MASARYKOVA UNIVERZITA
Přírodovědecká fakulta
Ústav fyziky kondenzovaných látek

Diplomová práce

Brno 2018 Jakub Wagner

MASARYKOVA UNIVERZITA
Přírodovědecká fakulta
Ústav fyziky kondenzovaných látek

Studium systémů mnoha elektronů metodou konfigurační interakce

Diplomová práce

Jakub Wagner

Vedoucí práce: Dominique Alain Geffroy, Ph.D.

Brno 2018

Bibliografický záznam

Autor: Jakub Wagner
Přírodovědecká fakulta, Masarykova univerzita
Ústav fyziky kondenzovaných látek

Název práce: Studium systémů mnoha elektronů metodou konfigurační interakce

Studijní program: Fyzika

Studijní obor: Fyzika kondenzovaných látek

Vedoucí práce: Dominique Alain Geffroy, Ph.D.

Akademický rok: 2017/2018

Počet stran: VII + 95

Klíčová slova: Metoda plné konfigurační interakce za použití metody Monte Carlo, FCIQMC, Konfigurační interakce, metoda Monte Carlo, Elektronový plyn, Simulace

Bibliographic entry

Author:	Jakub Wagner Faculty of Science, Masaryk University Department of condensed matter physics

Title of Thesis:	Investigation of many-electron systems by configuration interaction method
Degree Programme:	Physics
Field of Study:	Condensed matter physics
Supervisor:	Dominique Alain Geffroy, Ph.D.
Academic Year:	2017/2018
Number of Pages:	VII + 95
Keywords:	Full Configuration Interaction Quantum Monte Carlo, FCIQMC, Configuration Interaction, Monte Carlo method, Electron gas, Simulation

Abstrakt

Tato diplomová práce se zabývá metodou plné konfigurační interakce za použití metody Monte Carlo (FCIQMC). Popsány jsou základy kvantové mechaniky a na nich postavené kvantově-chemické metody (Hartree, Hartree-Fock, Konfigurační interakce), ze kterých FCIQMC vychází. Dále práce obsahuje kapitolu o metodě Monte Carlo, na níž navazuje metoda FCIQMC. Ta je podrobně rozebrána, včetně algoritmu pro komplexní čísla. Implementovány jsou celkem dva modely, „Elektronový plyn" a „Matice", která je schopna počítat libovolné modely, pokud je poskytnut jejich Hamiltonián. V kapitole s výsledky jsou prezentovány základní stavy a korespondující energie základních stavů studovaných modelů pro různé, jak fyzikální, tak čistě algoritmické parametry. Prezentován je současný stav vyvinutého programu a jeho možná vylepšení.

Abstract

This master thesis deals with the Full configuration interaction quantum Monte Carlo method (FCIQMC). Basics of quantum mechanics are described, followed by quantum-chemical methods (Hartree, Hartree-Fock, Configuration interaction) on which the FCIQMC is built on. Next, the thesis contains a chapter about the Monte Carlo method, followed by the FCIQMC method which is described in a great detail, including the algorithm which uses complex numbers. Two models are implemented in total, 'Electron gas' and 'Matrix' which is capable of computing arbitrary model whose Hamiltonian is provided. In the chapter with results, ground states and corresponding ground state energies are presented for various physical and purely algorithmical parameters. The current state of the developed program and its possible enhancements are presented as well.

MASARYKOVA UNIVERZITA
Přírodovědecká fakulta

ZADÁNÍ DIPLOMOVÉ PRÁCE

Akademický rok: 2017/2018

Ústav: Ústav fyziky kondenzovaných látek

Student: Bc. Jakub Wagner

Program: Fyzika

Obor: Fyzika kondenzovaných látek

Ředitel *Ústavu fyziky kondenzovaných látek* PřF MU Vám ve smyslu Studijního a zkušebního řádu MU určuje diplomovou práci s názvem:

Název práce: Studium systémů mnoha elektronů metodou konfigurační interakce

Název práce anglicky: Investigation of many-electron systems by configuration interaction method

Oficiální zadání:
Ač kvantová mechanika poskytuje hned několik rovnic, které na různé úrovni aproximace dobře popisují chování jednoho elektronu ve vakuu i ve vnějším poli, chování vzájemně interagujících elektronů se často řeší pouze pomocí různých typů jednoelektronových efektivních polí. Detailní a efektivní fyzikální popis chování mnoha elektronů stále zůstává nevyřešeným problémem. Přitom je zřejmé, že vícečásticové chování hraje významnou roli v řadě pozorovaných fyzikálních jevů. V současné době je jednou z nejpřesnějších metod studia víceelektronových systémů metoda konfigurační interakce (CI), která navazuje na Hartreeho-Fockovu metodu konstrukcí vlnové funkce z lineární kombinace Slaterových determinantů (SD); bohužel na úrok značné výpočetní zátěže. Student bude mít za úkol provést numerické řešení vícečásticového problému pomocí metody konfigurační interakce za využití metody Monte-Carlo. Obdržené výsledky pak srovná s řešením využívajícím exaktní diagonalizace, které bylo implementováno v nedávné době na Ústavu fyziky kondenzovaných látek. Práce může být vypracována v českém nebo anglickém jazyce.

Jazyk závěrečné práce: angličtina

Vedoucí práce: Dominique Alain Geffroy, Ph.D.

Konzultant: Mgr. Petr Klenovský, Ph.D.

Datum zadání práce: 2. 11. 2016

V Brně dne: 12. 9. 2017

Souhlasím se zadáním (podpis, datum):

Jakub Wagner	D Geffroy	Munzar
Bc. Jakub Wagner	Dominique Alain Geffroy, Ph.D.	doc. Mgr. Dominik Munzar, Dr.
student	**vedoucí práce**	**ředitel Ústavu fyziky kondenzovaných látek**

Acknowledgment

Computational resources were also supplied by the Ministry of Education, Youth and Sports of the Czech Republic under the Projects CESNET (Project No. LM2015042) and CERIT-Scientific Cloud (Project No. LM2015085) provided within the program Projects of Large Research, Development and Innovations Infrastructures.

Declaration

I declare that I have written the master thesis by myself, and have not used any sources without declaration in the text and appropriate citation. Thesis has not been submitted, in whole or in part, in any previous application for a degree.

Brno, 16 May 2016 .....

Jakub Wagner

Contents

Chapter 1

Basics of Quantum Mechanics

1. Postulates of quantum mechanics

Modern quantum mechanics relies on postulates, which are the result of years of effort towards the elaboration of the formalism. There are various exact wordings for these postulates e.g. [1] uses five postulates, while [2] introduces a thorough mathematical derivation, and mentions only two postulates, which are used to connect mathematical objects with their physical meaning. Here we state three postulates.

Postulate 1: State of the system, wavefunction

Every quantum state can be described by a normalized wavefunction ψ often written in Dirac notation [3] *as $|\psi\rangle$ to emphasize its vector nature. This state vector ψ lies in a state space $\mathcal{H}$ called the Hilbert space. The wavefunction ψ contains all information about the quantum state.*

The Hilbert space is an abstract complete vector space with inner product defined. Wavefunctions are vectors from this Hilbert space, and as such, every combination of wavefunctions from a Hilbert space describes a physical state. If two systems are described by two distinct Hilbert spaces, then a composite system can be described by the tensor product of those Hilbert spaces.

Postulate 2: Observables and operators

To every physical measurable quantity A corresponds a linear Hermitian operator $\hat{A}$ whose eigenvectors form a complete basis.

A Hermitian operator is an operator which is equal to its adjoint. In terms of matrices, this means that the matrix which represents such an operator is equal to its transposed complex conjugate transpose:

$$a_{ij} = \bar{a}_{ji}. \tag{1.1}$$

It can be shown that such a matrix is diagonalizable, with real eigenvalues.

The Hamiltonian $\hat{H}$ of a system corresponds to a physical observable: the total energy of the system. Therefore, it is always possible to use the eigenvectors of the Hamiltonian, which possess a clearly defined energy (the associated eigenvalue), in order to decompose any wavefunction Ψ:

$$|\Psi\rangle = \sum_i c_i \, |\Phi_i\rangle \, . \tag{1.2}$$

The eigenvector corresponding to the lowest eigenvalue is of special interest as it is the stable state at zero temperature. Note that depending on the symmetry of the system, it can be degenerate or not. In the nondegenerate case, we can write

$$|\Psi_{\mathrm{GS}}\rangle = |\Phi_0\rangle\,. \tag{1.3}$$

The energy of the system described by the wavefunction Ψ is obtained as the corresponding expectation value of the Hamiltonian operator:

$$E = \langle\Psi|\hat{H}|\Psi\rangle = \sum_i |c_i|^2 \epsilon_i, \tag{1.4}$$

where ϵ_i is the eigenvalue associated to the eigenvector $|\Phi_i\rangle$.

