



Master's thesis

Electronic structure calculations on quantum computers

Hugo Åström

2022

Supervisor: Dage Sundholm

Examiners: Prof. Dage Sundholm & Dr. Susi Lehtola

University of Helsinki
Faculty of Science



HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

MATEMAATTIS-LUONNONTIETEELLINEN TIEDEKUNTA
MATEMATISK-NATURVETENSKAPLIGA FAKULTETEN
FACULTY OF SCIENCE

Tiedekunta – Fakultet – Faculty Faculty of science		Koulutusohjelma – Utbildningsprogram – Degree programme Theoretical and computational methods	
Opintosuunta – Studierikning – Study track Chemistry			
Tekijä – Författare – Author Hugo Åström			
Työn nimi – Arbetets titel – Title Electronic structure calculations on quantum computers			
Työn laji – Arbetets art – Level Master's thesis	Aika – Datum – Month and year 11.2022	Sivumäärä – Sidoantal – Number of pages 34	
Tiivistelmä – Referat – Abstract I discuss recent work regarding electronic structure calculations on quantum computers. I introduce quantum computing and electronic structure theory, and then discuss different mappings from electrons and excitation operators, to qubits and unitary operators, mainly Jordan–Wigner and Bravyi–Kitaev. I discuss adiabatic quantum computing in connection to state preparation on quantum computers. I introduce the most important algorithms in the field, namely, quantum phase estimation (QPE) and variational quantum eigensolver (VQE). I also mention recent modifications and improvements to these algorithms. Then I take a detour to discuss noise and quantum operations, a model for understanding how quantum computations fail because of noise from the environment. Because of this noise, quantum simulators have risen as a tool for understanding quantum computers and I have used such simulators to do electronic structure calculations on small atoms. The algorithm I have used, QPE, yields the exact result within the employed basis. As a basis I use numerical orbitals, which are very robust due to their flexibility.			
Avainsanat – Nyckelord – Keywords electronic structure, quantum computing, quantum phase estimation, variational quantum eigensolver			
Säilytyspaikka – Förvaringställe – Where deposited e-thesis			
Muita tietoja – Övriga uppgifter – Additional information			

Contents

1	Introduction	2
1.1	Quantum computers	4
1.2	The electronic structure problem	7
2	Mapping of the problem	9
2.1	Electron to qubit mappings	10
2.2	Hamiltonian Simulation	15
3	State Preparation	17
4	Quantum Algorithms	18
4.1	Quantum Phase Estimation	19
4.2	Variational Quantum Eigensolver	22
5	Noisy Quantum Computation	24
5.1	Quantum Operations	25
5.2	Simulating Quantum Computers	27
6	Results	27
6.1	Numerical Orbitals	28
6.2	Quantum Simulations	28
7	Discussion	30

List of abbreviations

FCI	Full Configuration Interaction
VQE	Variational Quantum Eigensolver
QPE	Quantum Phase Estimation
IQPE	Iterative Quantum Phase Estimation
SE	Schrödinger equation
SCM	Symmetry Configuration mapping
UCC	Unitary Coupled Cluster
UCCSD	Unitary Coupled Cluster Singles Doubles
JW	Jordan–Wigner
BK	Bravyi–Kitaev
BKSF	Bravyi–Kitaev Superfast
STO	Slater Type Orbital
GTO	Gaussian Type Orbital
Hartree–Fock	HF
CC	Coupled Cluster

1 Introduction

The electronic structure problem is arguably the most fundamental problem in chemistry. Its main task is to find the wave function of a chemical system, such as an atom or a molecule. The wave function determines all the chemical properties of the system. The electronic structure problem is the underlying problem that has to be solved when it comes to all chemically interesting applications such as drug design, finding efficient catalysts or constructing molecules for capturing solar energy. Therefore, finding efficient methods for solving electronic structure problems can be considered of as some kind of "holy grail" in chemistry. Theoretical and computational chemists have been developing appropriate electronic structure methods for decades. The large underlying challenge lies in the fact that with increasing number of particles, the Hilbert space of the problem increases exponentially, which means that the Schrödinger equation of only the simplest model systems can be solved analytically. Thus, approximations have to be made when one wants to perform useful applications. The ultimate method is the full configuration interaction (FCI) method [1, 2], which provides the exact solution within the employed basis. However, the FCI method scales factorially with system size according to Weyl's dimension formula [1]. The best ab initio methods, such as second order Møller–Plesset perturbation level (MP2) or coupled cluster singles and doubles with perturbative treatment of the triples level (CCSD(T)) [1] are used in large scale applications. They scale as N^5 to N^7 , where N represents the system size. The accurate FCI method can only be

applied to small systems, and is used mainly for benchmarking. Even though CCSD(T) calculations can be performed on large molecules consisting of several hundreds of atoms [7], the obtained energies differ significantly from total energies that would be obtained at the FCI level due to the approximations made. Advances in quantum physics naturally lead to new approaches for solving electronic structure problems. The most recent and important one is quantum computing.

When thinking about computational sciences at a very fundamental level, one might ask what kind of physical systems can be used to perform a computation? For a very long time only classical systems were considered for this task, like the classical bit in an average laptop, which usually makes use of some electromagnetic phenomenon to be either in the state 0 or 1, and together, many bits can perform advanced computations. It wasn't until the 80s when pioneers like Richard Feynman thought of making the physical system quantum mechanical [5, 6]. The motivation was that to be able to simulate quantum mechanical systems (such as atoms and molecules) in an *efficient* manner compared to classical computations, we also need to make the computer quantum mechanical. This seems reasonable since the quantum mechanical world is so different from the classical world. It should be (and is) very difficult to simulate it using classical physics. However, if we make the computer itself quantum mechanical, the story could be different. Quantum chemistry is a field which deals with highly quantum mechanical systems and therefore it has risen as a natural field of application for quantum computers. It is however well known that making quantum computations tractable is an extremely difficult task due to *decoherence*. Decoherence is the fact that quantum mechanical systems are very sensitive and difficult to control. They tend to interact with the environment and instead of staying in the quantum mechanical world, they collapse to some classical state or become *entangled* with the environment, which causes the computations to fail. For this reason it has taken such a long time for quantum computing to get serious attention from the field of quantum chemistry. More on the concepts of decoherence and entanglement can be found e.g. in refs [3, 4].

Now that quantum computing is becoming more tractable, the need for quantum algorithms grows. I have studied a few existing quantum algorithms for solving the electronic structure problem, focusing on the helium atom. First I have computed the one- and two-electron interactions on a classical computer, using a numerical basis set. There are several advantages to using numerical basis sets instead of analytical functions [45]. They are more robust and flexible, which means that they can be used as exact orbitals. This is an advantage over analytical basis functions such as Gaussian type orbitals (GTO) or Slater

type orbitals (STO), which are approximations of exact orbitals. I have then implemented the integrals on a quantum computer simulator and run the quantum phase estimation algorithm for different basis set sizes. I conclude the introduction with some basics of the theory behind quantum computers and a short recap of classical calculations. After this I focus on how to map electronic structure problems to a quantum device, i.e., rewriting it in terms of qubits and quantum gates. I consider a few different mappings and transformations. Their task is trying to minimize the number of qubits and gates needed. After that I present two of the most important quantum algorithms for finding the electronic structure, namely quantum phase estimation and variational quantum eigensolver. This is followed by a section on the largest challenge facing quantum computing today, which is noise. Noise is when the quantum computer interacts with the environment, causing the computation to fail. I discuss a theory for modeling noisy quantum channels which produce errors. After this I present the results of my work, and end with a discussion and conclusions.

