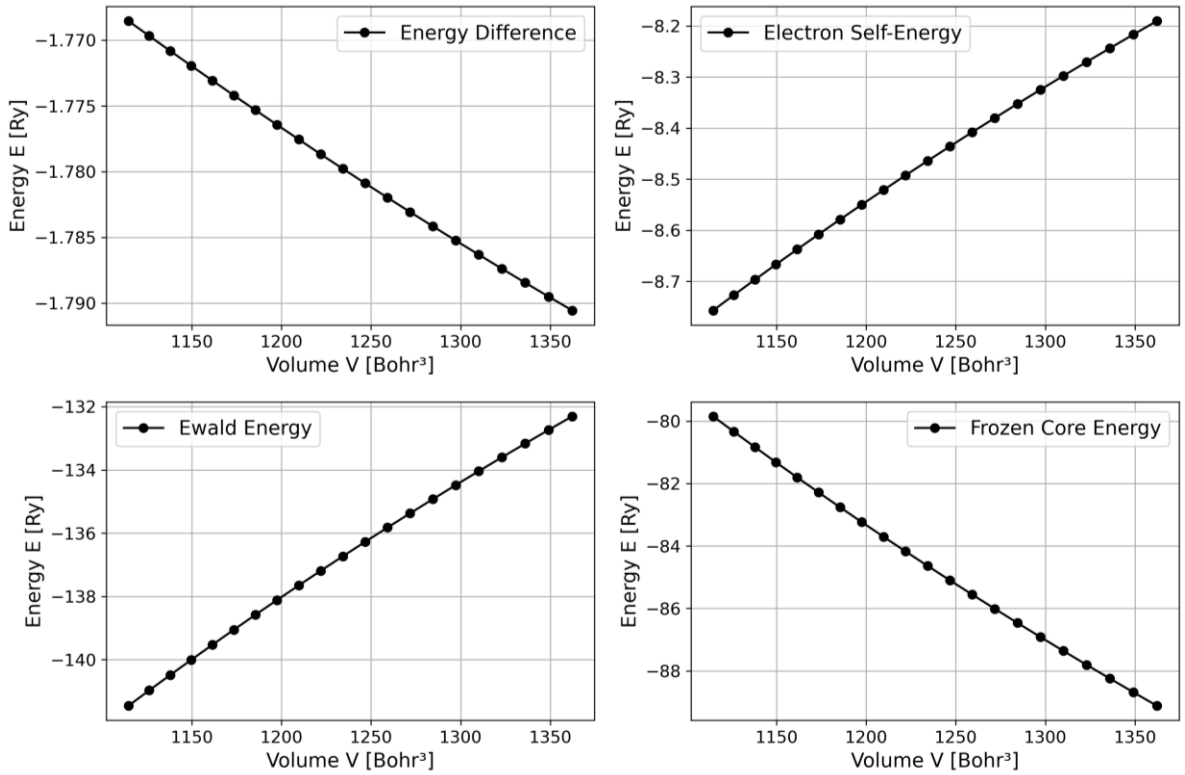


Figure 3.3.15: Comparison of the scalar energy contributions to the sought-after energy for NaCl. Top-left: Energy differences between DFT-based total energies and Dopyqo-based HF energies for NaCl. Top-right: Ewald energy of NaCl for different volumes. Bottom-left: Electron self-energy of NaCl for different volumes. Bottom-right: Frozen core energy of NaCl for different volumes.

Since all of these energies have a linear volume-dependency with slight curvature, just like the energy difference, each one could satisfy Equation (3.3.1). As a more accurate measure of the alignment with the energy sought, the first and second derivatives of the energy difference and the respective energy term need to be compared. For this purpose, the method shown in Code Snippet 3.3.1 is used.

```
    def factor_and_squares_first_derivative(y_ref, y_dat):

        # Derivative (slope)
        dy_ref = np.gradient(y_ref)
        dy_dat = np.gradient(y_dat)

        # global factor for the slope
        a = np.dot(dy_ref, dy_dat) / np.dot(dy_dat, dy_dat)

        # Residues
        res = dy_ref - a * dy_dat

        # Root mean square error
        mse = np.mean(res**2)
        rmse = np.sqrt(mse)

        return a, rmse
```

Code Snippet 3.3.1: Method for calculating a correction factor based on the slopes of the reference energy curve and the data energy curve.

This method calculates the slope at each data point of the two given data sets using the method `gradient()` from the `NumPy` Python library. These slopes are then used to calculate a regression factor which maps the given data set (energies of the examined scalar term) to the reference data set (Dopyqo-DFT energy difference) in the best possible way. Additionally, the root mean square error (RMSE) is calculated. An equivalent method is used for the second derivative using `gradient(gradient())`. These methods benefit greatly from the use of a larger number of data points. The resulting factors for the materials examined can be seen in Table 3.3.2.

Table 3.3.2: Calculated first derivative and second derivative factors for Al, diamond, and NaCl.

	Ewald Energy Factor		**Electron Self-Energy Factor**		**Frozen Core Energy Factor**	
	First Derivative	Second Derivative	First Derivative	Second Derivative	First Derivative	Second Derivative
Al	-0.0011	-0.0029	-0.0196	-0.0001	0.0011	0.0001
Diamond	-0.0177	0.0174	-0.1346	0.1322	0.0199	-0.0149
NaCl	-0.0024	-0.0014	-0.0388	-0.0238	0.0023	0.0014

All factors had an RMSE smaller than 10^{-3}. Ideally, the energy sought should have the same factor for the first and second derivatives. However, it should be noted that a slightly different result compared to DFT is to be expected, since the many-body post-processing aims to improve upon DFT. Accordingly, the calculated factors and RMSE values are only indicative. The respective factors differ depending on the material. This shows that a constant scale factor error, as defined in Equation (3.3.1), cannot be the source of the deviation, but it does not rule out a system-dependent factor. In addition, the two factors for first and second derivative do not align for any of the energies. It is therefore unlikely that a coefficient error in one of these three terms is the source of the error sought. The factors based on Code Snippet 3.3.1 can generally be used to verify the accordance of constant energy terms with the underlying energy difference.

At this stage, the investigation into the observed overbinding was concluded, given the constraints of the thesis's timeline and the absence of clear indications regarding the error's origin.

4 LUCJ Ansatz

As stated in Section 2.5, the trial state $|\Psi(\boldsymbol{\Theta})\rangle$ of a VQE iteration, as well as the parameterized quantum circuit $U(\boldsymbol{\Theta})$ generating it, are both referred to as the ansatz[2]. Since this work contributes to the computational framework Dopyqo, designed for quantum hardware of the noisy intermediate-scale quantum (NISQ) era, the choice of an appropriate ansatz is essential for its application potential[2, 6]. The local unitary cluster Jastrow (LUCJ) ansatz was introduced as an addition to the previously used unitary coupled cluster ansatz with single and double excitations (UCCSD). Since the LUCJ was developed to reduce the error rate of the ansatz circuit, this chapter compares the results and quantum circuits generated by the two ansatzes. Section 4.1 explores the theoretical background of VQE ansatzes, after which Section 4.2 describes the computational implementation of the LUCJ ansatz within Dopyqo. Section 4.3 provides an analysis of calculations based on the LUCJ Ansatz, comparing its accuracy and quantum resource requirements against the UCCSD method. Section 4.4 concludes the chapter by summarizing the advantages and disadvantages of the LUCJ ansatz for near-term quantum computations.

4.1 Theoretical Background of the Ansatzes

The VQE's ansatzes can be divided into two categories: hardware-efficient and chemically-inspired[2]. Hardware-efficient ansatzes are characterized by repeated, dense blocks of a limited selection of parameterized gates that are easy to implement with the available hardware. They aim to develop a flexible trial state with low circuit depth, i.e., a short path of sequential gates from the input to the output. This makes hardware-efficient ansatzes especially suited for quantum hardware of the NISQ era, which has short coherence times and constrained gate topologies. Even though hardware-efficient ansatzes seem beneficial for reducing errors in near-term calculations on quantum hardware, they seem unlikely to be suitable for larger systems, since they take into account no details of the simulated chemical system[2, 107]. As a result, these ansatzes do not naturally conserve the number of particles and yield a very flat energy surface, making classical optimization extremely difficult[108, 109]. Both of these problems tend to worsen with increasing system size. In contrast to this, chemically-inspired ansatzes result from adapting classical computational chemistry algorithms to run efficiently on quantum hardware[2]. As a result, these ansatzes provide physically more accurate descriptions that are easier to interpret. However, the circuit depth and number of parameters increase significantly with the number of electrons, which impairs optimization efficiency and limits scalability to larger systems[6, 107].

The chemically-inspired UCCSD ansatz[110], which is the default in Dopyqo, originates from the classical coupled cluster (CC) method[111] and is defined as

$$|\Psi\rangle = e^{T-T^\dagger}|\Phi_0\rangle\,. \tag{4.1.1}$$

Here, $|\Phi_0\rangle$ defines the reference determinant chosen as the initial state, often the HF state, i.e., the Slater determinant where the first N orbitals are occupied and the rest are unoccupied, while N is the total number of electrons[6, 112]. The excitation operator

$$T = T_1 + T_2 \tag{4.1.2}$$

consists of the single and double excitation operators

$$T_1 = \sum_{i \in \text{virt}, \alpha \in \text{occ}} t_{i\alpha}\, a_i^\dagger a_\alpha \tag{4.1.3}$$

and

$$T_1 = \sum_{i,j \in \text{virt}, \alpha,\beta \in \text{occ}} t_{ij\alpha\beta}\, a_i^\dagger a_j^\dagger a_\alpha a_\beta\,, \tag{4.1.4}$$

containing the excitation amplitudes t, which are the parameters that need to be optimized. The indices correspond to orbitals in the state $|\Phi_0\rangle$, where occ denotes occupied orbitals and virt denotes unoccupied (virtual) orbitals. The fact that the various excitation operators in the UCCSD ansatz do not commute, while the hardware can only execute the operators sequentially, necessitates the use of the Lie-Trotter-Suzuki decomposition[113], commonly referred to as Trotterization, to achieve the sequential execution of these parallel processes[2, 114]. This results in an increased circuit depth for each chosen decomposition step (Trotter step). For a detailed explanation of Trotterization, see [114]. The UCCSD ansatz vastly exceeds the capabilities of contemporary quantum devices in the NISQ era, leading to the design of more compact ansatzes, one of which is the local unitary cluster Jastrow (LUCJ) ansatz[6].