Postulate 3: Time evolution of a system

The time evolution of the wavefunction $|\psi(t)\rangle$ is described by the time-dependent Schrödinger equation:

$$i\frac{\partial|\psi(t)\rangle}{\partial t} = \hat{H}\,|\psi(t)\rangle\,. \tag{1.5}$$

It is convenient to introduce the time evolution operator operator $\hat{P}(t,t_0)$, which acts as a time translation operator:

$$|\psi(t)\rangle = \hat{P}(t,t_0)\,|\psi(t_0)\rangle\,. \tag{1.6}$$

Substituting this expression (1.6) into (1.5), and assuming that the Hamiltonian is not dependent on time, we can perform the integration and obtain the expression for the time evolution operator:

$$\begin{aligned} i\frac{\partial}{\partial t}\left(\hat{P}(t,t_0)\,|\psi(t_0)\rangle\right) &= \hat{H}\left(\hat{P}(t,t_0)\,|\psi(t_0)\rangle\right), \\ \frac{\partial\hat{P}(t,t_0)}{\partial t} &= -i\hat{H}\hat{P}(t,t_0), \\ \hat{P}(t,t_0) &= e^{-i(t-t_0)\hat{H}}. \end{aligned} \tag{1.7}$$

Substitution of this result into the definition (1.6) leads to:

$$|\psi(t)\rangle = e^{-i(t-t_0)\hat{H}}\,|\psi(t_0)\rangle\,. \tag{1.8}$$

If a Wick rotation [4] to imaginary time is used ($t \to i\tau$), Eq. (1.8) transforms to:

$$|\psi(\tau)\rangle = e^{-\tau\hat{H}}\,|\psi(0)\rangle\,. \tag{1.9}$$

2. Variational principle

The variational principle states that the energy of a system described by a wavefunction is greater or equal to the energy of the ground state:

$$\begin{aligned} E = \frac{\langle\psi|\hat{H}|\psi\rangle}{\langle\psi|\psi\rangle} &\geq E_{\mathrm{GS}}, \\ \sum_i |c_i|^2 \epsilon_i &\geq \epsilon_0. \end{aligned} \tag{1.10}$$

The variational principle is frequently used in the search for the ground state of a system, and its energy. A family of candidate wavefunctions is considered, which is dependent on a set of parameters. Varying these parameters increases or decreases the energy and changes the wavefunction. The lowest energy found is the best approximation to the ground state among the family of candidate wavefunctions.

Chapter 2

Many-Body Methods

In quantum mechanics, a system is described as a state in a Hilbert space. In order to study such a space and the elements it contains, it is desirable to choose a convenient basis. If the system contains a single particle, things are straightforward, because no interactions are present, and only one-body terms are present in the Hamiltonian. As soon as more than one particle are present and interact, things are much more complicated. It is nevertheless a convenient choice to build a basis for the multi-particle Hilbert space, which is built from the single-particle wavefunctions, in some appropriate way. The many-body methods described in this chapter are as many prescriptions for the construction of suitable many-particle basis.

All methods described in this chapter are variational, they use the variational principle (1.10). They are often referred to as quantum chemical methods, and differ mostly in the basis that they employ, or in the set of variable parameters which they use. The full configuration interaction quantum Monte Carlo method is a modern approach, built upon these theories.

1. Hartree

The first and most naive way of treating multi-particle systems is the Hartree method [5] introduced in 1928 by Douglas Rayner Hartree [6]. In this method, single-particle orbitals are varied and optimized in order to build a multi-particle basis. This method will be illustrated by the example of a single atom, where it was first used. It proposes to create a multi-particle wavefunction as a product of single-particle wavefunctions:

$$\psi(\xi_1, ..., \xi_N) \propto \phi_1(\xi_1) \cdot \phi_2(\xi_2) \cdot \ ... \ \cdot \phi_N(\xi_N). \tag{2.1}$$

To vary and improve the radial parts of the spin-orbitals, a self-consistent field approach is used. At first, a single electron is picked and all the other electrons are considered as a fixed distribution, a cloud of electric charge, through which the chosen electron i moves. Then we compute the Coulomb operator $\hat{J}_i$ between the chosen electron and the cloud of electric charge:

$$\hat{J}_i = \sum_{\substack{j=1 \\ j\neq i}}^{N} \frac{Q_i}{4\pi\epsilon_0} \int \frac{\rho_j}{\hat{r}_{ij}} dV_j = \sum_{\substack{j=1 \\ j\neq i}}^{N} \frac{e^2}{4\pi\epsilon_0} \int \frac{|\phi_j|^2}{\hat{r}_{ij}} dV_j = e'^2 \sum_{\substack{j=1 \\ j\neq i}}^{N} \int \frac{|\phi_j|^2}{\hat{r}_{ij}} dV_j. \tag{2.2}$$

where Q_i is a charge of i-th electron equal to $-e$, $\rho_j = -e|\phi_j|^2$ is a charge density per unit volume of the hypothetical charge cloud, $e'^2 = \frac{e^2}{4\pi\epsilon_0}$ and $\hat{r}_{ij}$ is the distance between i-th electron and an infinitesimal volume dV_j of the cloud.

With this, the interaction between the electrons is treated, but the potential from the nucleus must be considered:

$$\hat{V}_i(\hat{r}_i, \varphi_i, \theta_i) = \left[-\frac{Ze'^2}{\hat{r}_i} + \hat{J}_i \right]. \tag{2.3}$$

The nucleus is considered to be static in the centre of a coordinate system so $\hat{r}_i$ is the distance of the chosen electron from the nucleus.

Next, the central-field approximation is used. We assume that the effective potential acting on the electron can be approximated by a function dependent only on the distance from the nucleus r_i and thus the potential (2.3) can be integrated over angles:

$$\hat{V}_i(\hat{r}_i) = \frac{\int_0^{2\pi} \int_0^{\pi} \hat{V}_i(\hat{r}_i, \varphi_i, \theta_i) \sin\theta_i d\theta_i d\varphi_i}{\int_0^{2\pi} \int_0^{\pi} \sin\theta_i d\theta_i d\varphi_i}. \tag{2.4}$$

Then we use this potential and solve the Schrödinger equation for the chosen electron:

$$\left[-\frac{\hbar^2}{2m_e} \nabla_i^2 + \hat{V}_i \right] \phi_{i,R}^{(k+1)} = \epsilon_1 \phi_{i,R}^{(k+1)}. \tag{2.5}$$

From this equation, an improved radial part for the wavefunction ϕ_i of the i-th electron is obtained (The R in the subscript is to note that this is just a radial part of the wavefunction. The k index indicates the current iteration.). Then another electron is selected and the described procedure is repeated for it. After a number of iterations, the procedure converges. Then it is possible to compute the energy of the system. It includes the orbital energies ϵ_i, but the care needs to be taken that some double counting of Coulomb interaction between electrons has occurred through Eq. (2.5) (the orbital for the i-th electron with energy ϵ_i contains a Coulomb term from the cloud with the j-th electron and vice versa). The energy of the multi-particle system according to Hartree method is therefore:

$$E = \sum_i \epsilon_i - \sum_i \sum_{j>i} J_{ij}. \tag{2.6}$$

1.1. Summary of the method

The procedure of the Hartree self-consistent field method can be summarized this way:

1. Create the multi-particle wavefunction as a product of the single-particle wavefunctions.
2. Pick one electron.
3. Create a cloud from the other electrons.
4. Compute the interaction between the chosen i-th electron and the cloud.
5. Sum it up with the potential from the nucleus.

6. Average potential over angles.

7. Solve the Schrödinger equation (2.5) for the chosen electron to get an improved single-particle wavefunction.

8. Repeat until the total energy and the wavefunctions are converged.

This method uses the following approximations:

- The basis for multi-particle states is made up of product of single-particle wavefunctions.
- The Born-Oppenheimer approximation - Electrons are considered to move in a mean field created by the other electrons.
- Central-field approximation - It is assumed that the effective potential can be approximated by a function of radial distance only.