1.1 Quantum computers

There are many differences between how classical computers and quantum computers work in practice. An example is the fact that in quantum computing one *measures* the result, which makes the quantum computing itself an experimental science. There are also very important differences on the fundamental theoretical level. Two of them are the superposition principle and quantum entanglement. Classical computers use bits as their most basic unit, which can take on values 0 and 1. The corresponding units on quantum computers are called quantum bits, or *qubits*. A qubit is a two-level quantum mechanical system and they work quite differently from their classical counterpart. A physical qubit can be realized by any quantum mechanical system with two different states, like an atom with two energy levels or a superconducting circuit where the current can go either clockwise or counterclockwise. For the purposes of this thesis however, the qubits will only be abstract mathematical objects, which lets one explore the theory of quantum information. As all other quantum mechanical systems, they are represented as vectors in a Hilbert space, which for a qubit is two-dimensional. Therefore a qubit in the general case is written as a superposition of the two states $|0\rangle$ and $|1\rangle$ as

$$|q\rangle = a|0\rangle + b|1\rangle, \tag{1.1}$$

where $a, b \in \mathbb{C}$ are coefficients which satisfy

$$|a|^2 + |b|^2 = 1. \quad (1.2)$$

Consequently, a qubit can be in the state 0 and 1 simultaneously contrary to the bit, which is always either 0 or 1. A measurement in the basis $\{|0\rangle, |1\rangle\}$ will collapse the qubit to the state $|0\rangle$ with probability $|a|^2$ and $|1\rangle$ with probability $|b|^2$. This is the superposition principle. The other important property is quantum entanglement, a phenomenon where two or more qubits are dependent on each other, or entangled. One can not know anything about one specific qubit, without affecting the other qubit(s). It is perhaps best demonstrated by an example. The quantum state

$$|\psi\rangle = \frac{|01\rangle + |10\rangle}{\sqrt{2}} \quad (1.3)$$

is a highly entangled state. It is impossible to isolate one qubit from the state and write it as a product state. If we measure the value of one qubit, we immediately know the value of the second one. These two properties of quantum mechanical states result in new ways to think of computation both concerning storage of information, and manipulating the information through algorithms. Regarding information storage, one can again think of a string of qubits in a superposition state

$$|\psi\rangle = a_0|0\rangle + a_1|01\rangle + a_3|10\rangle + a_4|11\rangle. \quad (1.4)$$

This qubit register consisting of two qubits stores four numbers, a_0, a_1, \dots , and in general, n qubits can store 2^n numbers while n classical bits can only store n numbers. This exponential increase in information storage is one of the properties which makes quantum computers desirable.

The next question is then: How do we perform computations with these qubits? The qubit being a vector on a Hilbert space, we can of course operate on them with operators on the space. The operators are represented as unitary matrices and are called *quantum gates*, or just gates. Some of the most common gates are found in table 1.1. These are the quantum equivalents of classical logic gates.

From these gates we can put a qubit in a superposition state using the Hadamard gate, put two qubits in an entangled state using the CNOT gate and make up gates which perform

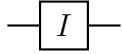
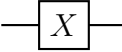
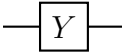
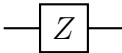
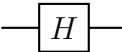
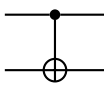
Name	Symbol	Matrix representation
Identity		$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$
Pauli-X		$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$
Pauli-Y		$\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$
Pauli-Z		$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$
Hadamard		$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$
CNOT		$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$

Table 1.1: The most common quantum gates and their symbol and matrix representations.

arbitrary rotations of angle θ of the qubit around an axis \hat{n}

$$R_{\hat{n}}(\theta) = \exp\left(\frac{-i\theta\hat{n}\cdot\vec{\sigma}}{2}\right) = \cos\left(\frac{\theta}{2}\right)I - i\sin\left(\frac{\theta}{2}\right)(n_xX + n_yY + n_zZ), \quad (1.5)$$

where $\vec{\sigma} = (X, Y, Z)$ denotes the vector of Pauli matrices. In general, any unitary transformation U of quantum mechanics, which performs the evolution

$$|\psi\rangle_f = U|\psi\rangle_i, \quad (1.6)$$

where $|\psi\rangle_f$ is the final state and $|\psi\rangle_i$ is the initial state, can be implemented on a quantum computer using a set of *universal quantum gates*.

Finally, at the end of the computation one must perform a measurement in the basis to get the result. The measurement can be represented by a set of measurement operators $\{M_m\}$, and the probability of getting the result m is

$$p(m) = \langle\psi|M_m^\dagger M_m|\psi\rangle. \quad (1.7)$$

As an example, in the computational basis one can measure a qubit using the projection

operators $|0\rangle\langle 0|$ and $|1\rangle\langle 1|$, which satisfy the completeness relation

$$\sum_m M_m^\dagger M_m = I. \quad (1.8)$$

As mentioned, the implementation of the operators and qubits is an experimental science, and many different methods have been proposed like trapped ion [30]-[34], superconducting quantum computers [35, 36] and photon systems [37, 38]. Now we have everything we need to start building quantum algorithms; the set of qubits, which form a *qubit register*, and the quantum gates.

1.2 The electronic structure problem

Before completely turning the attention to quantum computations, I will revisit classical quantum chemistry computations. This is necessary for understanding what the electronic structure problem is, why it is so difficult to solve on a classical computer and how it can be implemented on a quantum computer.

The electronic structure problem is quite simple to state: solve the Schrödinger equation (SE) and you know everything there is to know about the system. However, actually solving the equation is a much harder problem. The time-independent, non-relativistic SE can be stated as follows

$$H|\Psi\rangle = E|\Psi\rangle, \quad (1.9)$$

where H is the electronic Hamiltonian, $|\Psi\rangle$ is the quantum state and E the energy corresponding to that state. In the formalism of *first quantization* the electronic Hamiltonian is

$$H = -\frac{1}{2} \sum_i^N \nabla_i^2 - \sum_j^n \sum_i^N \frac{Z_j}{r_{ij}} + \sum_{i>j}^N \frac{1}{r_{ij}}, \quad (1.10)$$

where the first term is the kinetic energy operator, the second term gives the electron-nuclear interaction and the third the electron-electron interaction. N is the number of electrons, n the number of nuclei and Z_j the charge of nucleus j .

The state $|\Psi\rangle$ is an abstract vector in a Hilbert space and can be expanded in some basis $\{|\psi_i\rangle\}_{i=1}^n$ in the following way

$$|\Psi\rangle = \sum_{i=1}^n c_i |\psi_i\rangle, \quad (1.11)$$

where $c_i \in \mathbb{C}$ are the expansion coefficients. The wavefunction we want to deal with is

then obtained as an inner product

$$\Psi(\mathbf{x}) = \langle \mathbf{x} | \Psi \rangle = \sum_{i=1}^n c_i \langle \mathbf{x} | \psi_i \rangle = \sum_{i=1}^n c_i \psi_i(\mathbf{x}), \quad (1.12)$$

where the functions $\psi_n(\mathbf{x})$ are the basis functions.

Based on this, all the different computational approaches have individual ways of finding the wavefunction that solves the SE and FCI the one I focus on. Here the state is expanded as a Slater determinant

$$|\Phi\rangle = \frac{1}{\sqrt{N!}} \begin{vmatrix} \phi_1(1) & \phi_2(1) & \dots & \phi_N(1) \\ \phi_1(2) & \phi_2(2) & \dots & \phi_N(2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(N) & \phi_2(N) & \dots & \phi_N(N) \end{vmatrix}, \quad (1.13)$$

which is a product of one-particle spin functions $\phi_i(j)$ of electron j (the basis functions). The Slater determinant preserves the correct anti-symmetric properties of the state. By including all possible determinants made up from the functions in the basis set, one makes sure that all possible excitations are considered in the superposition state. Therefore, the solution will be exact within the basis set. The FCI state is

$$|\text{FCI}\rangle = \sum_i C_i |\Phi_i\rangle, \quad (1.14)$$

where $|\Phi\rangle$ are the Slater determinants. The energy of the system is

$$E = \sum_i \sum_j C_i C_j \langle \Phi_i | H | \Phi_j \rangle, \quad (1.15)$$

where we have the normalization condition

$$\sum_i |C_i|^2 \langle \Phi_i | \Phi_i \rangle = 1, \quad (1.16)$$

with the orthogonality condition

$$\langle \Phi_i | \Phi_j \rangle = 0, \quad i \neq j. \quad (1.17)$$

This allows us to write down the Lagrangian for FCI as

$$\mathcal{L} = \sum_i \sum_j C_i C_j \langle \Phi_i | H | \Phi_j \rangle - E \left(\sum_i |C_i|^2 \langle \Phi_i | \Phi_i \rangle - 1 \right) \quad (1.18)$$

and differentiating with respect to C_i gives an eigenvalue matrix equation

$$\mathbf{HC} = EC, \quad (1.19)$$

where the Hamiltonian matrix elements are

$$H_{ij} = \langle \Phi_i | H | \Phi_j \rangle. \quad (1.20)$$

The number of Slater determinants is

$$N_{\text{det}} = \binom{M}{N/2 + S} \binom{M}{N/2 - S}, \quad (1.21)$$

where N is the number of electrons and M is the number of orbitals. This large number of determinants is what makes classical computations so inefficient. A detailed discussion on classical quantum chemistry methods can be found e.g. in refs [1, 2] or other similar textbooks. It has now been demonstrated that FCI indeed scales factorially, which makes it useful mainly for benchmarking. The more tractable methods mentioned (CCSD(T) and MP2) yield energies which differ significantly from the FCI energy. Therefore, we now turn our attention to quantum computations instead.