The LUCJ ansatz is a modification of the unitary cluster Jastrow (UCJ) ansatz[115] which is a unitary variant of the cluster Jastrow form introduced by Neuscamman[6, 116]. It has been shown, that the UCJ yields promising accuracy with reduced circuit depths compared to UCCSD[115, 117]. It has the form of a product of L layers

$$|\Psi\rangle = \prod_{\mu=0}^{L-1} \mathcal{U}_\mu\, e^{i\mathcal{J}_\mu} \mathcal{U}_\mu^\dagger |\Phi_0\rangle\,, \tag{4.1.5}$$

where each $\mathcal{U}_\mu$ is an orbital rotation and each $\mathcal{J}_\mu$ is a diagonal Coulomb operator

$$\mathcal{J} = \frac{1}{2} \sum_{ij,\sigma\tau} J_{ij}^{\sigma\tau}\, n_{i,\sigma} n_{j,\tau}\,. \tag{4.1.6}$$

Here, $J_{ij}^{\sigma\tau}$ are the coefficient matrices and $n_{i,\sigma} = c_{i,\sigma}^\dagger c_{i,\sigma}$ is the occupation number operator for spatial orbital i and spin σ[6, 118]. For derivation and further details, see [115] and [6]. Since this ansatz is already defined as a product of sequential layers where the internal diagonal Jastrow terms naturally commute, it can be implemented directly on quantum hardware as a series of gates without requiring Trotterization[6]. While the UCJ ansatz has many desirable properties, its implementation on contemporary quantum devices is challenging due to its requirement of all-to-all qubit connectivity, or the use of a network of error-prone SWAP gates[6]. As currently only trapped ion architectures have all-to-all connectivity, a “local” approximation is introduced to the UCJ ansatz by imposing sparsity constraints on the matrices $J_{ij}^{\sigma\tau}$, which allow them to be implemented in constant depth on qubit

topologies with limited connectivity. The constraints are specified by a list of the qubit indices, i.e., a list of orbitals and spins allowed to interact. As an example, Figure 4.1.1 shows the connectivity constraints for a square lattice qubit topology.

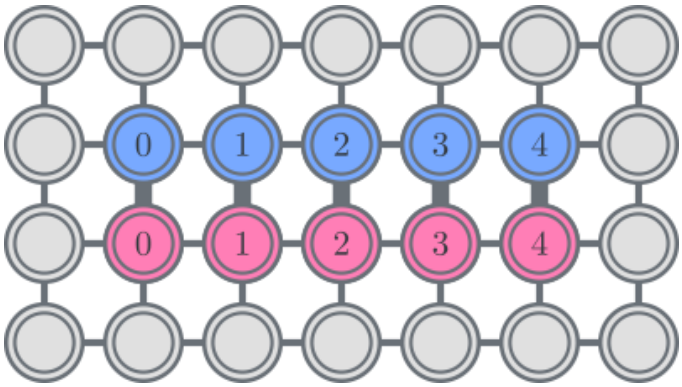


Figure 4.1.1: Connectivity constraints of a square lattice qubit topology. Gray and colored circles indicate qubits. Qubits marked in red or blue have the same spin, respectively. The number inside the circle denotes the index of the spatial orbital. Black lines connect adjacent qubits, while thicker lines indicate the connection of qubits associated with the same spatial orbital[6].

This configuration utilizes a ladder-like topology where same-spin orbitals are linked in linear chains and inter-spin connectivity is restricted to qubits sharing the same spatial orbital index[118]. Because of these constraints, the resulting LUCJ ansatz is less expressive, leading to the potential requirement of more ansatz repetitions, i.e., more layers L. The LUCJ ansatz is considered to lie between the extremes of chemically-inspired and hardware-efficient ansatzes[6]. It has been shown that the LUCJ ansatz can yield better accuracy than UCCSD while having a lower circuit depth[6].

4.2 Computational Implementation

The LUCJ ansatz was introduced as an extension of Dopyqo's existing internal method `solve_vqe_qiskit()`. This method serves the purpose of calculating the ground state of the system based on the VQE using the Qiskit Python library. Qiskit is the most widely used open-source software development kit for quantum computing[76]. Developed primarily by IBM, it allows for the design and execution of quantum programs either on real quantum hardware or by simulation using classical hardware. Code Snippet 4.2.1. shows the arguments of the method `solve_vqe_qiskit()`.

```
    def solve_vqe_qiskit(
        self,
        optimizer: dopyqo.VQEOptimizers = dopyqo.VQEOptimizers.L_BFGS_B,
        UCCSD_reps: int | None = 1,
        lucj_reps: int | None = 10,
        mapper: FermionicMapper | None = JordanWignerMapper(),
        maxiter: int | None = None,
        excitations_UCCSD: list[tuple[int, ...]] | dopyqo.ExcitationPools =
        dopyqo.ExcitationPools.SINGLES_DOUBLES,
        ansatz_type: str | None = "UCCSD",
        lucj_pairs_aa: list[tuple[int, int]] | None = None,
        lucj_pairs_ab: list[tuple[int, int]] | None = None
    ) -> VQEResult:
```

Code Snippet 4.2.1: Arguments of `solve_vqe_qiskit()`.

In this work, the JW mapping and the L_BFGS_B optimizer are used for the VQE, which are the default values of the method. The value `maxiter` sets the maximum number of optimizer iterations before stopping. `UCCSD_reps` defines the number of Trotter steps used for the UCCSD ansatz. Since it has been shown that a single Trotterization step can yield accurate results[2], this work focuses on calculations with a `UCCSD_reps` value of 1. The argument `ansatz_type` was added to specify the ansatz used. Three additional arguments were introduced for defining LUCJ parameters. `lucj_reps` sets the number of layers L, defined in Equation (4.1.5). Through the arguments `lucj_pairs_aa` and `lucj_pairs_ab`, the user can define the allowed interactions between qubits. For this purpose, two lists of tuples are used, each containing two integers referring to specific qubits. These allowed interactions are split into two lists based on interactions of parallel and opposing spins. If no interactions are set, a square lattice topology is set as the default. This choice was made since the newest quantum processor of the biggest quantum computer cloud provider, IBM, is based on a square lattice topology[65]. This topology is also used for all calculations in Section 4.3.

A value validation for the added arguments was implemented at the start of `solve_vqe_qiskit()`. The following preparation section, initializing the mapping, Hamiltonian, estimator, and initial state, was left unchanged. For the calculation, the code splits based on the chosen ansatz. For UCCSD, the code is unchanged with the exception of added instance attributes storing the number of qubits, the number of optimization parameters, the circuit depth, and the number of gates, which are used for comparing both ansatzes. For the UCCSD, the Qiskit-native `UCCSD`-object is used and then evaluated using Qiskit's method `VQE()`. However, since Qiskit does not include an implementation of the LUCJ ansatz, the Python library ffsim is used. The LUCJ implementation of ffsim is incompatible with Qiskit's VQE-method. For this reason, the method `minimize()` from the Python library SciPy is used to minimize the cost function shown in Code Snippet 4.2.2, corresponding to a VQE.

```
    def cost_func(params):
        nonlocal eval_count

        # 1. Load parameters in ffsim operator
        current_op = ffsim.UCJOpSpinBalanced.from_parameters(
            params,
            norb=norb,
            n_reps=lucj_reps,
            interaction_pairs=(pairs_aa, pairs_ab)
        )

        # 2. Generate Qiskit QuantumCircuit
        qc = QuantumCircuit(qubits)
        qc.append(ffsim.qiskit.PrepareHartreeFockJW(norb, nelec), qubits)
        qc.append(ffsim.qiskit.UCJOpSpinBalancedJW(current_op), qubits)
        # Decompose blocks into Standard-Qiskit-Gates
        qc = qc.decompose().decompose()

        # 3. Evaluate with Qiskit estimator
        job = estimator.run([qc], [pauli_sum_op])
        mean_energy = job.result().values[0]

        # 4. Storing and printing calculation progress
        eval_count += 1
        print(f"Optimizer evaluation #{eval_count}, Diff. to ref.:
        {np.abs(mean_energy + offset - self.reference_energy)}", end="\r")
        counts.append(eval_count)
        values.append(mean_energy)

        return mean_energy
```

Code Snippet 4.2.2: Cost function of the VQE implementation for the LUCJ ansatz.

As `solve_vqe_qiskit()` currently does not support spin-polarized calculations, the LUCJ ansatz was implemented based on the `UCJSpinBalanced`-object, which is initialized with new parameters for each iteration. For the first iteration, some random noise is added to the initial parameters, since the gradient at the HF state is zero for the LUCJ ansatz[6]. Without this noise, the optimizer does not change the parameters. To ensure the best possible comparability with the UCCSD ansatz, the `UCJSpinBalanced`-object is used to initialize a Qiskit quantum circuit, which is then evaluated by the estimator. Since `solve_vqe_qiskit()` saves Qiskit objects as results, which are used by other methods, imitation objects are created from the results of the LUCJ-VQE.

4.3 Comparison of UCCSD- and LUCJ-based Calculations

In this section, UCCSD- and LUCJ-based VQE calculations executed for NaCl and cubic diamond are compared based on the results and various calculation metrics. All calculations were performed on the same hardware as described in Section 3.3. As an example for a VQE convergence, Figure 4.3.1 shows the convergence of UCCSD and LUCJ-based calculations for NaCl with an active space of 4e4o.