This method also does not respect the Pauli principle by itself.

Using the Hartree method, the energy operator for electron i takes the following form:

$$\hat{H}_i = \left[-\frac{1}{2}\nabla_i^2 - \frac{Z}{r_i} + \sum_{\substack{j=1 \\ j\neq i}}^{N} 2\hat{J}_j + \hat{J}_i \right]. \tag{2.7}$$

Thus for an electron in an atom (assuming N orbitals, all of them fully occupied), the energy of each electron has the following contributions:

- Kinetic energy.
- Potential energy between the electron and nucleus.
- Potential energy between the electron and two electrons on each of doubly occupied orbital.
- Potential energy between the electron and an electron in the same orbital.

2. Hartree-Fock

2.1. Slater determinant

The Hartree method has one fundamental flaw, it treats electrons as distinguishable particles, while they are indistinguishable. The way indistinguishable particles can be handled efficiently in quantum mechanics is described in Ref. [7], whose presentation we follow below.

At first the transposition operator $\hat{P}_{jk}$ needs to be introduced. When applied to a many-particle wavefunction, it interchanges the positions of two particles:

$$\hat{P}_{jk}\Psi(\xi_1, ..., \xi_j, ..., \xi_k, ..., \xi_N) = \Psi(\xi_1, ..., \xi_k, ..., \xi_j, ..., \xi_N). \tag{2.8}$$

If the transposition operator $\hat{P}_{jk}$ is applied twice, the original wavefunction must be restored, which means that

$$\begin{aligned} \hat{P}_{jk}\hat{P}_{jk} &= 1 = \text{Identity}, \\ \hat{P}_{jk}^{-1} &= \hat{P}_{jk}. \end{aligned} \tag{2.9}$$

For any states $|\Phi\rangle$ and $|\Psi\rangle$, any observable O, and any (j,k), the following holds:

$$\langle\Phi|\hat{O}|\Psi\rangle = \langle\hat{P}_{jk}\Phi|\hat{O}|\hat{P}_{jk}\Psi\rangle = \langle\Phi|\hat{P}_{jk}^{\dagger}\hat{O}\hat{P}_{jk}|\Psi\rangle\,. \tag{2.10}$$

This implies the operator identity:

$$\hat{O} = \hat{P}_{jk}^{\dagger}\hat{O}\hat{P}_{jk}. \tag{2.11}$$

If the operator $\hat{O}$ is chosen to be the identity operator, it follows that

$$\begin{aligned} 1 &= \hat{P}_{jk}^{\dagger}\hat{P}_{jk}, \\ \hat{P}_{jk}^{\dagger} &= \hat{P}_{jk}, \\ \hat{P}_{jk}^{-1} &= \hat{P}_{jk} = \hat{P}_{jk}^{\dagger}. \end{aligned} \tag{2.12}$$

In other words, operators corresponding to transpositions of indistinguishable particles are self-adjoint and unitary.

By multiplying Eq. (2.11) by $\hat{P}_{jk}$ from the left, we obtain

$$\hat{P}_{jk}\hat{O} = \hat{P}_{jk}\hat{P}_{jk}^{\dagger}\hat{O}\hat{P}_{jk} = \hat{O}\hat{P}_{jk}, \tag{2.13}$$

$$\begin{aligned} \left[\hat{O}, \hat{P}_{jk}\right] &= 0, \\ \left[\hat{H}, \hat{P}_{jk}\right] &= 0, \end{aligned} \tag{2.14}$$

i. e. the transposition operators commute with any observable, and with the Hamiltonian. As a consequence, all the transposition operators can be simultaneously diagonalized. Let us thus consider Φ, a common eigenvector of all the $\hat{P}_{jk}$, associated with the eigenvalue $a_{(jk)}$.

$$\begin{aligned} \hat{P}_{jk}\Phi &= a_{(jk)}\Phi, \\ \Phi = \hat{P}_{jk}^{2}\Phi &= a_{(jk)}^{2}\Phi, \\ \Rightarrow a_{(jk)} &= \pm 1. \end{aligned} \tag{2.15}$$

Indices (j, k) can be chosen arbitrarily and the wavefunction Φ is an eigenfunction of all transposition operators $\hat{P}_{jk}$. It can be further proved that the eigenvalues of all $\hat{P}_{jk}$ are identical:

$$\begin{aligned} \hat{P}_{jk} &= \hat{P}_{j1}\hat{P}_{2k}\hat{P}_{12}\hat{P}_{2k}\hat{P}_{1j}, \\ \hat{P}_{jk}\Phi &= a_{(1j)}^{2}a_{(2k)}^{2}a_{(12)}\Phi = a_{(12)}\Phi, \\ \hat{P}_{jk}\Phi &= a_{(12)}\Phi, \end{aligned} \tag{2.16}$$

According to eigenvalues, we can divide functions to:

- Symmetric, where $\hat{P}_{jk}\Phi = +\Phi$ for all (j, k).
- Antisymmetric, where $\hat{P}_{jk}\Phi = -\Phi$ for all (j, k).

The eigenfunctions of a system of identical particles are therefore either all symmetric or either all antisymmetric. Symmetric wavefunctions describe systems of bosons. Systems of fermions are described by antisymmetric wavefunctions.

A permutation operator can be decomposed as a product of transpositions:

$$\hat{P} = \prod \hat{P}_{jk}. \tag{2.17}$$

It acts on an antisymmetric wavefunction Ψ_A in the following way:

$$\hat{P}\Psi_A = \mathrm{sgn}(\hat{P})\Psi_A, \tag{2.18}$$

where

$$\begin{aligned} \mathrm{sgn}(\hat{P}) &= +1 \qquad \text{for even number of transpositions.} \\ \mathrm{sgn}(\hat{P}) &= -1 \qquad \text{for odd number of transpositions.} \end{aligned} \tag{2.19}$$

Finally, a properly antisymmetrized wavefunction can be built, using the single particle wave functions, as a Slater determinant:

$$\begin{aligned} \Psi_A(\xi_1, ..., \xi_N) &= \frac{1}{\sqrt{N!}} \sum_P \mathrm{sgn}(\hat{P})\hat{P}(\phi_1(\xi_1)\phi_2(\xi_2)...\phi_N(\xi_N)) \\ &= \frac{1}{\sqrt{N!}} \begin{vmatrix} \phi_1(\xi_1) & \phi_1(\xi_2) & \dots & \phi_1(\xi_N) \\ \phi_2(\xi_1) & \phi_2(\xi_2) & \dots & \phi_2(\xi_N) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_N(\xi_1) & \phi_N(\xi_2) & \dots & \phi_N(\xi_N) \end{vmatrix} \end{aligned} \tag{2.20}$$

where $\frac{1}{\sqrt{N!}}$ is is the normalization constant.

2.2. Fock operator

Using Slater determinants as a multi-particle basis set, the energy is represented by the Fock operator, which takes the following form for a system with fully occupied levels:

$$\hat{F}_i = \left[-\frac{1}{2}\nabla_i^2 - \frac{Z}{r_i} + \sum_{j=1}^{n/2} \left(2\hat{J}_j - \hat{K}_j \right) \right], \tag{2.21}$$

where $\hat{J}_j$ is again the Coulomb operator:

$$\begin{aligned} \hat{J}_{ij} &= e'^2 \int \frac{|\phi_i(\mathbf{r})|^2 |\phi_i(\mathbf{r}')|^2}{|\mathbf{r} - \mathbf{r}'|} d\mathbf{r} d\mathbf{r}' \\ &= e'^2 \left\langle \phi_i(1)\phi_j(2) \middle| \frac{1}{r_{12}} \middle| \phi_i(1)\phi_j(2) \right\rangle. \end{aligned} \tag{2.22}$$

and $\hat{K}_j$ is the exchange operator:

$$\begin{aligned}\hat{K}_{ij} &= e'^2 \int \frac{\phi_i^*(\mathbf{r})\phi_j^*(\mathbf{r})\phi_j(\mathbf{r}')\phi_i(\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|} d\mathbf{r}d\mathbf{r}' \\ &= e'^2 \left\langle \phi_i(1)\phi_j(2) \middle| \frac{1}{r_{12}} \middle| \phi_j(1)\phi_i(2) \right\rangle .\end{aligned} \tag{2.23}$$

The algorithm for computing the ground state wavefunction Ψ_{GS} follows the same steps as the one used in the Hartree method, provided the new basis and new energy operator are employed. Spin-orbitals are varied until a minimum in energy is reached. The energy obtained by this method is called Hartree-Fock energy E_{HF}.