2 Mapping of the problem

It is necessary to map the problem to the quantum device before the actual calculation. Thus, one has to rewrite the problem in such a way that it can be implemented on a quantum computer. The idea is to map the states of the quantum system onto the qubit register. The Hamiltonian is then mapped to a combination of unitary operators. In this section a few approaches to map the space of states to qubits is presented. The aim is to minimize the number of qubits needed, which is especially important for today's early quantum computers. Then a few different ways to implement the Hamiltonian will be discussed. The operator is mapped to a sum of Pauli operators which are easily implemented on a quantum device. The methods use as few operators as possible to generate shallow

algorithms, while preserving the antisymmetric properties of the quantum states.

2.1 Electron to qubit mappings

The electronic structure problem starts with defining the Hamiltonian. The non-relativistic electronic Hamiltonian for N particles in *second quantization* formalism can be written as

$$H = \sum_{pq} h_{pq} a_p^\dagger a_q + \frac{1}{2} \sum_{pqrs} v_{pqrs} a_p^\dagger a_q^\dagger a_s a_r, \quad (2.1)$$

where the weights, h_{pq} and v_{pqrs} , are one- and two electron integrals respectively. The one electron weights are obtained as

$$h_{pq} = - \int \chi_p^*(\mathbf{r}) \left(\frac{1}{2} \nabla^2 + \sum_i \frac{Z_i}{|\mathbf{R}_i - \mathbf{r}|} \right) \chi_q(\mathbf{r}) d\mathbf{r}, \quad (2.2)$$

which is the expectation value of the kinetic energy term and the Coulomb attraction term between the electron and the nuclei, as in equation 1.2. The two-electron interaction terms are given by

$$v_{pqrs} = \int \frac{\chi_p^*(\mathbf{r}_1) \chi_q^*(\mathbf{r}_2) \chi_r(\mathbf{r}_2) \chi_s(\mathbf{r}_1)}{|\mathbf{r}_1 - \mathbf{r}_2|} d\mathbf{r}_1 d\mathbf{r}_2. \quad (2.3)$$

The integrals are calculated on a classical computer and implemented on the quantum device. Since the integrals have to be calculated only once, they do not lead to any overhead to the simulation. In equation 2.1 a_i and a_i^\dagger are fermionic annihilation and creation operators, respectively, i.e., they annihilate/create an electron in the spin orbital i . They obey the anticommutation relations

$$a_i a_j^\dagger - a_j^\dagger a_i = \{a_i, a_j^\dagger\} = \delta_{ij} \quad (2.4)$$

$$\{a_i, a_j\} = \{a_i^\dagger, a_j^\dagger\} = 0, \quad (2.5)$$

which is important because it means that the Fermi statistics of fermions is automatically encoded on the operator level. The second quantization formalism is more convenient than the first quantization of equation 1.2 when doing quantum simulations because it reduces the number of qubits needed [15].

The Hamiltonian acts on the space of states, the Fock space. It is defined as

$$\mathcal{F} = \bigoplus_{n=0}^K \mathcal{H}_{n,K}, \quad (2.6)$$

which denotes a direct sum of the Hilbert spaces $\mathcal{H}_{n,K}$ for K spin orbitals. n is the number of particles. An arbitrary basis state in the Fock space can be written as

$$|\psi\rangle = |n_1, n_2, \dots, n_N\rangle, \quad n_i \in \{0, 1\}, \quad (2.7)$$

where each n_i is the occupation number of the spin orbital i . It is 0 or 1 if the spin orbital is unoccupied or occupied respectively. Using the creation operators a generic state can be written as

$$|n_1, n_2, \dots\rangle = (a_1^\dagger)^{n_1} (a_2^\dagger)^{n_2} \dots |0\rangle. \quad (2.8)$$

Considering the anticommutation relations in equations 2.4 and 2.5, the action of a_i and a_i^\dagger on that state is

$$a_i^\dagger |\dots, n_i, \dots\rangle = \delta_{n_i,0} (-1)^{\sum_{j<i} n_j} |\dots, 1, \dots\rangle \quad (2.9)$$

and

$$a_i |\dots, n_i, \dots\rangle = \delta_{n_i,1} (-1)^{\sum_{j<i} n_j} |\dots, 0, \dots\rangle. \quad (2.10)$$

The whole Fock space is redundant. The mapping has to be restricted to the subspace of states containing constant number N particles of the system. These are then the states that need to be mapped to the qubit register.

The most straightforward mapping is the direct mapping (DM). In this mapping each spin orbital is mapped to one qubit

$$|n_1 n_2 \dots n_N\rangle \mapsto |q_1 q_2 \dots q_N\rangle, \quad n_i, q_i \in \{0, 1\}, \quad (2.11)$$

where N is the number of spin orbitals. The total composite system is the tensor product of the individual systems and the notation

$$|q_1\rangle \otimes \dots \otimes |q_N\rangle = |q_1 \dots q_N\rangle \quad (2.12)$$

is adopted. n_i (q_i) has the value 0 if the spin orbital is unoccupied, and value 1 if the spin orbital is occupied.

In this basis a few different transformations can be used to map a_i and a_i^\dagger to unitary operators, which can be implemented on a quantum computer. Jordan–Wigner transformation (JW) [14] is one of the most commonly used of such transformations. a_i and a_i^\dagger are

mapped to strings of Pauli operators in the following way

$$a_j^\dagger = \bigotimes_{n=1}^{j-1} Z_n \otimes \sigma_j^+ \quad (2.13)$$

and

$$a_j = \bigotimes_{n=1}^{j-1} Z_n \otimes \sigma_j^-, \quad (2.14)$$

where

$$\sigma^\pm = \frac{X \pm iY}{2} \quad (2.15)$$

are the raising and lowering operators. These strings are tensor products of Pauli operators with different indices. An operator with a specific index acts on the qubit with the same index. For simplicity, the Pauli operators σ_x , σ_y and σ_z are written as X_i , Y_i and Z_i , when acting on qubit i . The anti-symmetric property of the fermionic state in equation 2.9 needs to be inherited. It is ensured by the string of Z operators because Z changes the phase of a qubit in the state $|1\rangle$ from $+1$ to -1 , but leaves $|0\rangle$ unchanged. The raising (lowering) operator raises (lowers) the value of the qubit it acts on, as the name suggests.

There exist some problems with the JW transformation. As we can see in equations 2.13 and 2.14, we might have to operate on qubits which are not adjacent. This means that this transformation is *non-local* and it can be quite impractical to implement on a physical quantum computer.

Another transformation is the parity mapping. For this we need to make a change of basis from the occupation number basis to the parity basis. In the occupation number basis, the value n_i of qubit i was the occupation number of spin orbital i . Now we make a change of basis such that the value f_i of qubit i contains the parity of all qubits up to qubit i

$$f_i = \sum_{k=0}^i n_k \pmod{2}. \quad (2.16)$$

We can define an operator

$$P = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 0 & 1 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}, \quad (2.17)$$

such that if $|\psi\rangle$ is a occupation number basis state, then the corresponding parity basis

state $|\phi\rangle$ is

$$|\phi\rangle = P|\psi\rangle, \quad (2.18)$$

where addition is again $\pmod{2}$. Like in the DM, we can define the corresponding creation and annihilation operators in the parity basis for N electrons

$$a_j^\dagger = \bigotimes_{k=j+1}^N X_k \otimes \sigma_j^+ \quad (2.19)$$

and

$$a_j = \bigotimes_{k=j+1}^N X_k \otimes \sigma_j^-, \quad (2.20)$$

where

$$\sigma_j^\pm = \frac{Z_{j-1} \otimes X_j \mp iY_j}{2} \quad (2.21)$$

is the raising or lowering operator in the parity basis. However this has the same problems as the Jordan–Wigner transformation: we get a similar string of Pauli operators which may be non-local. Also, the scaling of qubit operations needed for both these transformations is linear $\mathcal{O}(N)$. This scaling can be improved.