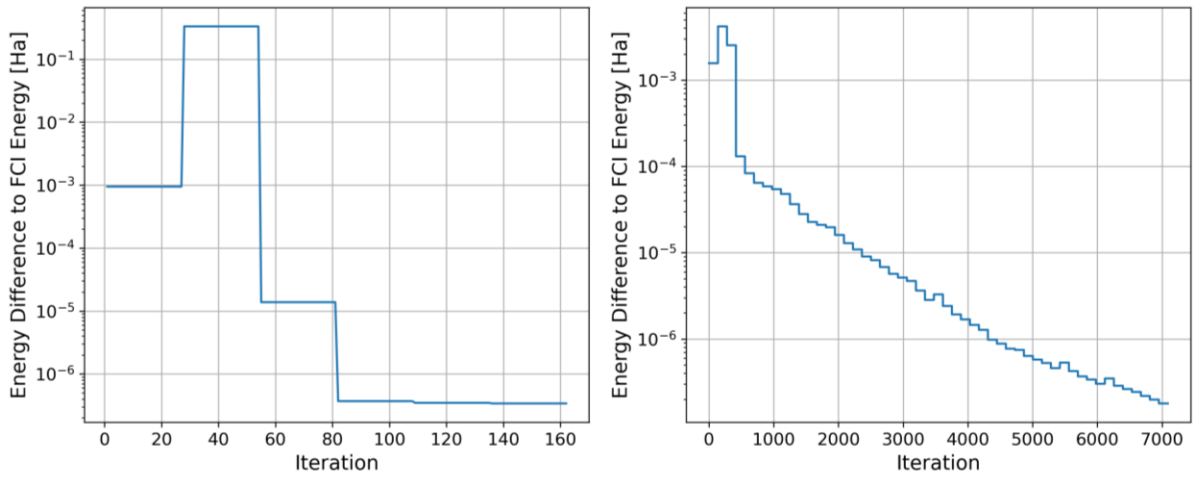


Figure 4.3.1: Convergence of the energy difference of the VQE result to the FCI energy for NaCl with the UCCSD ansatz (left) and the LUCJ ansatz (right). The active space for both calculations is 4e4o and the number of layers for the LUCJ ansatz is $L = 6$.

Based on the iteration count and the final energy difference relative to the ideal value, i.e., the FCI energy, it is clear that the optimizer has significantly greater difficulty finding the energy minimum for the LUCJ ansatz. This trend was observed for all of the calculations performed. As mentioned above, random noise was added to the initial parameters of the LUCJ ansatz. To minimize the impact of this randomness, LUCJ-based VQE calculations were performed five times each. All values and plots shown below are based on the average of these calculations.

The essential parameter for tuning the LUCJ ansatz is the number of layers L (`lucj_reps`), defined in Equation (4.1.5). The influence of L on the energy difference to the FCI energy and the calculation time is shown in Figure 4.3.2 for NaCl with an active space of 4e4o.

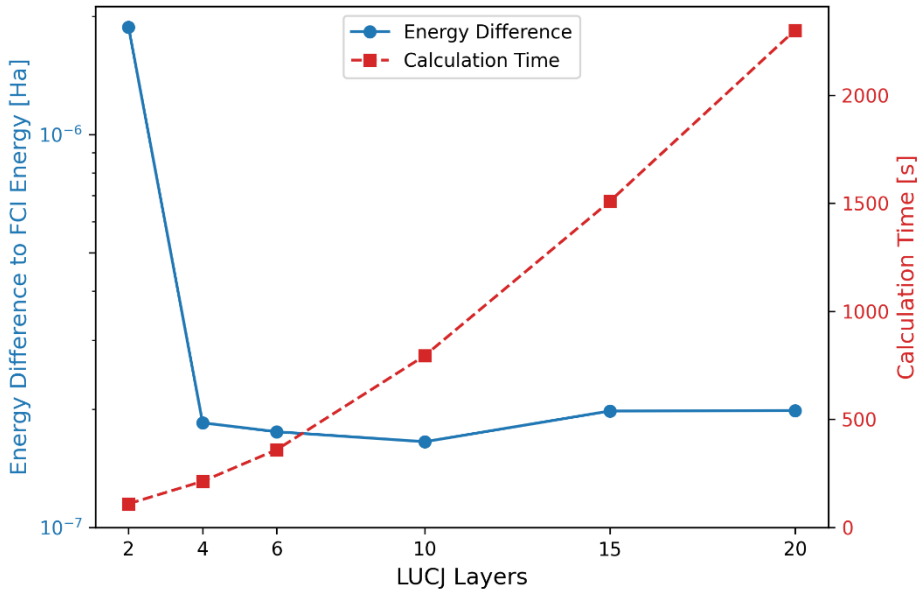


Figure 4.3.2: Energy difference of the VQE result to the FCI energy and VQE calculation time depending on the number of LUCJ layers L for NaCl with an active space of 4e4o.

Here, the energy is converged from $L = 4$ onwards, while the processing time continues to increase. This behavior is identical for diamond. For calculations involving larger active spaces than 6e6o, the computation times for $L > 6$ become too large for the limited timeframe of this thesis. As a representative selection of the computationally accessible range, calculations were performed with the active spaces 2e2o, 4e4o, and 6e6o for $L = 4$ and $L = 6$. The resulting energy differences to FCI are shown in Table 4.3.1.

Table 4.3.1: Energy differences to FCI for UCCSD- and LUCJ-VQE calculations.

		Energy Difference to FCI [Ha]		
		2e2o	4e4o	6e6o
NaCl	UCCSD	2.84×10^{-14}	1.04×10^{-10}	1.14×10^{-10}
	LUCJ ($L = 4$)	2.28×10^{-9}	1.34×10^{-7}	6.65×10^{-7}
	LUCJ ($L = 6$)	5.17×10^{-9}	2.26×10^{-7}	7.51×10^{-7}
Diamond	UCCSD	7.08×10^{-11}	3.40×10^{-7}	2.27×10^{-6}
	LUCJ ($L = 4$)	3.57×10^{-9}	3.10×10^{-7}	3.38×10^{-6}
	LUCJ ($L = 6$)	1.87×10^{-9}	5.55×10^{-7}	3.14×10^{-6}

As expected, the more sophisticated UCCSD ansatz provides more accurate results, with the sole exception of an outlier at $L = 4$ for 4e4o in diamond. While the energy deviation of the LUCJ ansatz decreases with an increasing number of layers, even the results for $L = 4$ remain well within chemical accuracy. Generally, the deviation from the FCI energy increases with growing active spaces, while at the same time, the discrepancy between UCCSD and LUCJ decreases. While the results for diamond show a smaller discrepancy between UCCSD and LUCJ in general, the underlying trends remain consistent. The LUCJ ansatz generates promising results in terms of accuracy, but requires significantly more VQE iterations and, consequently, a significantly longer computation time, as shown in Table 4.3.2.

Table 4.3.2: Calculation metrics concerning the convergence of the optimizer for UCCSD- and LUCJ-VQE calculations.

		Circuit Parameter Count			Iterations			Calculation Time [s]		
		2e2o	4e4o	6e6o	2e2o	4e4o	6e6o	2e2o	4e4o	6e6o
NaCl	UCCSD	3	26	117	16	135	708	0.3	20.8	1005.5
	LUCJ ($L = 4$)	28	92	188	476	3720	20261	13.5	210.8	2322.1
	LUCJ ($L = 6$)	42	138	282	636	5365	27168	20.2	360.4	4121.1
Diamond	UCCSD	3	26	117	24	162	944	0.5	24.6	1258.3
	LUCJ ($L = 4$)	28	92	188	470	6194	57267	14.1	388.7	8543.2
	LUCJ ($L = 6$)	42	138	282	731	7256	58921	24.7	542.1	12885

It is clear that both the number of iterations and the computation time for the LUCJ approach are many times higher, regardless of the size of the active space. Note that these execution times are significantly elevated in comparison to potential applications on quantum hardware, as the classical simulation of quantum circuits is inherently inefficient. As expected, an increase in the LUCJ layer count also increases all three metrics shown in Table 4.3.2. Since the circuit parameter count is significantly higher for the LUCJ ansatz compared to the UCCSD ansatz, the optimizer has significantly greater difficulty finding the global minimum. As the UCCSD ansatz includes all possible single- and double-excitations, while the LUCJ ansatz constrains the excitations based on topology, the relative trend in the number of circuit parameters and, consequently, the relative trend in the number of iterations and computation times should reverse for large systems in favor of the LUCJ ansatz. As stated in Chapter 4.1, the energy landscape for hardware-efficient ansatzes tends to be much flatter compared to chemically-inspired ansatzes like UCCSD. This could potentially be a reason for the slow convergence of the optimizer towards the global minimum. Based on these results, it can be assumed that the optimizer generally plays a crucial role in the efficiency of the VQE, especially for the LUCJ ansatz. Consequently, the choice of the optimizer, as well as the tuning of its convergence parameters, are essential factors for future research.

Since the LUCJ ansatz aims to generate results with the lowest possible error rate on near-term quantum hardware, the circuit depth (i.e., the number of layers of parallel-executable gates) and the number of CNOT gates, which are particularly prone to errors, are important metrics for the comparison to the UCCSD ansatz[6]. Table 4.3.3 shows a comparison of the relevant metrics of the quantum circuits. These values are identical for NaCl and diamond.

Table 4.3.3: Calculation metrics concerning the quantum circuits for UCCSD- and LUCJ-VQE calculations.

		Circuit Depth			Number of CNOT Gates			Number of Gates		
		2e2o	4e4o	6e6o	2e2o	4e4o	6e6o	2e2o	4e4o	6e6o
NaCl / Diamond	UCCSD	89	1871	12405	49	1257	9082	181	3251	19273
	LUCJ ($L = 4$)	67	357	531	44	272	608	185	1542	3650
	LUCJ ($L = 6$)	93	532	797	64	408	912	260	2302	5446

The circuit depth and gate counts remain identical across different materials because the structure of both the UCCSD and LUCJ ansatzes is determined solely by the size of the active space, rather than the specific chemical properties of the system. Instead of altering this fixed mathematical gate layout, the unique chemical identity of a material is encoded entirely within the material-specific Hamiltonian. For all three metrics, the increase in the active space leads to a substantial increase in value for the UCCSD ansatz, greatly exceeding the scaling of the LUCJ-based circuits. The two metrics most important for the error rate of a circuit, i.e., the circuit depth and the number of CNOT gates, are plotted for a wider range of active spaces in Figure 4.3.3.