2.3. Correlation energy

The Hartree-Fock energy only takes into account partially correlation effects - the part which is due to the Pauli exclusion principle. The term correlation energy is usually used in order to refer to the difference between the exact energy [1] and the Hartree-Fock energy:

$$E_{\mathrm{corr}} = E_{\mathrm{exact}} - E_{\mathrm{HF}}. \tag{2.24}$$

The Hartree-Fock approach ignores for instance the following effects:

- The Born-Oppenheimer (adiabatic) approximation - Electrons do not interact with each other instantaneously, as they do in the reality, but with a mean field created by all other electrons. This term is called the dynamic correlation.
- The Hartree-Fock method uses a single Slater determinant to describe the system, but the system's wavefunctions do not necessarily have the form of a single Slater determinant. This contribution is called static correlation.

On the other hand, the Hartree-Fock approach captures part of the electronic correlations. The form of the Slater determinants ensures that two electrons cannot occupy the same spin-orbital and thus, the correlation effects due to the Pauli principle are correctly captured. For comparison, the Hartree method does not capture any electronic correlation.

3. Configuration interaction

The configuration interaction (CI) method [8–10] is an approach where the ground state is looked for in the form of a linear combination of different Slater determinants, referred to as 'configurations'. The wavefunction is described by the following equation:

$$\Psi = \sum_i C_i \left| D_i \right\rangle . \tag{2.25}$$

[1] Note that in the whole thesis the term 'exact energy' is used for the non-relativistic energy of the system.

For a system of N electrons described by $2K$ spin-orbitals $\binom{2K}{N}$ Slater determinants can be constructed in total. The higher number of spin-orbitals is used, the better correlations are expressed. On the other hand the number of Slater determinants grows rapidly. In the limit where all allowed determinants are used, the method is called the full configuration interaction (FCI). Usually, however, not all possible determinants can be used, but only the most important are used and the CI basis is truncated.

The determinants in Eq. (4.1) can be divided into their respective excitation level n. Starting from the Hartree-Fock determinant, a n-excited determinant is produced when n single-particle orbitals of $|D_0\rangle$ are modified. If the single-particle orbital ϕ_a is replaced by the orbital ϕ_r, then the corresponding determinant is denoted as $|D_a^r\rangle$.

Eq. (4.1) can be expressed with excited determinants in two equivalent ways:

$$\begin{aligned}
\Psi &= C_0 |D_0\rangle + \sum_{ar} C_a^r |D_a^r\rangle + \sum_{\substack{a<b \\ r<s}} C_{ab}^{rs} |D_{ab}^{rs}\rangle \\
&\quad + \sum_{\substack{a<b<c \\ r<s<t}} C_{abc}^{rst} |D_{abc}^{rst}\rangle + \sum_{\substack{a<b<c<d \\ r<s<t<u}} C_{abcd}^{rstu} |D_{abcd}^{rstu}\rangle + \cdots , \\
\Psi &= C_0 |D_0\rangle + \left(\frac{1}{1!}\right)^2 \sum_{ar} C_a^r |D_a^r\rangle + \left(\frac{1}{2!}\right)^2 \sum_{\substack{ab \\ rs}} C_{ab}^{rs} |D_{ab}^{rs}\rangle + \\
&\quad + \left(\frac{1}{3!}\right)^2 \sum_{\substack{abc \\ rst}} C_{abc}^{rst} |D_{abc}^{rst}\rangle + \left(\frac{1}{4!}\right)^2 \sum_{\substack{abcd \\ rstu}} C_{abcd}^{rstu} |D_{abcd}^{rstu}\rangle + \cdots .
\end{aligned} \tag{2.26}$$

Terms involving n-excitations are often grouped by values of n:

$$\psi = C_0 |D_0\rangle + C_S |S\rangle + C_D |D\rangle + C_T |T\rangle + C_Q |Q\rangle + \cdots . \tag{2.27}$$

With this notation we can write an FCI matrix:

$$\begin{array}{c} \\ \\ \langle\psi_0| \\ \langle S| \\ \langle D| \\ \langle T| \\ \langle Q| \\ \vdots \end{array}
\begin{array}{c}
\begin{array}{cccccc} |D_a^r\rangle & |D_{ab}^{rs}\rangle & |D_{abc}^{rst}\rangle & |D_{abcd}^{rstu}\rangle & \cdots \\ |S\rangle & |D\rangle & |T\rangle & |Q\rangle & \cdots \end{array} \\
\begin{bmatrix}
\langle\psi_0|H|\psi_0\rangle & 0 & \langle\psi_0|H|D\rangle & 0 & 0 & \cdots \\
0 & \langle S|H|S\rangle & \langle S|H|D\rangle & \langle S|H|T\rangle & 0 & \cdots \\
\langle D|H|\psi_0\rangle & \langle D|H|S\rangle & \langle D|H|D\rangle & \langle D|H|T\rangle & \langle D|H|Q\rangle & \cdots \\
0 & \langle T|H|S\rangle & \langle T|H|D\rangle & \langle T|H|T\rangle & \langle T|H|Q\rangle & \cdots \\
0 & 0 & \langle Q|H|D\rangle & \langle T|H|Q\rangle & \langle Q|H|Q\rangle & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots
\end{bmatrix}
\end{array} \tag{2.28}$$

From this matrix following properties can be observed:

1. All matrix elements between Hartree-Fock ground state $|D_0\rangle$ and singly excited determinants are zero as a consequence of the Brillouin's theorem [8, 11], which states that singly excited determinants do not directly interact with Hartree-Fock determinant.

2. Matrix elements between determinants which differ by more than two spin-orbitals (are more than doubly excited) are zero as a consequence of Slater-Condon rules which will follow shortly.

The indirect proof of the Brillouin's theorem is simple. We can assume that the Hartree-Fock determinant $|\psi_0\rangle$ has the lowest energy and then look for a linear combination $c_0 |\psi_0\rangle + c_1 |S\rangle$ with a lower energy. A general property of determinants is that two determinants differing by a single row or column can be expressed as a single determinant. If follows that $c_0 |\psi_0\rangle + c_1 |S\rangle = |\psi_{\mathrm{LC}}\rangle$. If the new determinant $|\psi_{\mathrm{LC}}\rangle$ has lower energy than the Hartree-Fock determinant we arrive to a contradiction.

3.1. Second quantization

To clarify the FCI matrix it is good to derive Slater-Condon rules. These are the expressions for matrix elements between determinants. For writing Slater-Condon rules, the formalism of second quantization [7] is convenient. We will see later in Chapter 7 that this formalism is also very useful from the programming perspective.

We have seen that for a system with N particles, Slater determinants are a proper starting point. Second quantization takes this fact into account from the start, by the introduction of the creation operator $\hat{c}_k^\dagger$ and annihilation operator $\hat{c}_k$ which are mappings between Hilbert spaces of different particle number $\mathcal{H}(N)$. Operators have following properties:

$$\begin{aligned} \hat{c}_k &: \mathcal{H}(N) \to \mathcal{H}(N-1), \\ \hat{c}_k^\dagger &: \mathcal{H}(N) \to \mathcal{H}(N+1). \end{aligned} \tag{2.29}$$

An annihilation operator $\hat{c}_k$ annihilates a particle in the k-th spin-orbital, a creation operator $\hat{c}_k^\dagger$ creates a particle in the k-th spin-orbital.