The Bravyi–Kitaev (BK) mapping [16] is an improvement to these mappings. It is a somewhat complicated transformation that combines the two. We again need N qubits to represent N spin orbitals and now the indexing of the qubits starts from 0. The information qubit i holds depends on the index i and can be divided into three scenarios. First one checks if i is even or odd. If i is even, then the qubit stores the occupation number of the spin orbital, like the Jordan–Wigner transformation. If i is odd the qubit stores the parity of a subset of qubits and there are two scenarios for this subset. If $\log_2(i+1) \in \mathbb{Z}$, then the subset is the same as in the parity transformations, i.e., all qubits with index $j \leq i$. Otherwise the subset is all qubits with index $k < j \leq i$, where k is the last index such that $\log_2(k+1) \in \mathbb{Z}$. It is quite unintuitive and might be better illustrated with an example. One can consider the change of basis from the occupation number basis, to the

Bravyi–Kitaev basis for eight qubits by the matrix

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} o_0 \\ o_1 \\ o_2 \\ o_3 \\ o_4 \\ o_5 \\ o_6 \\ o_7 \end{pmatrix} = \begin{pmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \\ q_4 \\ q_5 \\ q_6 \\ q_7 \end{pmatrix} = \begin{pmatrix} o_0 \\ o_0 + o_1 \\ o_2 \\ o_0 + o_1 + o_2 + o_3 \\ o_4 \\ o_4 + o_5 \\ o_6 \\ o_1 + o_2 + o_3 + o_4 + o_5 + o_6 + o_7 \end{pmatrix}. \quad (2.22)$$

The scaling is $\mathcal{O}(\log_2 n)$ and it is an improvement from JW and parity mapping. It has been shown that the BK mapping gives a significant reduction in gate count in quantum chemistry simulations compared to the DM and PM [17]. The form of the creation and annihilation operators in the BK basis will not be further discussed as they also retain a complicated form, but they can be found in e.g. ref [17]. Bravyi–Kitaev Superfast (BKSF) is another improved transformation. It has been shown that the performance for calculations on H_2 was improved when using BKSF, even though the gate count was larger compared to BK [26].

The number of qubits needed can also be reduced by the use of Symmetry Configuration Mapping (SCM) [8]. In SCM one sets constraints on the states themselves to rule out some of them and make the space of states smaller. One can start by constructing a symmetry adapted basis $\{|\Psi_l\rangle^\Sigma\}$, where l enumerates the basis states belonging to the same symmetry configuration. In this basis the total Hamiltonian is written as a direct sum of Hamiltonians for different symmetry configurations

$$H = \bigoplus_{\Sigma} H^\Sigma, \quad (2.23)$$

which means that we can solve equation 1.9 separately for each symmetry configuration

$$H^\Sigma |\Psi^\Sigma\rangle = E^\Sigma |\Psi^\Sigma\rangle. \quad (2.24)$$

The symmetry configuration

$$\Sigma = (N, S, S_z, \Gamma) \quad (2.25)$$

contains a set of parameters which restrict the state and the Hamiltonian. These are N , the number of electrons, S , the total spin, S_z , the total spin projection and Γ , the symmetry

representation of the state. The number of qubits needed is now dependent on the rank Λ^Σ of the symmetry configured Hamiltonian H^Σ as

$$Q^\Sigma = \lceil \log_2 \Lambda^\Sigma \rceil. \quad (2.26)$$

This can be an effective method to use, especially in single point calculations where the symmetry of the state is known from experiments. In ref [8] they performed calculations on the ground state of F_2 . They were able to reduce the number of qubits from 16 to 4 by using SCM. First they applied the frozen core approximation, leaving $N = 14$ electrons when assuming that the four electrons in the four $1s$ -orbitals remain constant. They further set $S_z = 0$ and $\Gamma = A_g$. They performed the calculation in the minimal atomic basis with 16 spin orbitals.

2.2 Hamiltonian Simulation

Now we can choose a basis and a transformation to map our operators to Pauli strings. The next step is to implement these operators onto the quantum device. To do that, we first divide the Hamiltonian into a sum of local Hamiltonians

$$H = \sum_k H_k \quad (2.27)$$

such that each H_k acts at most on a constant number of qubits. Next we exponentiate this operator to construct a unitary operator

$$U = e^{iH} \quad (2.28)$$

and in general, we would like to write

$$e^{\sum_k H_k} = \prod_k e^{H_k}. \quad (2.29)$$

However, since the different H_k do not usually commute, i.e.,

$$H_k H_l - H_l H_k = [H_k, H_l] \neq 0 \quad (2.30)$$

in the general case, equation 2.29 does not hold. Instead we need the Trotter–Suzuki formula

$$\lim_{m \rightarrow \infty} \left(e^{\sum_k H_k/m} \right)^m = \prod_k e^{H_k/m}. \quad (2.31)$$

The local Hamiltonians H_k are just Pauli operators. Applying equation 1.5, this operator can be implemented on a quantum computer in form of rotations. However, the problem lies in the fact that we can not realistically take m to infinity and need to make an approximation and cut it off at some point. This will introduce errors.

There are other methods for generating quantum circuits from a Hamiltonian and one of them is truncated Taylor series [11]. Again we start with the operator in equation 2.28 where the Hamiltonian is a sum of Pauli operators. Here it is important to note that each H_k must be unitary. Now, instead of using Trotterization, we make use of Taylor series

$$U \approx \sum_{k=0}^K \frac{1}{k!} (-iH)^k, \quad (2.32)$$

but cut it off at order K . If we want the simulation to accuracy ϵ , K will be of order

$$K = \mathcal{O}\left(\frac{\log(1/\epsilon)}{\log \log(1/\epsilon)}\right). \quad (2.33)$$

However, the powers of the Hamiltonian might not be unitary but combining this with equation 2.27 we can expand the series

$$U \approx \sum_{k=0}^K \sum_{l_1, \dots, l_k} a_{l_1} \dots a_{l_k} H_{l_1} \dots H_{l_k}, \quad (2.34)$$

where we can set each $a_l > 0$. This operator has a form which can be implemented on a quantum computer. I will not however go further into the implementation of the operator and instead focus on the more used Trotterization method.

These methods for implementing the Hamiltonian introduce errors. One source for errors is that the operator in the truncated Taylor series method is not exactly unitary, since the series expansion is cut off. However, it is close to unitary and the error can be controlled. Implementation of Hamiltonians remains an active area of research and other approaches such as qubitization [19] have been suggested. Thus, we now have the tools to prepare Hamiltonians on a quantum computer.

3 State Preparation

The last step to consider before discussing the quantum algorithms is preparation of the initial state, which is one of the strengths of quantum computers. Preparing a good initial state with sufficient overlap with the true ground state can be a crucial starting point to many algorithms. For regular closed shell singlet molecules the Hartree–Fock (HF) state is usually a good guess. Around the equilibrium geometry the overlap tends to be about $|\langle \Psi_{\text{HF}} | \Psi_{\text{exact}} \rangle|^2 \approx 0.9$, but this overlap can decrease for molecules beyond these criteria [10].

My focus will lie on adiabatic state preparation, since it is probably the best known state preparation technique in quantum computing. The idea stems from the adiabatic theorem, which makes the following statement:

A physical system remains in the eigenstate of a Hamiltonian when acted upon with a perturbation if the evolution is sufficiently slow and there is a gap between the eigenvalue of the system and the rest of the spectrum of the Hamiltonian.

One can now make the Hamiltonian time dependent

$$H(\xi), \quad \xi \in [0, 1], \quad (3.1)$$

where the ground state of $H(0)$ is some known state that can be prepared on a quantum computer, and the ground state of $H(1)$ is the desired state. $H(1)$ could e.g. be obtained by slowly turning on some perturbation say

$$H(1) = H_0 + H', \quad (3.2)$$

where $H_0 = H(0)$ and H' is a perturbation. Now one can make use of the time-dependent SE

$$i \frac{d}{dt} |\Psi(t)\rangle = H(t/t_f) |\Psi(t)\rangle, \quad t \in [0, t_f] \quad (3.3)$$

to evolve the system along a determined path. Given that the spectrum of the Hamiltonian along this path is gapped and the evolution is sufficiently slow, the final state will be the ground state of the $H(1)$. The total time, t_f , depends on the gap between the two lowest eigenvalues of the Hamiltonian along the path, and a parameter ε as [18]

$$t_f > \frac{\varepsilon}{g_{\min}^2}, \quad (3.4)$$

where

$$g_{\min} = \min(E_1(t) - E_0(t)) \quad (3.5)$$

is the gap and

$$\varepsilon = \max \left| \langle \psi_1(t) | \frac{dH(t/t_f)}{dt} | \psi_0(t) \rangle \right|, \quad (3.6)$$

where $|\psi_0(t)\rangle$ is the ground state and $|\psi_1(t)\rangle$ is the first excited state. Thus, if t_f is constant or grows polynomially, there will be no excess overhead to the simulation. On the other hand, large t_f might require deep circuits, which is still a problem as mentioned. Another challenge could lie in the fact that the spectrum along the whole path might not be known and degeneracies could lead to the wrong final state. An algorithm designed for state preparation for molecules with open shell character has also been suggested [20]. It is based on Serber construction [47, 48] and has the advantage that the circuit depth is only 2.