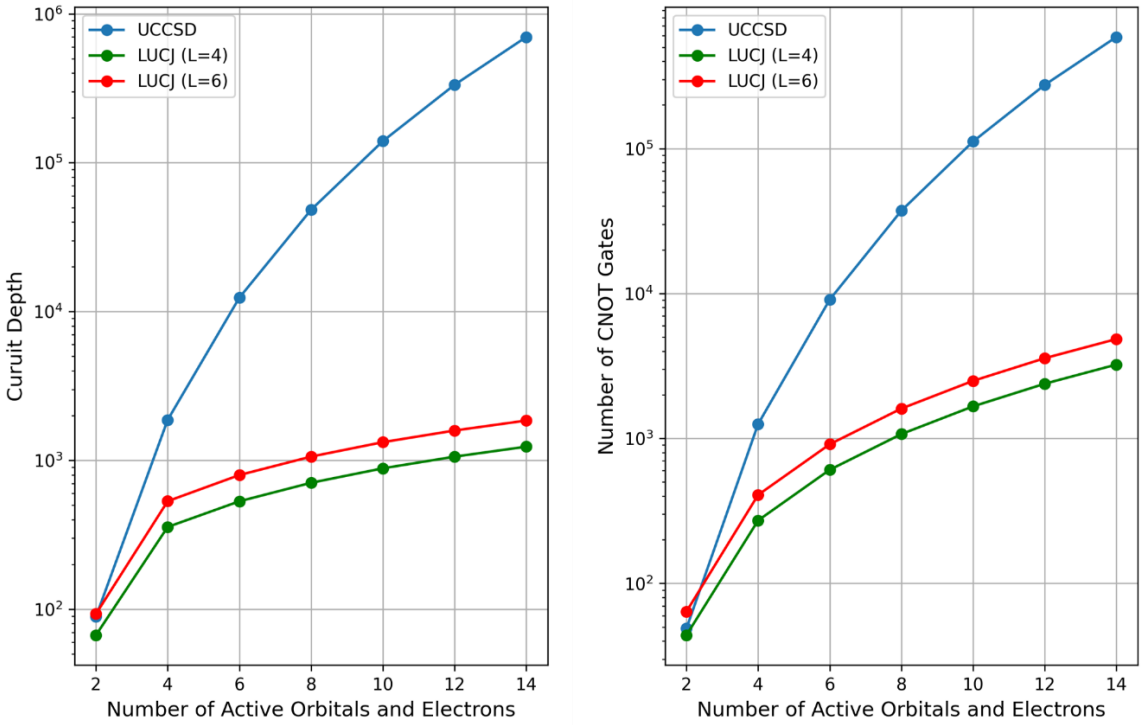


Figure 4.3.3: Circuit depth (left) and number of CNOT gates (right) depending on the active space for VQE calculations with the UCCSD and LUCJ ansatzes.

As a result of the much slower growth in circuit size, it can be assumed that the LUCJ ansatz has significantly greater potential for application on quantum hardware in the NISQ era compared to the UCCSD approach, especially for the calculation of larger systems.

5 Conclusions and Outlook

It has been shown that bulk moduli calculated with Dopyqo's many-body post-processing approach experience strong overbinding across a range of different materials. The calculations performed demonstrate that the error is independent of all examined parameters for DFT and Dopyqo. Despite investigating all relevant parameters for convergence, as well as different approaches to finding the origin of the overbinding, no clear indications were found. This suggests that the source of the overbinding likely lies within the implementation of Dopyqo itself, potentially due to a numerical discrepancy or a fundamental conceptual error. Furthermore, a physical omission remains a possibility, such as the absence of a constant energy contribution as mentioned in Section 3.3. The existing energy terms were compared to the DFT-Dopyqo energy difference, which did not yield any clear indications towards the error's origin. For further analysis of energy contributions, the method presented in Code Snippet 3.3.1 can be used as a benchmark for assessing compliance with the underlying deviation. To verify whether the absence of overbinding for the MgH_2-based supercell is attributable to the isolated nature of the bands near the Fermi energy, finding and analyzing systems with similar band structures could lead to new insights. Additionally, it must be considered whether the underlying concept of Dopyqo, the calculation of the many-body Hamiltonian using KS orbitals, is inherently faulty. KS orbitals are optimized by minimizing the total energy based on DFT approximations, such as DFAs. It is conceivable that a systematic error arises when these orbitals are used in the many-body Hamiltonian for which they were not specifically optimized. Optimizing the orbitals directly within the Dopyqo framework would clarify this issue. If the overbinding remains present with Dopyqo-optimized orbitals, the systematic error cannot be attributed to the use of KS orbitals. Furthermore, as Dopyqo relies on frozen core approximations from PPs optimized for specific DFT functionals, a potential inconsistency may arise. This could be investigated by performing Dopyqo-based calculations on a small-scale benchmark system using an all-electron approach. If the systematic error is eliminated in the all-electron treatment, the discrepancy is likely attributable to an incompatibility between standard DFT pseudopotentials and the many-body formalism of Dopyqo. As implementing the aforementioned approaches exceeds the scope of the present thesis, they remain a subject to future research. Nevertheless, the developed Python module and the method presented in Code Snippet 3.3.1 provide a foundation for further research into the observed overbinding and the overall performance of Dopyqo relative to alternative approaches and experimental data. Finding the origin of the overbinding holds the potential to establish Dopyqo as a viable framework for near-term quantum simulations for periodic solids. On the other hand, should the issue persist, or should the concept of using KS orbitals for many-body post-processing turn out to be inherently faulty, using KS orbitals as a superior starting point for orbital optimization based on the many-body approach might be a promising alternative approach for future applications.

As quantum-resource-saving ansatzes play an important role in the utilization of NISQ era quantum hardware for the electronic structure problem, the superior performance of the LUCJ ansatz within the Dopyqo framework in terms of scalability has been proven. While the VQE calculations based on the LUCJ ansatz provide less accurate results than the equivalent UCCSD-based calculations, the deviation is well within chemical accuracy. Given the additional prospect that UCCSD-based VQE calculations for larger systems are likely to encounter excessively high error rates on NISQ era quantum hardware, the

LUCJ ansatz appears to be the better option in the near term. Even though LUCJ-based VQE calculations need vastly more iterations and correspondingly longer calculation times, it is very likely that this ansatz will gain importance since it should be able to calculate useful results for large systems where UCCSD fails. The high number of iterations observed in the VQE calculations could induce further investigations into the convergence behavior of the LUCJ ansatz, potentially improving its application potential. These studies should aim to test the use of various gradient-based and non-gradient-based minimization algorithms and their impact on LUCJ convergence. Additionally, optimizer tuning is an essential part of future research to investigate the influence of optimization parameters on convergence and result accuracy. Given the difficulty of converging LUCJ-based VQE calculations, the approach of choosing the best possible starting point in the form of KS orbitals mentioned above could lead to significant reductions in the number of necessary VQE calculations and, therefore, save substantial computation time for quantum-computer-based orbital optimizations. The implementation of additional hardware-efficient ansatzes and qubit mappings could add further utility to Dopyqo as a research toolkit.

In general, the development of theoretical and computational frameworks for error-prone quantum hardware is an important field of research for generating utility for NISQ era quantum computers. I envision that the computational framework Dopyqo, as well as my contributions to it, will help harness the potential of near-term quantum computers for the benefit of everyone.

References

[1] F. Jensen, *Introduction to computational chemistry*. John wiley & sons, 2017.

[2] S. McArdle, S. Endo, A. Aspuru-Guzik, S. C. Benjamin, and X. Yuan, "Quantum computational chemistry," *Reviews of Modern Physics,* vol. 92, no. 1, p. 015003, 2020.

[3] T. Helgaker, P. Jorgensen, and J. Olsen, *Molecular electronic-structure theory*. John Wiley & Sons, 2013.

[4] A. Jameson, "Re-engineering the design process through computation," *Journal of Aircraft,* vol. 36, no. 1, pp. 36–50, 1999.

[5] J. Pokluda, M. Černý, M. Šob, and Y. Umeno, "Ab initio calculations of mechanical properties: Methods and applications," *Progress in Materials Science,* vol. 73, pp. 127–158, 2015.

[6] M. Motta, K. J. Sung, K. B. Whaley, M. Head-Gordon, and J. Shee, "Bridging physical intuition and hardware efficiency for correlated electronic states: the local unitary cluster Jastrow ansatz for electronic structure," *Chemical Science,* vol. 14, no. 40, pp. 11213–11227, 2023.

[7] C. Kittel and P. McEuen, *Introduction to solid state physics*. John Wiley & Sons, 2018.

[8] R. Evarestov and V. Smirnov, "Special points of the Brillouin zone and their use in the solid state theory," *physica status solidi (b),* vol. 119, no. 1, pp. 9–40, 1983.

[9] C. Cohen-Tannoudji, B. Diu, and F. Laloë, *Quantum mechanics, volume 3: fermions, bosons, photons, correlations, and entanglement*. John Wiley & Sons, 2019, pp. 213–250.

[10] M. Born, & Oppenheimer, R., "Zur Quantentheorie der Molekeln," *Annalen der Physik,* vol. 389, pp. 457–484, 1927.

[11] O. Christiansen, "Selected new developments in vibrational structure theory: potential construction and vibrational wave function calculations," *Physical Chemistry Chemical Physics,* vol. 14, no. 19, pp. 6672–6687, 2012.

[12] E. Schultheis, A. Rehn, and G. Breuil, "Many-body post-processing of density functional calculations using the variational quantum eigensolver for Bader charge analysis," *arXiv preprint arXiv:2510.12887,* 2025.