The state with zero particle is called the vacuum state and is denoted as an empty ket:

$$\text{Vacuum state} \equiv |\emptyset\rangle \,. \tag{2.30}$$

Starting from this vacuum state,arbitrary Slater determinant can be build by the successive action of creation operators on it:

$$\hat{c}_a^\dagger \hat{c}_b^\dagger |\emptyset\rangle = |\phi_a \phi_b\rangle = |ab\rangle \,. \tag{2.31}$$

The detail action of these operators on Slater determinants is shown in Eqs. (2.32) and (2.33):

$$\hat{c}_j \frac{1}{\sqrt{N!}} \begin{vmatrix} \phi_1(\xi_1) & \dots & \phi_1(\xi_N) \\ \vdots & & \vdots \\ \phi_j(\xi_1) & \dots & \phi_j(\xi_N) \\ \vdots & & \vdots \\ \phi_N(\xi_1) & \dots & \phi_N(\xi_N) \end{vmatrix} = \hat{c}_j \frac{(-1)^{j-1}}{\sqrt{N!}} \begin{vmatrix} \phi_j(\xi_1) & \dots & \phi_j(\xi_N) \\ \phi_1(\xi_1) & \dots & \phi_1(\xi_N) \\ \vdots & & \vdots \\ \phi_{j-1}(\xi_1) & \dots & \phi_{j-1}(\xi_N) \\ \phi_{j+1}(\xi_1) & \dots & \phi_{j+1}(\xi_N) \\ \vdots & & \vdots \\ \phi_N(\xi_1) & \dots & \phi_N(\xi_N) \end{vmatrix} =$$

$$= \frac{(-1)^{j-1}}{\sqrt{(N-1)!}} \begin{vmatrix} \cancel{\phi_j(\xi_1)} & \dots & \cancel{\phi_j(\xi_{N-1})} & \cancel{\phi_j(\xi_N)} \\ \phi_1(\xi_1) & \dots & \phi_1(\xi_{N-1}) & \phi_1(\xi_N) \\ \vdots & & \vdots & \\ \phi_{j-1}(\xi_1) & \dots & \phi_{j-1}(\xi_{N-1}) & \phi_{j-1}(\xi_N) \\ \phi_{j+1}(\xi_1) & \dots & \phi_{j+1}(\xi_{N-1}) & \phi_{j+1}(\xi_N) \\ \vdots & & \vdots & \\ \phi_N(\xi_1) & \dots & \phi_N(\xi_{N-1}) & \phi_N(\xi_N) \end{vmatrix} = \tag{2.32}$$

$$= \frac{(-1)^{j-1}}{\sqrt{(N-1)!}} \begin{vmatrix} \phi_j(\xi_1) & \dots & \phi_j(\xi_{N-1}) \\ \phi_1(\xi_1) & \dots & \phi_1(\xi_{N-1}) \\ \vdots & & \vdots \\ \phi_{j-1}(\xi_1) & \dots & \phi_{j-1}(\xi_{N-1}) \\ \phi_{j+1}(\xi_1) & \dots & \phi_{j+1}(\xi_{N-1}) \\ \vdots & & \vdots \\ \phi_N(\xi_1) & \dots & \phi_N(\xi_{N-1}) \end{vmatrix}$$

$$\hat{c}_j^\dagger \frac{1}{\sqrt{N!}} \begin{vmatrix} \phi_1(\xi_1) & \dots & \phi_1(\xi_N) \\ \vdots & & \vdots \\ \phi_N(\xi_1) & \dots & \phi_N(\xi_N) \end{vmatrix} = \frac{1}{\sqrt{(N+1)!}} \begin{vmatrix} \phi_j(\xi_1) & \dots & \phi_j(\xi_N) & \phi_j(\xi_{N+1}) \\ \phi_1(\xi_1) & \dots & \phi_1(\xi_N) & \phi_1(\xi_{N+1}) \\ \vdots & & \vdots & \vdots \\ \phi_N(\xi_1) & \dots & \phi_N(\xi_N) & \phi_N(\xi_{N+1}) \end{vmatrix} =$$

$$= \frac{(-1)^{j}}{\sqrt{(N+1)!}} \begin{vmatrix} \phi_1(\xi_1) & \dots & \phi_1(\xi_{N+1}) \\ \vdots & & \vdots \\ \phi_i(\xi_1) & \dots & \phi_i(\xi_{N+1}) \\ \phi_j(\xi_1) & \dots & \phi_j(\xi_{N+1}) \\ \phi_{i+1}(\xi_1) & \dots & \phi_{i+1}(\xi_{N+1}) \\ \vdots & & \vdots \\ \phi_N(\xi_1) & \dots & \phi_N(\xi_{N+1}) \end{vmatrix} \tag{2.33}$$

Another way to express a Slater determinant is to use the occupation number formalism, where the vacuum state is a ket filled with as many zeros as we have spin-orbitals:

$$\begin{array}{rcl} & & \phi_a \ \phi_b \ \phi_c \ \phi_d \ \cdots \\ \text{Vacuum state} & = & |\ 0 \quad 0 \quad 0 \quad 0 \quad \cdots\rangle\,. \end{array} \tag{2.34}$$

Empty (filled) orbitals are marked with zero (one). An example of the action of a creation operator follows:

$$\begin{array}{cccccccccc} & & \phi_a & \phi_b & \phi_c & \phi_d & & \phi_a & \phi_b & \phi_c & \phi_d \\ c_a^\dagger & \vert & 0 & 0 & 0 & 0 & \rangle & = \vert & 1 & 0 & 0 & 0 & \rangle . \end{array} \tag{2.35}$$

3.2. Slater-Condon rules

Since the Slater determinants are the building blocks of the CI approach, it is useful to be able to evaluate the matrix elements of the Hamiltonian in that basis. These expressions of the integrals of one and two-body operators form the so-called Slater-Condon rules [8, 12]. They can be simply derived using the formalism of second quantization stated above. We use $\hat{O}_1$ for one-body operators and $\hat{O}_2$ for two-body operators here. We use the following shorthand notation for the integrals:

$$\begin{aligned} \langle i|\hat{O}_1|j\rangle &= \int \phi_i^*(x_1)\hat{O}_1(x_1)\phi_j(x_i)dx_1, \\ \langle ij|kl\rangle &= \langle ij|\hat{O}_2|kl\rangle = \int \phi_i^*(x_1)\phi_j^*(x_2)\hat{O}_2\phi_k(x_1)\phi_l(x_2), \\ \langle ij||kl\rangle &= \langle ij|kl\rangle - \langle ij|lk\rangle \\ &= \int \phi_i^*(x_1)\phi_j^*(x_2)\hat{O}_2(1-\hat{P}_{12})\phi_k(x_1)\phi_l(x_2). \end{aligned} \tag{2.36}$$

For each operator we can distinguish three cases:

- Case 1: Determinants are identical.
- Case 2: Determinants differ by single spin-orbital.
- Case 3: Determinants differ by two spin-orbitals.
- Case 4: Determinants differ by three or more spin-orbitals.

In the fourth case, the matrix element is always zero. Using this notation we introduce the Slater-Condon rules which are stated below.

For one-particle operators:

Case 1: $|K\rangle = |\cdots mn\cdots\rangle$

$$\langle K|\hat{O}_1|K\rangle = \sum_m^N \langle m|\hat{O}_1|m\rangle , \tag{2.37}$$

Case 2: $|K\rangle = |\cdots mn\cdots\rangle$
$|L\rangle = |\cdots pn\cdots\rangle$

$$\langle K|\hat{O}_1|L\rangle = \langle m|\hat{O}_1|p\rangle , \tag{2.38}$$

Case 3: $|K\rangle = |\cdots mn\cdots\rangle$
$|L\rangle = |\cdots pq\cdots\rangle$

$$\langle K|\hat{O}_1|L\rangle = 0. \tag{2.39}$$

For two-particle operators

Case 1: $|K\rangle = |\cdots mn\cdots\rangle$

$$\langle K|\hat{O}_2|K\rangle = \frac{1}{2}\sum_{m}^{N}\sum_{n}^{N}\langle mn||mn\rangle\,, \tag{2.40}$$

Case 2: $|K\rangle = |\cdots mn\cdots\rangle$
$|L\rangle = |\cdots pn\cdots\rangle$

$$\langle K|\hat{O}_2|L\rangle = \sum_{n}^{N}\langle mn||pn\rangle\,, \tag{2.41}$$

Case 3: $|K\rangle = |\cdots mn\cdots\rangle$
$|L\rangle = |\cdots pq\cdots\rangle$

$$\langle K|\hat{O}_2|L\rangle = \langle mn||pq\rangle\,. \tag{2.42}$$

3.3. Example - hydrogen molecule

In order to get some insight into the way configuration interaction works, its use is demonstrated on the example of the hydrogen molecule. Single atoms are described by atomic orbitals marked by χ. In the case of hydrogen, it is a $1s$ atomic orbital. As atoms get closer to one another, a chemical bound is created and we can not describe the system as two isolated atoms but as a molecule. For this, we use molecular orbitals. Atomic orbitals, when they create molecular orbitals, interfere constructively or destructively. The result is that one molecular orbital has lower energy than the atomic orbitals, while the other has higher energy. This is illustrated in Fig. 2.1.