An alternative technique is variational state preparation. This will be discussed in more detail in section 4.2 in connection to VQE. This now leads us to the subject of quantum algorithms.

4 Quantum Algorithms

The algorithm of most interest is Quantum Phase Estimation (QPE) and state preparation and implementation of the Hamiltonian are the starting points for that. Now that we have these tools, we can start to explore the algorithm.

As mentioned, the exact solution to the SE within the employed basis is classically given by FCI. The corresponding algorithm for quantum computers is QPE [9]. It is an algorithm for determining eigenvalues of unitary operators. Like FCI, it gives the exact solution to the SE within the basis. The advantage is that it scales polynomially with system size, compared to factorially for FCI. As mentioned, the starting points are the initial state and the implementation of the Hamiltonian. The task of the algorithm is to project the initial state onto an eigenstate of the Hamiltonian. A measurement then yields the corresponding eigenvalue of the eigenstate exactly. The success rate is proportional to the overlap $|\langle \psi_{\text{initial}} | \psi_{\text{exact}} \rangle|^2$ between the initial state and the exact state. Thus, state preparation can be of great value. Given exponentially decreasing overlap between the HF state and multiconfigurational states, an exponential numbers of runs of the algorithm is also needed. For singlet closed-shell molecules around their optimal geometry the HF state is usually sufficient.

The problem with QPE is that it is sensitive to noise. Therefore, VQE is another important algorithm, especially for near time quantum computers because it is more robust. It takes advantage of quantum computers' ability to prepare and measure a quantum state, while utilizing classical computers to keep the quantum circuits from becoming too deep. It is based on Ritz variational principle [21] for quantum mechanics, which is a common method in many classical algorithms as well. The two algorithms will be discussed in detail in this section.

4.1 Quantum Phase Estimation

QPE is an algorithm for finding eigenvalues of unitary operators on quantum computers. The eigenvalue of a unitary operator U can be represented by a point on the unit circle in the following way

$$U|u\rangle = e^{i\varphi}|u\rangle. \quad (4.1)$$

QPE estimates the value of φ with very high accuracy. The Hamiltonian 2.1 is an Hermitian operator. Thus, we again construct a unitary operator by exponentiating H as in equation 2.28

$$U|u\rangle = e^{-iH}|u\rangle = e^{i\varphi}|u\rangle, \quad (4.2)$$

where φ , the "phase", is an eigenvalue of H .

The algorithm consists of two qubit registers. One with t qubits, the ancilla register, from which we in the end will read out the eigenvalue, and another with m qubits, representing the initial state $|u\rangle$. The computer starts in the product state

$$|\psi\rangle = |0\rangle^{\otimes t} \otimes |u\rangle = |0\rangle^{\otimes t}|u\rangle. \quad (4.3)$$

First, we apply to the first register a sequence of Hadamard gates, $H^{\otimes t}$, to put the computer in the equal superposition state

$$|\psi\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} |j\rangle|u\rangle. \quad (4.4)$$

Next, we apply controlled U^j -operations with the j :th qubit in the first register as the control qubit and $|u\rangle$ as the target

$$|\psi\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} |j\rangle U^j|u\rangle. \quad (4.5)$$

We can write $|u\rangle$ as a superposition of states in the computational basis

$$|\psi\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} |j\rangle U^j \sum_k c_k |\phi_k\rangle \quad (4.6)$$

$$= \frac{1}{\sqrt{N}} \sum_k c_k \sum_{j=0}^{N-1} |j\rangle U^j |\phi_k\rangle \quad (4.7)$$

$$= \frac{1}{\sqrt{N}} \sum_k c_k \sum_{j=0}^{N-1} |j\rangle (\lambda_k)^j |\phi_k\rangle, \quad (4.8)$$

where λ_k is the k :th eigenvalue of U , with eigenvector $|\phi_k\rangle$. If we now write the eigenvalues as $e^{i\varphi_k}$ according to 4.2, we obtain

$$|\psi\rangle = \frac{1}{\sqrt{N}} \sum_k c_k |\phi_k\rangle \sum_{j=0}^{N-1} e^{i\varphi_k j} |j\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{i\varphi_k j} |j\rangle |u\rangle. \quad (4.9)$$

We can now make an interesting observation: the first qubit register has the form of quantum Fourier transformation (QFT) of the state $|\varphi_k\rangle$. We apply the inverse QFT

$$\text{QFT}^\dagger \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{i\varphi_k j} |j\rangle = |\varphi_k\rangle \quad (4.10)$$

to obtain a state

$$|\psi\rangle = |\varphi_k\rangle |u\rangle \quad (4.11)$$

where the first qubit register contains the t -bit representation of λ_k . We then measure it in the computational basis to obtain the eigenvalue λ_k with probability $|\langle u|\phi_k\rangle|^2 = |c_k|^2$. The circuit diagram showing the first part of the algorithm, up until the inverse QFT, can be found in figure 4.1. A more detailed description of the QFT subroutine can be found in ref [3].

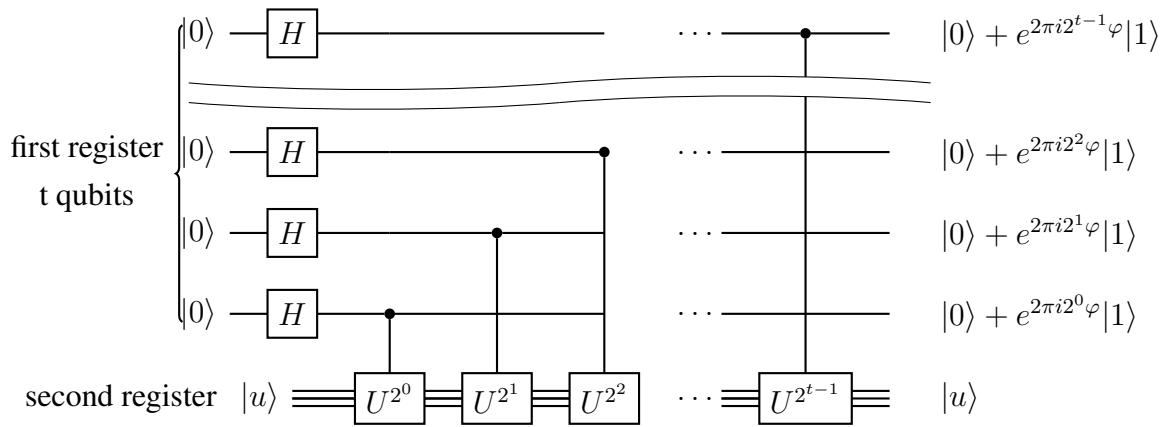


Figure 4.1: Circuit diagram outlining the first part of the QPE algorithm.

The accuracy of the algorithm depends on the number of qubits in the first qubits register. One can modify the algorithm such that the accuracy is independent of the number of qubits used. This procedure is called iterative QPE (IQPE) and it reduces the number of qubits needed [10]. As the name suggests, the algorithm proceeds in an iterative manner. In each iteration one digit of the eigenvalue is measured, starting from the last and working towards the first ones. One problem might be that one has to prepare the trial state in each iteration if one does not want to maintain throughout the whole algorithm, but if the state is easily prepared then it is no problem. Otherwise IQPE works in the same way as QPE and the circuit for iteration k can be found in figure 4.2. The rotation $R_z(\omega_k)$ and the second Hadamard gate form the inverse QFT for the first qubit.

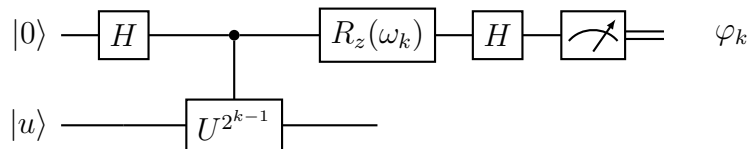


Figure 4.2: The circuit for iteration k of IQPE.

This version however does not remove the bottleneck of the algorithm, namely the implementation of the operator U .

4.2 Variational Quantum Eigensolver

QPE is not useful for doing computations on near-term quantum computers. VQE however, is more robust against noise. It is a hybrid quantum-classical algorithm. It utilizes quantum computers' ability to prepare and measure quantum states but takes advantage of classical computers to avoid deep circuits. The algorithm starts by finding a good trial state, or *ansatz*, $|\Psi(\vec{\theta})\rangle$ and a trial quantum circuit $U(\vec{\theta})$. They are parameterized by a set $\vec{\theta} = (\theta_1, \dots, \theta_n)$ of parameters we want to optimize. Contrary to QPE, this method is iterative and in each iteration we prepare a new state and new circuit. At the end of the iteration the energy is measured. Then the parameters are optimized with respect to the energy on a classical computer. That way a new set of parameters are obtained for the next iteration.