[13] P. Meredith, C. Bettinger, M. Irimia-Vladu, A. Mostert, and P. E. Schwenn, "Electronic and optoelectronic materials and devices inspired by nature," *Reports on Progress in Physics,* vol. 76, no. 3, p. 034501, 2013.

[14] P. Yu, Y. Zhen, H. Dong, and W. Hu, "Crystal engineering of organic optoelectronic materials," *Chem,* vol. 5, no. 11, pp. 2814–2853, 2019.

[15] O. Ostroverkhova, "Organic optoelectronic materials: mechanisms and applications," *Chemical reviews,* vol. 116, no. 22, pp. 13279–13412, 2016.

[16] K. Burke and L. O. Wagner, "DFT in a nutshell," *International Journal of Quantum Chemistry,* vol. 113, no. 2, pp. 96–101, 2013.

[17] D. R. Hartree and W. Hartree, "Self-consistent field, with exchange, for beryllium," *Proceedings of the Royal Society of London. Series A-Mathematical and Physical Sciences,* vol. 150, no. 869, pp. 9–33, 1935.

[18] A. Szabo and N. S. Ostlund, *Modern quantum chemistry: introduction to advanced electronic structure theory*. Courier Corporation, 2012.

[19] T. Helgaker, S. Coriani, P. Jørgensen, K. Kristensen, J. Olsen, and K. Ruud, "Recent advances in wave function-based methods of molecular-property calculations," *Chemical reviews,* vol. 112, no. 1, pp. 543–631, 2012.

[20] B. Roos, "A new method for large-scale Cl calculations," *Chemical Physics Letters,* vol. 15, no. 2, pp. 153–159, 1972.

[21] K. Raghavachari, G. W. Trucks, J. A. Pople, and M. Head-Gordon, "A fifth-order perturbation comparison of electron correlation theories," *Chemical Physics Letters,* vol. 157, no. 6, pp. 479–483, 1989.

[22] P. Hohenberg and W. Kohn, "Inhomogeneous electron gas," *Physical review,* vol. 136, no. 3B, p. B864, 1964.

[23] W. Kohn, A. D. Becke, and R. G. Parr, "Density functional theory of electronic structure," *The journal of physical chemistry,* vol. 100, no. 31, pp. 12974–12980, 1996.

[24] L. H. Thomas, "The calculation of atomic fields," in *Mathematical proceedings of the Cambridge philosophical society*, 1927, vol. 23, no. 5: Cambridge University Press, pp. 542–548.

[25] E. Fermi, "Eine statistische Methode zur Bestimmung einiger Eigenschaften des Atoms und ihre Anwendung auf die Theorie des periodischen Systems der Elemente," *Zeitschrift für Physik,* vol. 48, no. 1, pp. 73–79, 1928.

[26] E. Teller, "On the stability of molecules in the Thomas-Fermi theory," *Reviews of Modern Physics,* vol. 34, no. 4, p. 627, 1962.

[27] W. Kohn and L. J. Sham, "Self-consistent equations including exchange and correlation effects," *Physical review,* vol. 140, no. 4A, p. A1133, 1965.

[28] U. Von Barth and L. Hedin, "A local exchange-correlation potential for the spin polarized case. i," *Journal of Physics C: Solid State Physics,* vol. 5, no. 13, p. 1629, 1972.

[29] G.-X. Zhang, A. M. Reilly, A. Tkatchenko, and M. Scheffler, "Performance of various density-functional approximations for cohesive properties of 64 bulk solids," *New Journal of Physics,* vol. 20, no. 6, p. 063020, 2018.

[30] J. P. Perdew and K. Schmidt, "Jacob's ladder of density functional approximations for the exchange-correlation energy," in *AIP conference proceedings*, 2001, vol. 577, no. 1: American Institute of Physics, pp. 1–20.

[31] J. P. Perdew, A. Ruzsinszky, J. Tao, V. N. Staroverov, G. E. Scuseria, and G. I. Csonka, "Prescription for the design and selection of density functional approximations: More constraint satisfaction with fewer fits," *The Journal of chemical physics,* vol. 123, no. 6, 2005.

[32] F. Tran, J. Stelzl, and P. Blaha, "Rungs 1 to 4 of DFT Jacob's ladder: Extensive test on the lattice constant, bulk modulus, and cohesive energy of solids," *The Journal of chemical physics,* vol. 144, no. 20, 2016.

[33] J. P. Perdew, K. Burke, and M. Ernzerhof, "Generalized gradient approximation made simple," *Physical review letters,* vol. 77, no. 18, p. 3865, 1996.

[34] A. D. Becke, "A new mixing of Hartree-Fock and local density-functional theories," *Journal of chemical Physics,* vol. 98, no. 2, pp. 1372–1377, 1993.

[35] J. Harl, L. Schimka, and G. Kresse, "Assessing the quality of the random phase approximation for lattice constants and atomization energies of solids," *Physical Review B—Condensed Matter and Materials Physics,* vol. 81, no. 11, p. 115126, 2010.

[36] M. Rossmannek, P. K. Barkoutsos, P. J. Ollitrault, and I. Tavernelli, "Quantum HF/DFT-embedding algorithms for electronic structure calculations: Scaling up to complex molecular systems," *The Journal of Chemical Physics,* vol. 154, no. 11, 2021.

[37] R. Babbush, N. Wiebe, J. McClean, J. McClain, H. Neven, and G. K.-L. Chan, "Low-depth quantum simulation of materials," *Physical Review X,* vol. 8, no. 1, p. 011044, 2018.

[38] C. Hattig, W. Klopper, A. Kohn, and D. P. Tew, "Explicitly correlated electrons in molecules," *Chemical reviews,* vol. 112, no. 1, pp. 4–74, 2012.

[39] L. M. Fraser, W. Foulkes, G. Rajagopal, R. Needs, S. Kenny, and A. Williamson, "Finite-size effects and Coulomb interactions in quantum Monte Carlo calculations for homogeneous systems with periodic boundary conditions," *Physical Review B,* vol. 53, no. 4, p. 1814, 1996.

[40] Y. Shan, J. L. Klepeis, M. P. Eastwood, R. O. Dror, and D. E. Shaw, "Gaussian split Ewald: A fast Ewald mesh method for molecular simulation," *The Journal of chemical physics,* vol. 122, no. 5, p. 054101, 2005.

[41] P. P. Ewald, "Die Berechnung optischer und elektrostatischer Gitterpotentiale," *Annalen der physik,* vol. 369, no. 3, pp. 253–287, 1921.

[42] L. Clinton *et al.*, "Towards near-term quantum simulation of materials," *Nature Communications,* vol. 15, no. 1, p. 211, 2024.

[43] S. Yalouz, E. Koridon, B. Senjean, B. Lasorne, F. Buda, and L. Visscher, "Analytical nonadiabatic couplings and gradients within the state-averaged orbital-optimized variational quantum eigensolver," *Journal of chemical theory and computation,* vol. 18, no. 2, pp. 776–794, 2022.

[44] D. I. Lyakh, M. Musiał, V. F. Lotrich, and R. J. Bartlett, "Multireference nature of chemistry: The coupled-cluster view," *Chemical reviews,* vol. 112, no. 1, pp. 182–243, 2012.

[45] *Dopyqo*. (2025). [Online]. Available: https://github.com/dlr-wf/Dopyqo

[46] A. Aspuru-Guzik, A. D. Dutoi, P. J. Love, and M. Head-Gordon, "Simulated quantum computation of molecular energies," *Science,* vol. 309, no. 5741, pp. 1704–1707, 2005.

[47] R. P. Feynman, "Simulating physics with computers," in *Feynman and computation*: cRc Press, 2018, pp. 133–153.

[48] A. Aspuru-Guzik, R. Lindh, and M. Reiher, "The matter simulation (r) evolution," *ACS central science,* vol. 4, no. 2, pp. 144–152, 2018.

[49] M. Reiher, N. Wiebe, K. M. Svore, D. Wecker, and M. Troyer, "Elucidating reaction mechanisms on quantum computers," *Proceedings of the national academy of sciences,* vol. 114, no. 29, pp. 7555–7560, 2017.

[50] Y. Cao, J. Romero, and A. Aspuru-Guzik, "Potential of quantum computing for drug discovery," *IBM Journal of Research and Development,* vol. 62, no. 6, pp. 6: 1–6: 20, 2018.

[51] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*. Cambridge university press Cambridge, 2001.

[52] S. Bravyi, D. Gosset, and R. König, "Quantum advantage with shallow circuits," *Science,* vol. 362, no. 6412, pp. 308–311, 2018.

[53] D. Deutsch and R. Jozsa, "Rapid solution of problems by quantum computation," *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences,* vol. 439, no. 1907, pp. 553–558, 1992.

[54] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings 35th annual symposium on foundations of computer science*, 1994: Ieee, pp. 124–134.

[55] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, 1996, pp. 212–219.

[56] A. Peruzzo *et al.*, "A variational eigenvalue solver on a photonic quantum processor," *Nature communications,* vol. 5, no. 1, p. 4213, 2014.

[57] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, "The theory of variational hybrid quantum-classical algorithms," *New Journal of Physics,* vol. 18, no. 2, p. 023023, 2016.

[58] W. Ritz, "Über eine neue Methode zur Lösung gewisser Variationsprobleme der mathematischen Physik," 1909.

[59] G. Buchs *et al.*, "The role of quantum computing in advancing scientific high-performance computing: A perspective from the adac institute," *arXiv preprint arXiv:2508.11765,* 2025.

[60] C. D. Bruzewicz, J. Chiaverini, R. McConnell, and J. M. Sage, "Trapped-ion quantum computing: Progress and challenges," *Applied physics reviews,* vol. 6, no. 2, 2019.

[61] F. Flamini, N. Spagnolo, and F. Sciarrino, "Photonic quantum information processing: a review," *Reports on Progress in Physics,* vol. 82, no. 1, p. 016001, 2019.

[62] S. Slussarenko and G. J. Pryde, "Photonic quantum information processing: A concise review," *Applied physics reviews,* vol. 6, no. 4, 2019.