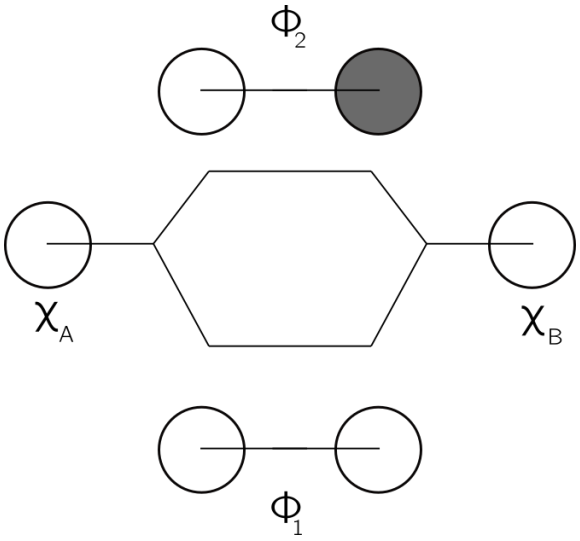


Figure 2.1: Creation of a H_2 molecule from atomic orbitals (marked by χ). The energy levels (horizontal lines) are shown and the corresponding molecular orbitals (marked with ϕ) are below or above them. The gray color indicates negative values of the associated atomic wavefunction.

There are two possible orbitals, or four spin-orbitals, and two electrons. As was stated before, $\binom{2K}{N}$ determinants can be constructed, so a total of six possible Slater determinants can be created. They are shown in Fig. 2.2.

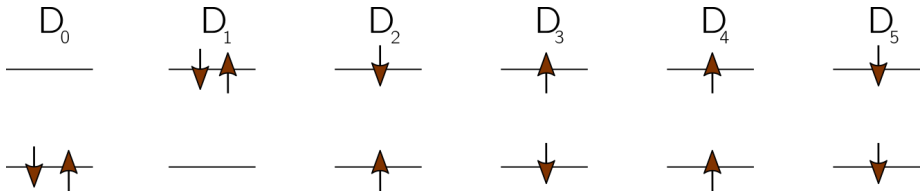


Figure 2.2: Possible determinants describing H_2 molecule from Fig. 2.1.

Now, when determinants were created out of spin-orbitals, symmetry of the problem is considered and all determinants with different symmetry, than the Hartree-Fock determinant D_0 has, are excluded.

Here, the inversion through the center of symmetry, which lies in the midpoint between atoms, is considered. Used molecular orbitals have different symmetry, ϕ_1 is symmetric (even, gerade) and ϕ_2 is antisymmetric (odd, ungerade). The symmetry of a determinant is determined by molecular orbitals it consists of. The Hartree-Fock orbital has two electrons in a symmetric molecular orbital. Symmetric function times symmetric function is again a symmetric function, so the determinant is of gerade symmetry. The doubly excited determinant D_1 is of gerade symmetry as well, but other determinants are of ungerade symmetry so they are excluded.

Symmetry considerations greatly reduce the number of determinants it is necessary to consider. For larger molecules with more spin-orbitals, there are many more possible determinants, even after considering symmetry issues, so that truncation

can be necessary. Here, we are left with two possible determinants so this step is skipped.

Now only the spatial parts of the determinants are considered:

$$\begin{aligned} D_0 &= \phi_1(1)\phi_1(2) \\ &= (\chi_A(1) + \chi_B(1))(\chi_A(2) + \chi_B(2)) \\ &= \chi_A(1)\chi_A(2) + \chi_A(1)\chi_B(2) + \chi_B(1)\chi_A(2) + \chi_B(1)\chi_B(2), \\ D_1 &= \phi_2(1)\phi_2(2) \\ &= (\chi_A(1) - \chi_B(1))(\chi_A(2) - \chi_B(2)) \\ &= \chi_A(1)\chi_A(2) - \chi_A(1)\chi_B(2) - \chi_B(1)\chi_A(2) + \chi_B(1)\chi_B(2). \end{aligned} \tag{2.43}$$

Eq. (2.43) can be inserted into the CI wavefunction (4.1):

$$\begin{aligned} \Psi_{\mathrm{CI}} &= c_0 D_0 + c_1 D_1 \\ &= (c_0 + c_1)(\chi_A(1)\chi_A(2) + \chi_B(1)\chi_B(2)) + (c_0 - c_1)(\chi_A(1)\chi_B(2) + \chi_A(2)\chi_B(1)). \end{aligned} \tag{2.44}$$

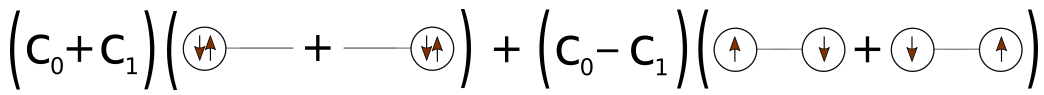


Figure 2.3: The graphical representation of terms in Eq. (2.44)

The configuration interaction method is variational, and the coefficients c_0 and c_1 thus need to be optimized, under the constraint of normalization. From Eq. (2.44), illustrated by Fig. 2.3, it is apparent that the CI method is able to describe both ionic configurations where two electrons sit on a single atom and covalent configurations, where electrons are split equally between the atoms.

The configuration interaction method can be now compared with the Hartree-Fock method. If the Hartree-Fock method was used, the system would be described by determinant D_0. This means that the system is equally described by ionic and covalent configurations. This treatment does not allow the correct determination of the bond length for example.

Chapter 3

Monte Carlo Method

1. Introduction

The Monte Carlo (MC) method is an approach used to solve various problems in physics [13, 14] and in other areas. Although the Monte Carlo method is extremely general and the variety of MC algorithms is huge, there are a few principles all of them have in common:

- Ergodicity - all possible states of the system are attainable, i. e. it is possible to reach each state with a non-zero probability
- Equiprobability - each configuration can be selected an equal probability

2. Markov Chains

A stochastic process is a process where the probability P of an observable X to have the value S changes with time. A Markovian process is a stochastic process whose future state depends only on its current state. Mathematically it can be stated by the following equation:

$$P(X_t = S_i|X_{t-1} = S_{i-1}|X_{t-2} = S_{i-2}|\cdots) = P(X_t = S_i|X_{t-1} = S_{i-1}). \tag{3.1}$$

The future state is conditionally dependent only on the current state and is independent on the previous states of the stochastic process. The sequence of states X_t is called a Markov chain. The conditional probability (3.1) can be understood as the transition probability from state i to state j:

$$W_{ij} = W(S_i \to S_j) = P(X_t = S_j|X_{t-1} = S_i). \tag{3.2}$$

The probability W_{ij} lies in the interval $[0, 1]$ and we further require $\sum_j W_{ij} = 1$ for fixed i. It is possible to compute the probability of a system being in the state S_i at time t:

$$P(X_t = S_i) = W_{ji}P(X_{t-1} = S_j), \tag{3.3}$$

from which follows the master equation:

$$\frac{\partial P(X_t = S_j)}{\partial t} = \sum_i \left(-P(X_t = S_j)W_{ji} + P(X_t = S_i)W_{ij}\right), \tag{3.4}$$

which expresses that the total probability remains conserved. At equilibrium, where $\frac{\partial P(X_t=S_j)}{\partial t} = 0$, Eq. (3.4) is reduced to the detailed balance condition:

$$W_{ij}P(X_t = S_i) = W_{ji}P(X_t = S_j). \tag{3.5}$$

3. Metropolis algorithm

The Metropolis algorithm [15] can be used to obtain transition probabilities between different states of a Markov chain, so that the detailed balance condition is respected. Assuming that the distribution $f(S)$ of all configurations S at equilibrium is known, it gives an expression for the transition probability

$$W_{ij} = \min\left(1, \frac{f(S_j)}{f(S_i)}\right). \tag{3.6}$$

Eq. (3.6) fulfils the detailed balance condition (3.5), which is a sufficient condition for the Markov chain to converge to the distribution of a function. When the Markov chain contains enough samples, $t \to \infty$, the probability to find the system in the i-th state, $P(X_t = S_i)$, will be proportional to the distribution of a function $f(S_i)$.