The outline of the algorithm is as follows:

1. Prepare the circuit $U(\vec{\theta})$.
2. Run the circuit to get $|\Psi(\vec{\theta})\rangle$.
3. Measure expectation value of H .
4. Optimize the parameters $\vec{\theta}$.
5. If it has not converged go back to step 1.

There are different ways of preparing the ansatz and the circuit. The original approach was unitary coupled cluster (UCC) [22]. It is based on the classical coupled cluster (CC) method [42, 43], but modified to be unitary. UCC offers several advantages over the non-unitary version [44] but there is no way to implement it efficiently on a classical computer. Now the new state is obtained as

$$|\Psi(\vec{\theta})\rangle = U(\vec{\theta})|\Phi\rangle, \quad (4.12)$$

where the ansatz $|\Phi\rangle$ is e.g. an HF determinant.

To construct the circuit we will use the unitary cluster operator, which is given by

$$|\Psi_{\text{UCC}}\rangle = \exp[T - T^\dagger]|\Phi\rangle, \quad (4.13)$$

where T is the cluster operator when we have n electrons

$$T = \sum_{k=1}^n T_k. \quad (4.14)$$

The cluster operator is a sum of excitation operators

$$T_1 = \sum_{\substack{i \in \text{occ} \\ k \in \text{virt}}} t_i^a a_a^\dagger a_i \quad (4.15)$$

$$T_2 = \frac{1}{4} \sum_{\substack{i, j \in \text{occ} \\ a, b \in \text{virt}}} t_{ij}^{ab} a_a^\dagger a_b^\dagger a_i a_j \quad (4.16)$$

⋮

$$T_n = \frac{1}{(n!)^2} \sum_{\substack{i_1, \dots, i_n \in \text{occ} \\ a_1, \dots, a_n \in \text{virt}}} t_{i_1 \dots i_n}^{a_1 \dots a_n} a_{a_1}^\dagger \dots a_{a_n}^\dagger a_{i_1} \dots a_{i_n}, \quad (4.17)$$

i.e., T_m excites m electrons from occupied orbitals to virtual orbitals. Here the indices i, j, k, l, \dots indicate occupied orbitals and a, b, c, d, \dots indicate virtual orbitals. We can write the circuit as

$$U(\vec{\theta}) = \exp \left[\sum_k \theta_k (T_k - T_k^\dagger) \right] \quad (4.18)$$

and make a Suzuki–Trotter decomposition

$$U(\vec{\theta}) \approx \left(\prod_k e^{\frac{\theta_k}{n} (T_k - T_k^\dagger)} \right)^n. \quad (4.19)$$

Usually it suffices to go as high as single and double excitations (UCCSD), i.e.,

$$T \approx T_1 + T_2 \quad (4.20)$$

to obtain good results.

There have been several modifications made to the originally suggested UCCSD approach [23, 24], which are also based on UCC. A modification suited for strongly correlated systems has been suggested, called ADAPT-VQE [25]. It starts by defining a set

$$\{t_q^p, t_{rs}^{pq}, \dots\} \quad (4.21)$$

of excitation operators and the HF state. In each iteration the gradient of the energy with respect to each operator in the set is calculated

$$\frac{\partial E^{(n)}}{\partial \theta_i} = \langle \psi^{(n)} | [H, O_i] | \psi^{(n)} \rangle, \quad (4.22)$$

where n denotes the iteration and i the operator index. The operator with the largest gradient is then applied to the trial state, the parameters are optimized and a new energy is obtained. If the gradient is converged the algorithm is done. The final state at convergence is

$$|\psi^{\text{adapt}}\rangle = (e^{\tau_N})(e^{\tau_{N-1}}) \dots (e^{\tau_1})|\psi^{\text{HF}}\rangle, \quad (4.23)$$

where each e^{τ_n} is computed at iteration n . The idea behind this approach is that the algorithm builds the trial state itself step by step, so that the system itself decides the state. They showed that the performance of the method is very good for both uncorrelated and correlated systems, outperforming the original approach.

A hardware-efficient method has also been proposed [13]. It uses alternating single qubit rotations, $U^{q,d}(\vec{\theta})$ and entangling operations, U_{ent} . Here q denotes the qubit index and d the circuit depth. The initial state is the state $|00 \dots 0\rangle$, which yields the trial state

$$|\psi(\vec{\theta})\rangle = \prod_{q=1}^N (U^{q,d}(\vec{\theta})) U_{\text{ent}} \prod_{q=1}^N (U^{q,d-1}(\vec{\theta})) \dots U_{\text{ent}} \prod_{q=1}^N (U^{q,0}(\vec{\theta})) |00 \dots 0\rangle, \quad (4.24)$$

where the single qubit rotations have the form

$$U^{q,i}(\vec{\theta}) = R_z(\theta_1^{q,i}) R_x(\theta_2^{q,i}) R_z(\theta_3^{q,i}), \quad (4.25)$$

where the R_n are the rotation operators from equation 1.5. This method is useful because it yields shallow circuits. Kandala et al. got good results for their studied molecules, however, there are no results of the method giving good results for highly correlated systems. Moreover, it requires $N(3d+2)$ parameters, which is higher than that of ADAPT-VQE, where the number of parameters is equal to the number of operators in the set.

An improvement to the ADAPT-VQE has also been suggested [27], referred to as a hardware-efficient ADAPT-VQE, or qubit-ADAPT-VQE. It replaces the excitations operator set with Pauli strings and introduces a completeness relation for the operators. They show that this results in the operator set scaling linearly with the number of qubits, resulting in a more hardware-efficient algorithm and lower-depth circuits.

5 Noisy Quantum Computation

Before discussing the applications of the quantum algorithms, I will take a small detour into the theory of noise. The model of noise is somewhat mathematical and might not be relevant for quantum chemistry computations, but offers insight as to what goes on when

the quantum gates don't work as planned. This will then lead to the subject of quantum computer simulators, which have been the focus of my experiments.

One of the biggest challenges with today's quantum computers is that they are not closed quantum systems, but open quantum systems. This means that they interact with their environment, and this interaction produces noise. This noise can be e.g. heat coming in to the system from the surroundings, or electromagnetic fields interacting with the qubit. It can cause different problems, like collapsing of the superposition of the qubit or a bit flip. These cause the computation to fail. We can study noise both from a theoretical point of view with a mathematical model, and from a practical point of view using quantum computer simulators. The theoretical viewpoint requires construction of a new formalism, which is discussed in this section. This is followed by a discussion on quantum computer simulators.

5.1 Quantum Operations

To understand noise we can construct a model called the quantum operations formalism. It is now easier to work in the density matrix formulation. If a quantum system is in the state $|\psi_i\rangle$ with probability p_i , we have an ensemble of pure states, $\{p_i, |\psi_i\rangle\}$. Then we can define a density matrix, or density operator as

$$\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|. \quad (5.1)$$

Just as in the state vector representation, all the information about the quantum system is contained within the density matrix and all the postulates of quantum mechanics can be restated in the density operator formalism. We begin by stating that there is a noisy interaction U which is not necessarily a unitary interaction. This is visualized in figure 5.1 where ρ_{env} is the density matrix of the environment and ρ' is the final state of the system. We can define a map \mathcal{E}

$$\rho' = \mathcal{E}(\rho), \quad (5.2)$$

which we call a *quantum operation*. This model of noise focuses a lot on how to develop a formalism for these operations.

Next we can introduce the state space of the interacting environment $|e_k\rangle$. The environment starts in some state $|e_0\rangle$. Then to obtain the density matrix of the system we trace out the environment from the composite system $U\rho \otimes \rho_{\text{env}}U^\dagger$, using partial trace

$$\mathcal{E}(\rho) = \text{Tr}_{\text{env}}(U\rho \otimes \rho_{\text{env}}U^\dagger) \quad (5.3)$$

$$= \sum_k \langle e_k | U \rho \otimes |e_0\rangle \langle e_0| U^\dagger |e_k\rangle \quad (5.4)$$

$$= \sum_k E_k \rho E_k^\dagger. \quad (5.5)$$

The $E_k = \langle e_k | U |e_0\rangle$ are operators on the space of states of our system.

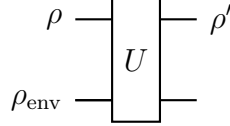


Figure 5.1: The density matrix of the system interacts through a black box with the environment to become a new state.