[63] P. Krantz, M. Kjaergaard, F. Yan, T. P. Orlando, S. Gustavsson, and W. D. Oliver, "A quantum engineer's guide to superconducting qubits," *Applied physics reviews,* vol. 6, no. 2, 2019.

[64] M. Kjaergaard *et al.*, "Superconducting qubits: Current state of play," *Annual Review of Condensed Matter Physics,* vol. 11, no. 1, pp. 369–395, 2020.

[65] "IBM Quantum Hardware." IBM. https://www.ibm.com/quantum/hardware (accessed 13.03., 2026).

[66] "Quantum error correction below the surface code threshold," *Nature,* vol. 638, no. 8052, pp. 920–926, 2025.

[67] N. P. Sawaya, M. Smelyanskiy, J. R. McClean, and A. Aspuru-Guzik, "Error sensitivity to environmental noise in quantum circuits for chemical state preparation," *Journal of chemical theory and computation,* vol. 12, no. 7, pp. 3097–3108, 2016.

[68] Y. Li and S. C. Benjamin, "Efficient variational quantum simulator incorporating active error minimization," *Physical Review X,* vol. 7, no. 2, p. 021050, 2017.

[69] K. Temme, S. Bravyi, and J. M. Gambetta, "Error mitigation for short-depth quantum circuits," *Physical review letters,* vol. 119, no. 18, p. 180509, 2017.

[70] S. McArdle, X. Yuan, and S. Benjamin, "Error-mitigated digital quantum simulation," *Physical review letters,* vol. 122, no. 18, p. 180501, 2019.

[71] X. Bonet-Monroig, R. Sagastizabal, M. Singh, and T. O'Brien, "Low-cost error mitigation by symmetry verification," *Physical Review A,* vol. 98, no. 6, p. 062339, 2018.

[72] J. Roffe, "Quantum error correction: an introductory guide," *Contemporary Physics,* vol. 60, no. 3, pp. 226–245, 2019.

[73] W. Li *et al.*, "TenCirChem: An efficient quantum computational chemistry package for the NISQ era," *Journal of Chemical Theory and Computation,* vol. 19, no. 13, pp. 3966–3981, 2023.

[74] Q. Sun *et al.*, "Recent developments in the PySCF program package," *The Journal of chemical physics,* vol. 153, no. 2, 2020.

[75] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Mathematical programming,* vol. 45, no. 1, pp. 503–528, 1989.

[76] IBM. "Qiskit." https://www.ibm.com/quantum/qiskit (accessed 16.03., 2026).

[77] E. Ziambaras and E. Schröder, "Theory for structure and bulk modulus determination," *Physical Review B,* vol. 68, no. 6, p. 064112, 2003.

[78] Y. Yourdshahyan, C. Ruberto, L. Bengtsson, and B. I. Lundqvist, "First-principles calculations on the atomic and electronic structure of κ− Al 2 O 3," *Physical Review B,* vol. 56, no. 14, p. 8553, 1997.

[79] R. Gaudoin and W. Foulkes, "Ab initio calculations of bulk moduli and comparison with experiment," *Physical Review B,* vol. 66, no. 5, p. 052104, 2002.

[80] R. G. Parr, "Density functional theory of atoms and molecules," in *Horizons of Quantum Chemistry: Proceedings of the Third International Congress of Quantum Chemistry Held at Kyoto, Japan, October 29-November 3, 1979*, 1989: Springer, pp. 5–15.

[81] P. Vinet, J. Ferrante, J. H. Rose, and J. R. Smith, "Compressibility of solids," *Journal of Geophysical Research: Solid Earth,* vol. 92, no. B9, pp. 9319–9325, 1987.

[82] H. Eckstein and W. Schattke, "Variational quantum Monte Carlo ground state of lithium on a Slater orbital basis," *Physica A: statistical mechanics and its applications,* vol. 216, no. 1-2, pp. 151–157, 1995.

[83] G. Yao, J. Xu, and X. Wang, "Pseudopotential variational quantum Monte Carlo approach to bcc lithium," *Physical Review B,* vol. 54, no. 12, p. 8393, 1996.

[84] C.-L. Fu and K.-M. Ho, "First-principles calculation of the equilibrium ground-state properties of transition metals: Applications to Nb and Mo," *Physical Review B,* vol. 28, no. 10, p. 5480, 1983.

[85] F. D. Murnaghan, "The compressibility of media under extreme pressures," *Proceedings of the National Academy of Sciences,* vol. 30, no. 9, pp. 244–247, 1944.

[86] J.-P. Poirier, *Introduction to the Physics of the Earth's Interior*. Cambridge University Press, 2000.

[87] M. Hebbache and M. Zemzemi, "Ab initio study of high-pressure behavior of a low compressibility metal and a hard material: Osmium and diamond," *Physical Review B—Condensed Matter and Materials Physics,* vol. 70, no. 22, p. 224107, 2004.

[88] F. Birch, "Finite elastic strain of cubic crystals," *Physical review,* vol. 71, no. 11, p. 809, 1947.

[89] A. J. Jackson, J. M. Skelton, C. H. Hendon, K. T. Butler, and A. Walsh, "Crystal structure optimisation using an auxiliary equation of state," *The Journal of Chemical Physics,* vol. 143, no. 18, 2015.

[90] P. Giannozzi *et al.*, "QUANTUM ESPRESSO: a modular and open-source software project for quantumsimulations of materials," *Journal of physics: Condensed matter,* vol. 21, no. 39, p. 395502, 2009.

[91] M. C. Payne, M. P. Teter, D. C. Allan, T. Arias, and a. J. Joannopoulos, "Iterative minimization techniques for ab initio total-energy calculations: molecular dynamics and conjugate gradients," *Reviews of modern physics,* vol. 64, no. 4, p. 1045, 1992.

[92] G. Francis and M. Payne, "Finite basis set corrections to total energy pseudopotential calculations," *Journal of Physics: Condensed Matter,* vol. 2, no. 19, p. 4395, 1990.

[93] "Input File Description " https://www.quantum-espresso.org/Doc/INPUT_PW.html#id35 (accessed 12.02., 2026).

[94] "matplotlib Backends." https://matplotlib.org/stable/users/explain/figure/backends.html (accessed 14.02., 2026).

[95] D. R. Hamann, "Optimized norm-conserving Vanderbilt pseudopotentials," *Physical Review B—Condensed Matter and Materials Physics,* vol. 88, no. 8, p. 085117, 2013.

[96] F. Birch, "Equation of state and thermodynamic parameters of NaCl to 300 kbar in the high-temperature domain," *Journal of Geophysical Research: Solid Earth,* vol. 91, no. B5, pp. 4949–4954, 1986.

[97] M. Anderson, C. Swenson, and D. Peterson, "Experimental equations of state for calcium, strontium, and barium metals to 20 kbar from 4 to 295 K," *Physical Review B,* vol. 41, no. 6, p. 3329, 1990.

[98] R. G. Greene, H. Luo, and A. L. Ruoff, "Al as a simple solid: high pressure study to 220 GPa (2.2 Mbar)," *Physical review letters,* vol. 73, no. 15, p. 2075, 1994.

[99] *Al Crystal Structure: Datasheet from "PAULING FILE Multinaries Edition – 2022" in SpringerMaterials (https://materials.springer.com/isp/crystallographic/docs/sd_0382975)*. Springer-Verlag Berlin Heidelberg & Material Phases Data System (MPDS), Switzerland & National Institute for Materials Science (NIMS), Japan. [Online]. Available: https://materials.springer.com/isp/crystallographic/docs/sd_0382975

[100] *Ca Crystal Structure: Datasheet from "PAULING FILE Multinaries Edition – 2022" in SpringerMaterials (https://materials.springer.com/isp/crystallographic/docs/sd_0530909)*. Springer-Verlag Berlin Heidelberg & Material Phases Data System (MPDS), Switzerland & National Institute for Materials Science (NIMS), Japan. [Online]. Available: https://materials.springer.com/isp/crystallographic/docs/sd_0530909

[101] *Si Crystal Structure: Datasheet from "PAULING FILE Multinaries Edition – 2022" in SpringerMaterials (https://materials.springer.com/isp/crystallographic/docs/sd_1602690)*. Springer-Verlag Berlin Heidelberg & Material Phases Data System (MPDS), Switzerland & National Institute for Materials Science (NIMS), Japan. [Online]. Available: https://materials.springer.com/isp/crystallographic/docs/sd_1602690

[102] *Diamond (C dia) Crystal Structure: Datasheet from "PAULING FILE Multinaries Edition – 2022" in SpringerMaterials (https://materials.springer.com/isp/crystallographic/docs/sd_1939112)*. Springer-Verlag Berlin Heidelberg & Material Phases Data System (MPDS), Switzerland & National Institute for Materials Science (NIMS), Japan. [Online]. Available: https://materials.springer.com/isp/crystallographic/docs/sd_1939112

[103] *NaCl Crystal Structure: Datasheet from "PAULING FILE Multinaries Edition – 2022" in SpringerMaterials (https://materials.springer.com/isp/crystallographic/docs/sd_1250977)*.

Springer-Verlag Berlin Heidelberg & Material Phases Data System (MPDS), Switzerland & National Institute for Materials Science (NIMS), Japan. [Online]. Available: https://materials.springer.com/isp/crystallographic/docs/sd_1250977

[104] Y.-M. Juan and E. Kaxiras, "Application of gradient corrections to density-functional theory for atoms and solids," *Physical Review B,* vol. 48, no. 20, p. 14944, 1993.

[105] N. Dubrovinskaia, L. Dubrovinsky, R. Caracas, and M. Hanfland, "Diamond as a high pressure gauge up to 2.7 Mbar," *Applied Physics Letters,* vol. 97, no. 25, 2010.

[106] M. Methfessel and A. T. Paxton, "High-precision sampling for Brillouin-zone integration in metals," *physical review B,* vol. 40, no. 6, p. 3616, 1989.