4. Projector Monte Carlo

The projector Monte Carlo method is a technique which allows the computation of the ground state $|\Psi_{\mathrm{GS}}\rangle$ from an arbitrary state $|\Psi\rangle$ which has a non-zero overlap with it. For this the projection operator $\hat{P}_\tau$ is used:

$$\hat{P}_\tau = e^{-\tau \hat{H}}, \tag{3.7}$$

where τ is the imaginary time. Every state at time τ can be expressed as the result of the action of a projection operator on the initial state:

$$|\Psi(\tau)\rangle = \hat{P}_\tau |D_0\rangle. \tag{3.8}$$

The idea of the projector Monte Carlo approach is to divide the time τ into elementary steps:

$$\tau = n \cdot \delta\tau. \tag{3.9}$$

Then Eqs. (3.7) and (3.9) can be merged and the projection operator is expressed as a sequence of projection operators:

$$\hat{P}_\tau = \hat{P}_{n\cdot\delta\tau} = e^{-n\cdot\delta\tau(\hat{H})} = (\hat{P}_{\delta\tau})^n. \tag{3.10}$$

This can be viewed as a Markov chain:

$$\Psi^{(n+1)} = \hat{P}_{\delta\tau}\Psi^{(n)}. \tag{3.11}$$

By repeated application of the projection operator on a starting state $|D_0\rangle$, this state projects out its components on other states. In the long-time limit the projection operator no longer has any effect and the steady state is reached. This state corresponds to the ground state wavefunction:

$$|\Psi_{\mathrm{GS}}\rangle = \lim_{\tau\to\infty} \hat{P}_\tau |D_0\rangle. \tag{3.12}$$

In the full configuration interaction quantum Monte Carlo method we use the configuration interaction basis composed from Slater determinants, but convergence to the ground state is better seen from the eigenvector basis. Let

$$|\Psi\rangle = \sum_i c_i |\Phi_i\rangle , \tag{3.13}$$

be a general wavefunction composed from eigenfunctions $|\Phi_i\rangle$ of the Hamiltonian $\hat{H}$. In this basis the Hamiltonian is diagonal and to each eigenfunction corresponds one value on the Hamiltonian diagonal, E_i. Assuming that eigenvalues are nondegenerate, the ground state is the state with the lowest such eigenvalue. In this basis, the projected state can be written as:

$$|\Psi(\tau)\rangle = \hat{P}_\tau |\Psi(0)\rangle = \sum_i c_i |\Phi_i\rangle e^{-\tau E_i}. \tag{3.14}$$

As the ground state energy has the lowest value, which we choose to be $E_0 = 0$ without loss of generality, the exponent associated to E_0 is unity, while all other exponents are strictly smaller than one. In the long-time limit, all coefficients vanish except c_{GS}:

$$\lim_{\tau\to\infty} \hat{P}_\tau |\Psi(0)\rangle = \lim_{\tau\to\infty} \sum_i c_i |\Phi_i\rangle e^{-\tau E_i} = c_o |\Phi_0\rangle = |\Psi_{\mathrm{GS}}\rangle . \tag{3.15}$$

Chapter 4

Full Configuration Interaction Quantum Monte Carlo

1. Principle

The full configuration interaction quantum Monte Carlo (FCIQMC) is a method where the full configuration interaction basis of Slater determinants (4.1) is used together with the projector quantum Monte Carlo method to find the ground state of a quantum system:

$$\Psi = \sum_i C_i \left| D_i \right\rangle . \tag{4.1}$$

To develop this method we have to start with the imaginary-time Schrödinger equation:

$$\frac{\partial \psi}{\partial \tau} = -\hat{H}\psi. \tag{4.2}$$

As was described in Chapter 1 the following expression for a wavefunction at imaginary time τ (assuming a time-independent Hamiltonian) exists:

$$\psi(\tau) \propto e^{-\tau \hat{H}} \psi(0). \tag{4.3}$$

The long-time limit of this equation projects the starting state D_0 onto excited states, resulting into the ground state wavefunction:

$$\Psi_0 = \lim_{\tau \to \infty} e^{-\tau(\hat{H} - E_0)} D_0. \tag{4.4}$$

Writing Eq. (4.1) again with an explicit dependence of the CI coefficients on the imaginary time τ:

$$\Psi(\tau) = \sum_i C_i(\tau) \left| D_i \right\rangle , \tag{4.5}$$

this equation can be inserted into the imaginary-time Schrödinger equation (4.2) and retrieve a set of coupled linear first-order differential equations:

$$\begin{aligned} \sum_i \frac{\partial C_i(\tau)}{\partial \tau} \left| D_i \right\rangle &= -\sum_j C_j(\tau) \hat{H} \left| D_j \right\rangle , \\ \frac{\partial C_i(\tau)}{\partial \tau} &= -\sum_j \left\langle D_i | \hat{H} | D_j \right\rangle C_j(\tau). \end{aligned} \tag{4.6}$$

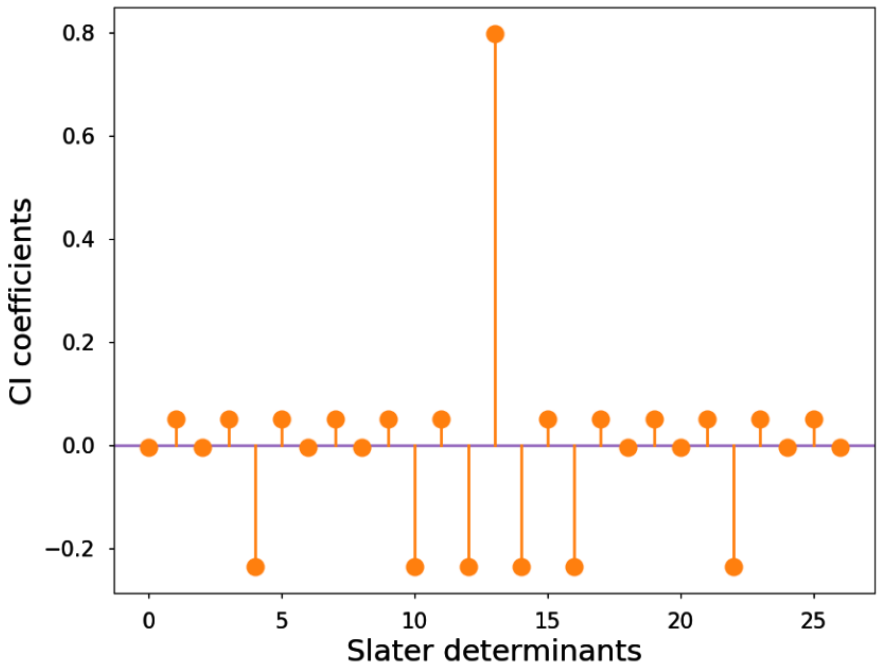


Figure 4.1: Coefficients of the wavefunction (vertical axis) expressed in the basis of Slater determinants (horizontal axis).

Next, we proceed to express the CI coefficients $C_i(\tau)$ as a sum of infinitesimal elements. We consider a population of $N_W(\tau)$ walkers living in the Slater determinant space [1]

A population of $N_i(\tau)$ walkers is associated with each determinant. Each walker has a sign $s_j = \pm 1$. The amplitude $C_i(\tau)$ of the i-th determinant is defined as the sum of the signs of the walkers living on it:

$$C_i(\tau) \propto \sum_{j=1}^{N_{\mathrm{W}}(\tau)} s_j \delta_{i,i_j} = N_i(\tau), \tag{4.7}$$

where the sum runs over all walkers and the meaning of δ_{i,i_j} is that the j-th walker has to live on the i-th determinant. This step of the discretization of the coefficients of the wavefunction expressed in the basis of Slater determinants is illustrated in Figs. 4.1, 4.2 and 4.3.

The set of equations (4.6) is solved by a stochastic integration using a population dynamics algorithm which is described in the next section. As the population grows the relative weight of each individual walker decreases, and becomes infinitesimal, as can be seen by comparing Figs. 4.2 and 4.3.

[1] In the literature 'walkers' can be called 'psips', according to the Ref. [16], as they do not move through the Slater determinant space directly. Algorithms where walkers move through the configuration space exists, for example [17]. We will, however, use the term 'walkers'.