The quantum operation satisfies three properties:

1. The probability of the process represented by \mathcal{E} occurring is represented by the trace $\text{Tr}(\mathcal{E})$.
2. For a set of probabilities $\{p_i\}_{i=1}^n$ and density matrices $\{\rho_i\}_{i=1}^n$, \mathcal{E} satisfies

$$\mathcal{E}\left(\sum_{i=1}^n p_i \rho_i\right) = \sum_{i=1}^n p_i \mathcal{E}(\rho_i). \quad (5.6)$$

3. \mathcal{E} is completely positive.

If these three properties are satisfied, then the quantum operation can be represented as in equation 5.5 [3], where E_k satisfy

$$\sum_{k=1}^N E_k E_k^\dagger \leq I. \quad (5.7)$$

The quantum operations formalism is good for describing noisy quantum channels. As an example, one can consider the bit flip channel. It flips a qubit from $|0\rangle$ to $|1\rangle$ and vice versa with probability p , and leaves it be with probability $1 - p$. Then we have

$$E_0 = \sqrt{1-p}I = \sqrt{1-p} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad E_1 = \sqrt{p}X = \sqrt{p} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}. \quad (5.8)$$

From this one can see that the quantum operation \mathcal{E} flips a qubit with probability p . More on quantum operations and the effects of noise can be found in textbooks like ref [3]. The effect of noisy quantum channels can be studied using quantum computer simulators.

5.2 Simulating Quantum Computers

So far I have discussed quantum computers from a theoretical point of view. We want to study quantum computers from a practical one also. This is impractical, or in some cases not possible, to do using real quantum computers. As an example, the goal of my work is to study electronic structure calculations on quantum computers that correspond to classical FCI calculations. This would firstly be very resource consuming to do on actual quantum computers when algorithms are run many times. Secondly, it would not give many useful results since these algorithms are still very prone to errors. The solutions to these problems is to simulate a quantum computation. This can be done on classical computers and there already exist several different programs for simulating both error free and noisy quantum computations [28, 29].

6 Results

The goal of this work is to test quantum computer simulators by calculating the electronic structure and ground state energy of the helium atom using quantum algorithms corresponding to classical FCI calculations. First the one- and two electron integrals needed to be calculated and for this purpose a basis set had to be defined. I started only with the HF level without any unoccupied, or virtual orbitals. I optimized the electronic structure and included more orbitals in the next calculation, again optimizing the electronic structure, and so on, building larger basis sets of top of the previous ones. I used numerical orbitals which I will discuss shortly in the next section. After this I implemented the integrals on the quantum simulator to calculate the ground state energy of the atom. I did the calculations for a few different basis sets and parameter combinations, and compared the results.

The quantum simulations were done on Atos quantum learning machine Kvasi, which has a Python interface. One can implement noisy quantum channels using the quantum operations formalism from section 5.1 to simulate realistic quantum computers of today, however I have opted to simulate error free computations since the goal was to understand how the algorithms work in a perfect environment. In this section I first discuss the idea

behind the numerical basis set I have used, and then go through the implementation of the quantum algorithm in more detail.

6.1 Numerical Orbitals

As mentioned in section 2.2, the one- and two-electron weights are calculated on a classical computer before the quantum simulation. To calculate these, we first need a basis set and I have made use of numerical orbitals since they have several advantages over analytical functions. A detailed discussion on the method can be found in refs [39]-[41]. The orbital functions I have used can be divided into two parts, a radial part and an angular part

$$\phi_{nlm}(r, \vartheta, \varphi) = R_{nl}(r)Y_{lm}(\vartheta, \varphi), \quad (6.1)$$

where the angular part is an analytical function and the radial part is numerical. The radial function is defined on a grid in such a way, that inside each element $r \in [x_i, x_{i+1}]$ it is defined by a polynomial. At the gridpoints $r = x_i$ continuity between the polynomials at adjacent elements is assured.

One big advantage when using numerical orbitals is that they are more flexible than common analytical basis functions like Slater type orbitals (STO) or Gaussian type orbitals (GTO), which means that they can be adapted to represent exact orbitals [45]. Therefore, faster convergence to the complete basis set (CBS) limit is achieved. This is important for current quantum devices especially because it will mean that fewer qubits are required. This will lead to further advantages if one uses VQE because fewer qubits means more shallow circuits. As an example, looking at the calculations in ref [13] where they used operators of the form of equation 4.25, it is clear that fewer qubits will result in more shallow circuits.

6.2 Quantum Simulations

To map the electronic states to the qubit register I used DM and since I did a single point calculation, the mapping was done considering only symmetry configuration of the ground state. The JW transformation was used to map the Hamiltonian to Pauli operators and then Trotter–Suzuki decomposition was used to build the circuit.

The one- and two-electron parts of the Hamiltonian are separately implemented as tensors of rank two and four. By ensuring that these tensors have the right form, one can obtain the desired symmetry configuration. This is perhaps best demonstrated with an example. Take the helium atom and a basis set of the 1- 2- and 3s-orbitals, which gives six spin

orbitals. The ground state has the term symbol 1S_0 , i.e., zero spin projection and s -symmetry. The full Fock space can first and foremost be reduced to all states containing only two electrons

$$\{|\Psi\rangle\} = \{|110000\rangle, |101000\rangle, \dots, |000011\rangle\}. \quad (6.2)$$

Further reductions can be made if we make the constraint that $s_z = 0$ so the electrons must have opposite spin. Then the space of states becomes

$$\{|\Psi\rangle\} = \{|110000\rangle, |100100\rangle, |100001\rangle, |011000\rangle, \dots, |000011\rangle\} \quad (6.3)$$

and therefore we only want to consider excitations which preserve the total spin of the system in the one- and two-body tensors. Then the one-body tensors have the form h_{ij} , such that the element is zero if one of the indices is odd, corresponding to spin down and one even, corresponding to spin up. This ensures that we only excite spin up to spin up and spin down to spin down. The two-electron integrals have the form $\langle \mathbf{x}_1 \mathbf{x}_2 | \mathbf{x}_2 \mathbf{x}_1 \rangle$, which means that the two-body tensors will have the form h_{ijkl} , where the non-zero elements are the ones where i and l are odd and j and k are even or vice versa.

After the integrals were implemented it was time to choose a mapping of the operators to quantum gates and as mentioned, I opted for the JW transformation since it is the simplest and most the commonly used one.

Another parameter that can be varied is the number of qubits needed for the ancilla qubit register containing the eigenvalue, and it can be obtained by analysis of performance [3]. To obtain φ accurate to n bits we need t qubits according to the following equation

$$t = n + \left\lceil \log \left(2 + \frac{1}{2\epsilon} \right) \right\rceil. \quad (6.4)$$

Choosing a success rate of 0.9, i.e., $\epsilon = 0.1$ and $n = 6$, which should be sufficient accuracy, gives $t = 8$. The energy given when increasing the number of ancilla qubits can be found in figure 6.1. One can notice that it converges at 8 qubits.

There is excellent agreement with the HF energy of helium done with QPE and the classical "exact" result. Unfortunately, I was not able to reproduce this for larger basis sets. There is still some agreement when performing an FCI computation with two s -functions using QPE, -2.887968 *a.u.* and the classical calculation, -2.8779968 *a.u.*, but the algorithm fails for larger basis sets. I have reason to believe that the disagreement is due to limitations in the quantum simulator when simulating QPE, since diagonalizing the

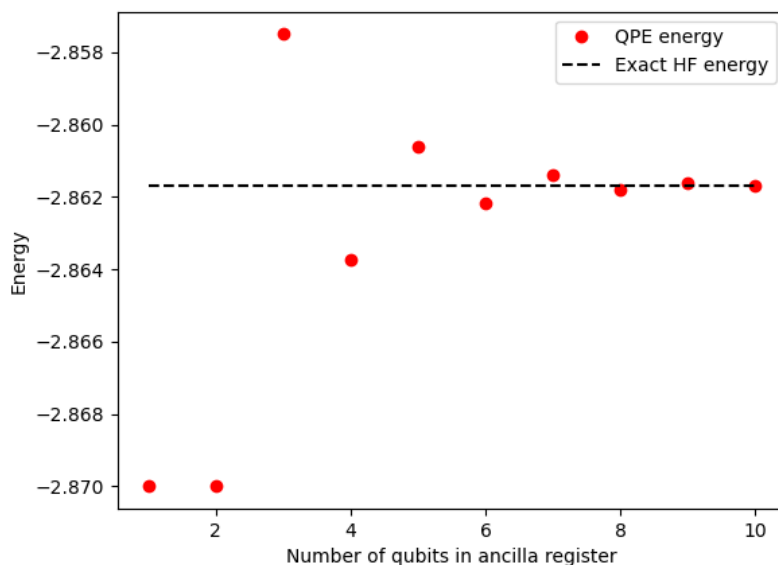


Figure 6.1: Energy of helium atom given by QPE as a function of the number of ancilla qubits. Calculations done on Kvasi with HF basis set.