[107] P. K. Barkoutsos *et al.*, "Quantum algorithms for electronic structure calculations: Particle-hole Hamiltonian and optimized wave-function expansions," *Physical Review A,* vol. 98, no. 2, p. 022322, 2018.

[108] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, "Barren plateaus in quantum neural network training landscapes," *Nature communications,* vol. 9, no. 1, p. 4812, 2018.

[109] L. Leone, S. F. Oliviero, L. Cincio, and M. Cerezo, "On the practical usefulness of the hardware efficient ansatz," *Quantum,* vol. 8, p. 1395, 2024.

[110] J. Romero, R. Babbush, J. R. McClean, C. Hempel, P. J. Love, and A. Aspuru-Guzik, "Strategies for quantum computing molecular energies using the unitary coupled cluster ansatz," *Quantum Science and Technology,* vol. 4, no. 1, p. 014008, 2019.

[111] R. J. Bartlett and M. Musiał, "Coupled-cluster theory in quantum chemistry," *Reviews of Modern Physics,* vol. 79, no. 1, pp. 291–352, 2007.

[112] A. Anand *et al.*, "A quantum computing view on unitary coupled cluster theory," *Chemical Society Reviews,* vol. 51, no. 5, pp. 1659–1684, 2022.

[113] H. F. Trotter, "On the product of semi-groups of operators," *Proceedings of the American Mathematical Society,* vol. 10, no. 4, pp. 545–551, 1959.

[114] T. Sadhukhan and C. Bangalore, "Suzuki-Trotter Decomposition," *complexity,* vol. 3, no. 4, p. 1, 2025.

[115] Y. Matsuzawa and Y. Kurashige, "Jastrow-type decomposition in quantum chemistry for low-depth quantum circuits," *Journal of chemical theory and computation,* vol. 16, no. 2, pp. 944–952, 2020.

[116] E. Neuscamman, "The Jastrow antisymmetric geminal power in Hilbert space: Theory, benchmarking, and application to a novel transition state," *The Journal of chemical physics,* vol. 139, no. 19, 2013.

[117] U. Baek *et al.*, "Say no to optimization: A nonorthogonal quantum eigensolver," *PRX Quantum,* vol. 4, no. 3, p. 030307, 2023.

[118] W.-H. Lin, F. Liang, M. Motta, H. Zhang, K. M. Merz Jr, and K. J. Sung, "Improved parameter initialization for the (local) unitary cluster Jastrow ansatz," *arXiv preprint arXiv:2511.22476,* 2025.

Additional Resources

In this work, following generative artificial intelligences were used:

- ChatGPT 5.2 - https://chatgpt.com/
- Google Gemini 3 - https://gemini.google.com/?hl=de

These tools were only used as a supporting tool for understanding complex concepts, finding relevant publications, and finding spelling mistakes in the text.

Appendix

Appendix A: QE Input File: NaCl

```
&CONTROL
  calculation = 'scf'
  etot_conv_thr = 8.0000000000d-05
  forc_conv_thr = 1.0000000000d-04
  outdir = '/home/hage_nc/Masterarbeit/BulkModulus/NaCl/QE_outputs'
  prefix = 'NaCl_scf_5'
  pseudo_dir = '/home/hage_nc/Masterarbeit/Pseudopotentials'
  tprnfor = .true.
  tstress = .true.
  verbosity = 'high'
/
&SYSTEM
  degauss = 2.0000000000d-02
  ecutrho = 200
  ecutwfc = 50
  ibrav = 0
  nat = 8
  nosym = .false.
  ntyp = 2
  occupations = 'fixed'
  nbnd = 40
/
&ELECTRONS
  conv_thr = 1.6000000000d-09
  mixing_beta = 4.0000000000d-01
/
ATOMIC_SPECIES
Cl     35.453 Cl_ONCV_PBE-1.2.upf
Na     22.98977 Na_ONCV_PBE-1.2.upf

CELL_PARAMETERS bohr
10.738854951    0.0000000000   0.000000000
0.0000000000    10.738854951   0.000000000
0.0000000000    0.0000000000   10.738854951

ATOMIC_POSITIONS (crystal)
Na            0.0000000000       -0.0000000000        0.0000000000
Na            0.0000000000        0.5000000000        0.5000000000
Na            0.5000000000        0.0000000000        0.5000000000
Na            0.5000000000        0.5000000000        0.0000000000
Cl            0.0000000000        0.0000000000        0.5000000000
Cl            0.0000000000        0.5000000000        0.0000000000
Cl            0.5000000000       -0.0000000000        0.0000000000
Cl            0.5000000000        0.5000000000        0.5000000000

K_POINTS automatic
6 6 6 1 1 1
```

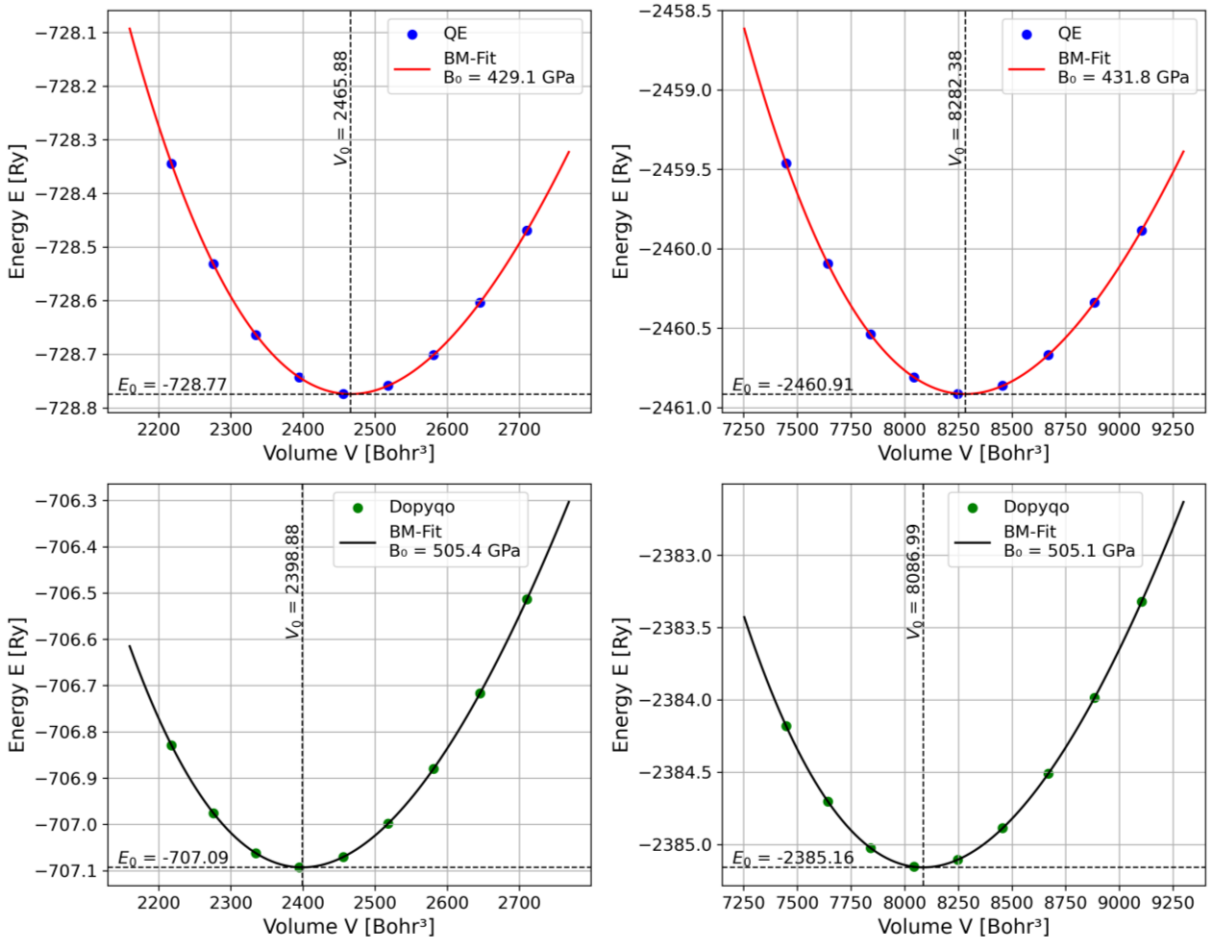
Appendix B: Fitting Result File: NaCl

```
# ------------------------------------------------------------
# Energy-Volume-Data
# ------------------------------------------------------------
#  Volume [Bohr³]          Energy [Ry]
# ------------------------------------------------------------
      1028.626000        -462.56791389
      1078.658700        -462.58843994
      1130.288000        -462.60220297
      1183.539100        -462.61014355
      1238.437000        -462.61323547
      1295.006800        -462.61204835
      1353.273600        -462.60729332
      1413.262400        -462.59945139
      1474.998300        -462.58911729

# ------------------------------------------------------------------------
# Fitting Parameters of Birch-Murnaghan Equation of State
# ------------------------------------------------------------------------
# Label               E₀ [Ry]      V₀ [Bohr³]        B₀ [GPa]        B₀'
# ------------------------------------------------------------------------
  Values:       -462.61329458   1249.53330143     23.81306128     4.5757
  Std. dev.:       0.00001216      0.10264144      0.01755074     0.0198
```