Hamiltonian matrix gives eigenvalues which are in good agreement with the exact energy. The limitations could lie in the fact that the gate count rises quite high when going to larger bases, making it hard to simulate classically.

7 Discussion

I have explored progress in the field of electronic structure calculations in the age of quantum computing, discussing the important algorithms QPE and VQE and their modifications. I have computed the electronic structure of the He atom on classical quantum computer simulators. These simulators are an important tool in the age of noisy quantum computers. The computations are in good agreement for minimal basis sets, but as the dimensions of the Hamiltonian matrix grows, the algorithm fails. I have used numerical orbitals, which converge faster to the CBS limit than analytical basis sets, due to their ability to represent exact orbitals. This in turn could result in fewer qubits, especially when using VQE. Thus, one might in the future try to implement these numerical calculations to VQE and see if they have an impact.

There of course remain many great challenges to overcome before quantum computers can make new contributions to computational chemistry. The biggest challenge remains to

minimize the impact of noise in quantum computations. There is progress in this area with many different approaches being taken, e.g. trapped-ion quantum computers [30, 31, 32], but this remains an engineering problem which is a whole field of its own. Moore's law, which is the statement that the number of transistors in computers doubles about every two years, has held up quite well but stagnated a bit in recent years. Could one perhaps apply a sort of quantum version of this law, holding for qubits? It might not be at all impossible with such a broad spectrum of research going on in this field and new progress being made all the time.

References

- [1] T. Helgaker, P. Jorgensen, J. Olsen, *Molecular Electronic-structure Theory*, (2000), John Wiley & sons
- [2] A. Szabo, N. S. Ostlund, *Modern Quantum Chemistry: Introduction to Advanced Electronic Structure Theory*, (1989), McGraw-Hill, New York
- [3] M. Nielsen, I. Chuang, *Quantum Computation and Quantum Information*, (2010), Cambridge University Press
- [4] S. Kais, S. A. Rice, A. R. Dinner, *Quantum Information and Computation for Chemistry*, (2014), John Wiley & sons
- [5] R.P. Feynman, "Simulating physics with computers", *Int. J. Theor. Phys.* 21 (1982) 467–488.
- [6] D. Deutsch, (1985) *Proc. R. Soc. Lond. A* 400, 97-117.
- [7] P. R. Nagy, M. Kállay, (2019), *J. Chem. Theory Comput.*, 15, 10, 5275–5298
- [8] S. Ficher, D. Gunlycke, (2019), arXiv:1907.01493v2
- [9] D. Abrams, S. Lloyd, (1999), *Phys. Rev. Lett.* 83, 5162
- [10] A. Aspuru-Guzik, A. D. Dutoi, P. J. Love, M. Head-Gordon, (2005), arXiv:quant-ph/0604193
- [11] D. W. Berry, A. M. Childs, R. Cleve, R. Kothari, R. D. Somma, (2014), *Phys. Rev. Lett.* 114, 090502
- [12] A. Kay, (2020), arXiv:1809.03842.
- [13] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, (2017), *Nature* 549, 242.
- [14] P. Jordan and E. Wigner, (1928), *Z. Phys.* 47, 631
- [15] B. Bauer, S. B. Bravyi, M. Motta, G. Kin-Lic Chan, (2020), *Chem. Rev.* 120, 22, 12685–12717
- [16] S. B. Bravyi, A. Yu. Kitaev, (2002), *Ann. Phys.* 298: 210–26.

- [17] A. Tranter, P. J. Love, F. Mintert, P. V. Coveney, (2018), *J. Chem. Theory Comput.*, 14, 11, 5617–5630
- [18] L. Veis, J. Pittner, (2014), *J. Chem. Phys.* 140, 214111
- [19] G. Hao Low, I. L. Chuang, (2019), *Quantum* 3, 163
- [20] K. Sugisaki, S. Yamamoto, S. Nakazawa, K. Toyota, K. Sato, D. Shiomi, T. Takui, (2019), *Chem. Phys. Lett.* 737 100002
- [21] W. Ritz, (1909), DOI:10.1515/crll.1909.135.1
- [22] A. Peruzzo, J. McClean, P. Shadbolt, M. Yung, X. Zhou, P. J. Love, A. Aspuru-Guzik, J. L. O’Brien, (2014), *Nat. Commun.* 5, 4213
- [23] P.-L. Dallaire-Demers, J. Romero, L. Veis, S. Sim, A. Aspuru-Guzik, (2019), *Quantum Sci. Technol.* 4 045005
- [24] J. Lee, W. J. Huggins, M. Head-Gordon, K. B. J. Whaley, (2019), *Chem. Theory Comput.* 15, 311–324
- [25] H. R. Grimsley, S. E. Economou, E. Barnes, N. J. Mayhall, (2019), *Nat. Comm.* 10, 3007
- [26] K. Setia, J. D. Whitfield, (2018), *J. Chem. Phys.* 148, 164104
- [27] H. L. Tang, V. O. Shkolnikov, G. S. Barron, H. R. Grimsley, N. J. Mayhall, E. Barnes, S. E., Economou, (2021), *PRX Quantum* 2, 020310
- [28] J. R. McClean, et al., (2019), arXiv:1710.07629v5
- [29] G. Aleksandrowicz et al., (2019), *Qiskit: An open-source framework for quantum computing*
- [30] R. Srinivas, S. C. Burd, H. M. Knaack, R. T. Sutherland, A. Kwiatkowski, S. Glancy, E. Knill, D. J. Wineland, D. Leibfried, A. C. Wilson, D. T. C. Allcock, D. H. Slichter, (2021), *Nature* 597, pages 209–213
- [31] J. Hilder, D. Pijn, O. Onishchenko, A. Stahl, M. Orth, B. Lekitsch, A. Rodriguez-Blanco, M. Müller, F. Schmidt-Kaler, U. G. Poschinger, (2022), *Phys. Rev. X* 12, 011032

- [32] T. P. Harty, D. Allcock, C. J. Ballance, L. Guidoni, H. Janacek, N. Linke, D. Stacey, D. M. Lucas, (2014), *Phys. Rev. Lett.* 113, 220501
- [33] C. J. Ballance, T. P. Harty, N. M. Linke, M. A. Sepiol, D. M. Lucas, (2016), *Phys. Rev. Lett.* 117, 060504
- [34] T. Monz, P. Schindler, J. T. Barreiro, M. Chwalla, D. Nigg, W. A. Coish, M. Harlander, W. Hänsel, M. Hennrich, R. Blatt, (2011), *Phys. Rev. Lett.* 106, 130506
- [35] J. M. Chow, J. M. Gambetta, A. D. Corcoles, S. T. Merkel, J. A. Smolin, C. Rigetti, S. Poletto, G. A. Keefe, M. B. Rothwell, J. R. Rozen, M. B. Ketchen, M. Steffen, (2012), *Phys. Rev. Lett.* 109, 060501
- [36] F. Arute, et al., (2019), *Nature* 574, 505–510
- [37] L.-K. Chen, et al., (2017), *Class. Quantum Grav.* 29 224011
- [38] H.-S. Zhong, et al., (2018), *Phys. Rev. Lett.* 121, 250505.
- [39] D. Sundholm, J. Olsen, P. Å. Malmqvist, B. O. Roos, "Numerical MCSCF in One and Two Dimensions", in *Numerical Determination of the Electronic Structure of Atoms, Diatomic and Polyatomic Molecules*, (1989), Proc. NATO Advanced Research Workshop, Versailles 1988, Edited by M. Defranceschi and J. Delhalle. Dordrecht: Reidel, pp. 329-334.
- [40] D. Sundholm, J. Olsen, (1990), *Phys. Rev. A* 42, 2614-2621
- [41] D. Sundholm, J. Olsen, (1990), *Chem. Phys. Lett.* 171, 53-57
- [42] R. J. Bartlett, M. Musiał, (2007), *Rev. Mod. Phys.* 79, 291
- [43] T. D. Crawford, H. F. Schaefer, (2000), *Rev. Comput. Chem.* 14, 33
- [44] A. G. Taube, R. J. Bartlett, (2006), *Int. J. Quantum Chem.* 106: 3393-3401
- [45] S. Lehtola, (2019), *Int J Quantum Chem.* 119:e25968
- [46] J. Preskill, (2018), *Quantum* 2, 79
- [47] R. Serber, (1934) *Phys. Rev.* 45, 461–467.
- [48] R. Serber, (1934), *J. Chem. Phys.* 2, 697–710