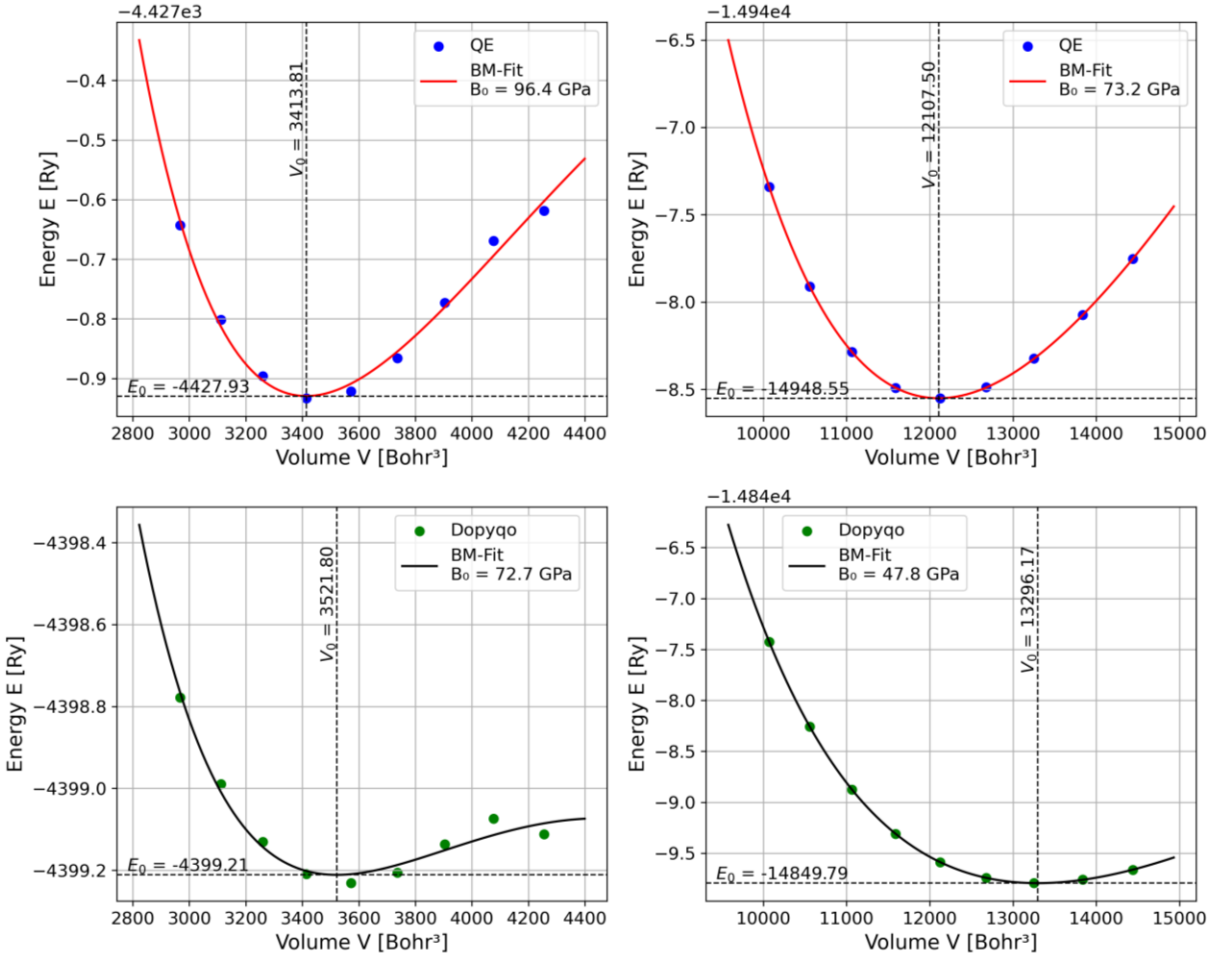
Appendix C: Calculation of Supercells

To include the crystal's periodicity while calculating only the Γ-point, 2x2x2 and 3x3x3 supercells were constructed based on the conventional unit cells for Al, Na and diamond. These structures were relaxed and tested for convergence as described for all other structures in Section 3.3. The resulting bulk modulus plots for diamond are shown in Appendix Figure 1.



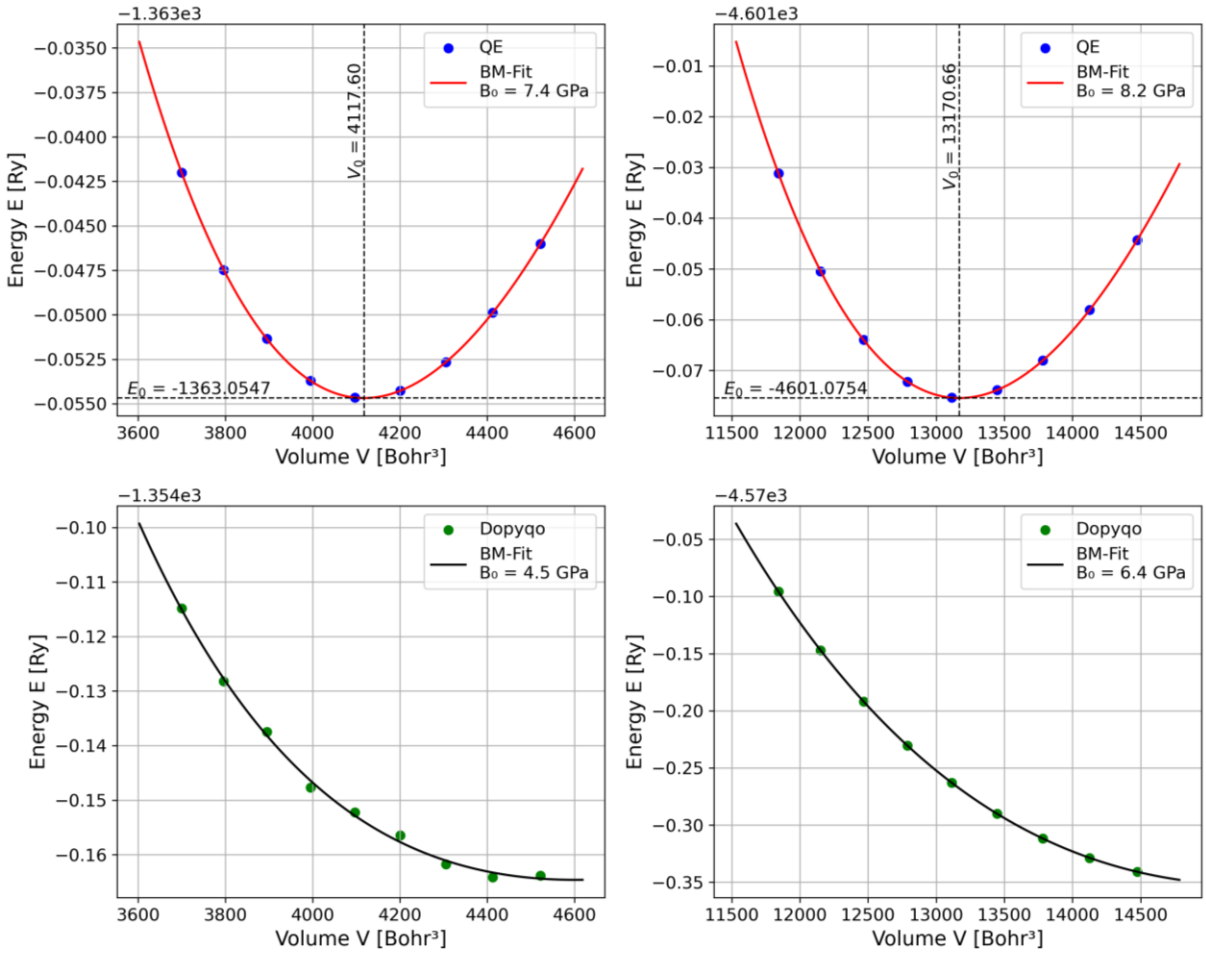
Appendix Figure 1: Supercell BM fits for diamond. Top-left: 2x2x2 supercell, QE-based. Top-right: 3x3x3 supercell, QE-based. Bottom-left: 2x2x2 supercell, Dopyqo-based. Bottom-right: 3x3x3 supercell, Dopyqo-based.

For both Dopyqo-based calculations, overbinding occurs in comparison to the DFT-based calculation. Even though the overbinding is not as strong as seen in Table 3.3.1, this indicates a k-mesh independence of the phenomenon. However, this parallelism with k-mesh-based calculations cannot be observed for Al and Na. The corresponding curves for Al are shown in Appendix Figure 2.



Appendix Figure 2: Supercell BM fits for Al. Top-left: 2x2x2 supercell, QE-based. Top-right: 3x3x3 supercell, QE-based. Bottom-left: 2x2x2 supercell, Dopyqo-based. Bottom-right: 3x3x3 supercell, Dopyqo-based.

Here, signs of underbinding are visible, i.e., an increased equilibrium volume and a decreased bulk modulus. Since the difference between the 2x2x2 and the 3x3x3 fits is large, these structures cannot be considered converged and therefore no conclusions should be made based on them. The same applies to the fits for Na shown in Appendix Figure 3.



Appendix Figure 3: Supercell BM fits for Na. Top-left: 2x2x2 supercell, QE-based. Top-right: 3x3x3 supercell, QE-based. Bottom-left: 2x2x2 supercell, Dopyqo-based. Bottom-right: 3x3x3 supercell, Dopyqo-based.

To achieve a converged calculation for these structures, a 4x4x4 supercell or even larger cells would need to be calculated. Since even a 4x4x4 supercell calculation can take several weeks for these structures, it is not possible to achieve convergence, and therefore these structures do not provide any clear insights. Since the calculation for diamond suggests that this structure is already close to convergence, Appendix Figure 1 can be taken as an indication that overbinding also occurs independently of a k-mesh.

Declaration of Authorship

I hereby declare,

Name, First Name: Hagemann, Nico
Student Number: 10002385

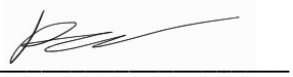
that I have written the present work

Title of Work: Calculation of Bulk Moduli using Many-Body Hamiltonians and Quantum Algorithms
Field of study: Chemistry

independently, have not submitted it elsewhere as an examination paper, and have used no sources or resources other than those indicated. I have identified all passages or thoughts adopted directly or indirectly as such. This includes the use of electronic media as well as text- or other content-generating IT tools such as ChatGPT.

In addition, I assure that when using text- or other content-generating IT tools such as ChatGPT, I have fully listed these tools in the section "Additional Resources" with their product name, my source of reference (e.g., URL), and details regarding the software functions used as well as the scope of use.

06.04.2026, Hannover
Date, Place


Signature