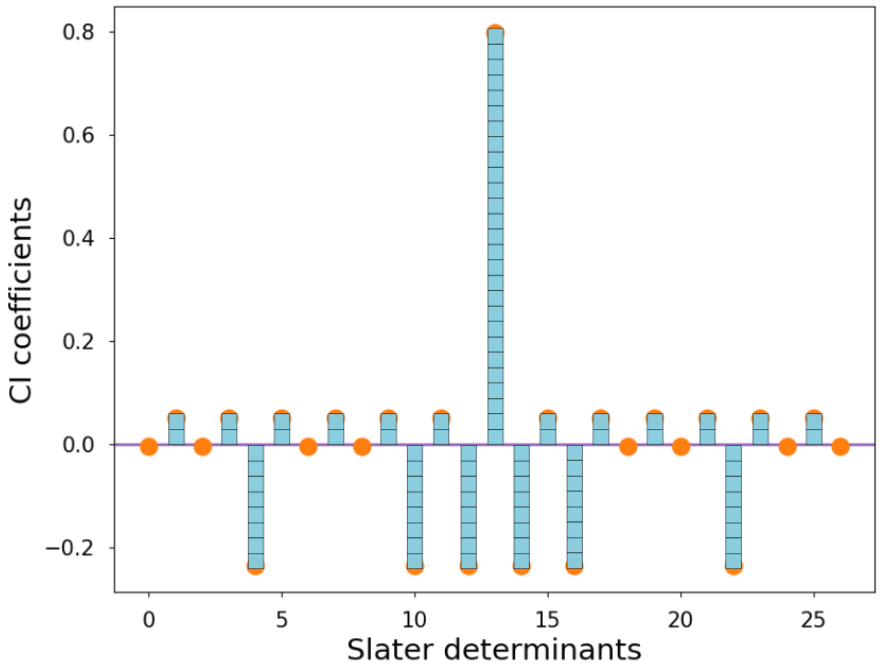


Figure 4.2: The coefficients of the wavefunction (vertical axis) are discretized, thus making up a small population of walkers on each Slater determinant (horizontal axis).

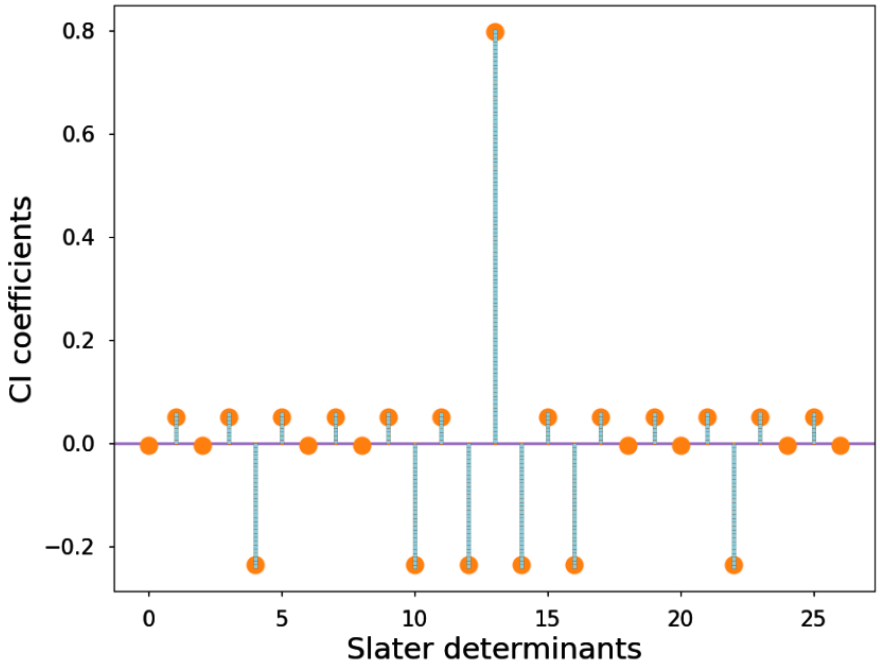


Figure 4.3: Representation of CI coefficients in the limit of large population.

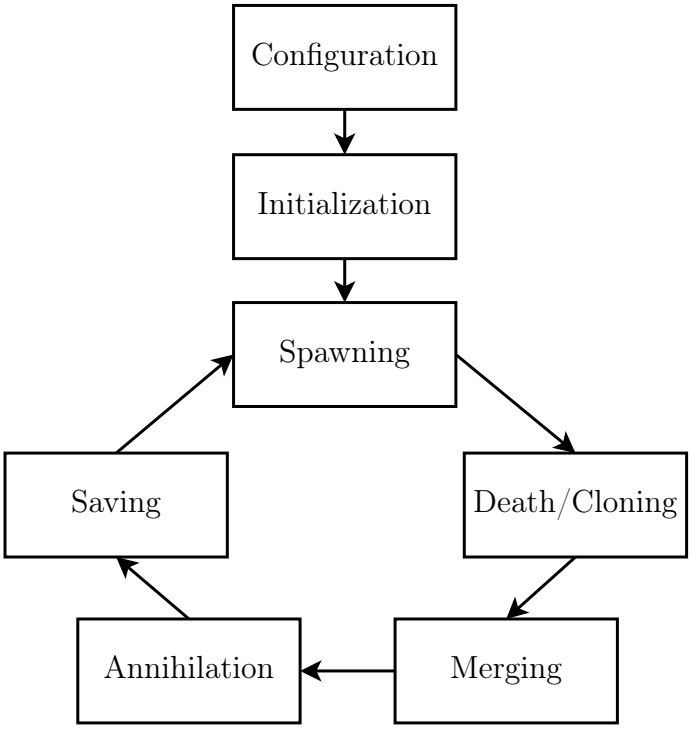


Figure 4.4: Overview of the population dynamics algorithm

2. Population dynamics algorithm (real amplitudes)

The algorithm followed by the population consists of three steps which are repeated and simulate the set of coupled differential equations in imaginary time. These steps are:

- Spawning
- Death or cloning
- Annihilation

An overview of the population dynamics algorithm is shown in the diagram 4.4.

At first we need to configure the simulation and the set of parameters. Then the simulation is initialized and the main cycle starts.

The population of walkers before spawning starts is called the parent population. The spawning step creates a new separated population, called the child population. These two populations, old and new, are merged after the death/cloning step. The merged population is the parent population for the next iteration. In the diagram, there is no end to the simulation as we usually run simulations for a predetermined time and interrupt it when the time is exceeded. The better way which is not implemented would be to monitor the energy and stop when it does not change by more than some fixed amount, between two steps.

In order to describe why three steps of the population dynamics algorithm are needed it is helpful in terms of population size to substitute (4.7) into equation (4.6).

The diagonal and nondiagonal terms can then be separated:

$$\frac{\partial N_i(\tau)}{\partial \tau} = -H_{ii}N_i - \sum_{j \neq i} H_{ij}N_j. \tag{4.8}$$

An additional energy shift S can be included in the diagonal terms of this expression:

$$\frac{\partial N_i(\tau)}{\partial \tau} = -(H_{ii} - S)N_i - \sum_{j \neq i} H_{ij}N_j. \tag{4.9}$$

Its role is the population control and it will be discussed later in Sec. 4.4..

The spawning step simulates the second term on the right side of Eq. (4.8) underlined in blue. Walkers on different determinants affect the population on the current determinant $|D_i\rangle$.

The first term on the right side of Eq. (4.8) underlined in violet color is simulated by the death/cloning step. Walkers on the current determinant can die and decrease the population on the current determinant. They can also be cloned, in which case the population increases.

The last step is the annihilation. This step can not be seen in Eq. (4.8) but it is of great importance for the efficiency of the algorithm. In this step walkers which are on the same determinant but has different signs are removed according to annihilation rules.

These steps are described in detail in following subsections.

2.1. The spawning step

The first step, the spawning step, simulates the second term on the right side of Eq. (4.8):$-\sum_{j \neq i} H_{ij}N_j$. A diagram of the spawning step is presented in Fig. 4.5. This step is executed for each parent walker. Existing walkers are selected one by one. Every walker is located on a determinant $|D_j\rangle$. A coupled determinant $|D_i\rangle$ is chosen with a normalized selection probability P_{C}. For computing the selection probability the number N_{C} of coupled determinants with the determinant $|D_i\rangle$ of the walker needs to be enumerated [2]. The probability is then computed as

$$P_{\mathrm{C}} = \frac{1}{N_{\mathrm{C}}}. \tag{4.10}$$

Afterwards, the spawning probability is determined as

$$P_{\mathrm{S}} = \frac{\delta\tau |H_{ij}|}{P_{\mathrm{C}}}. \tag{4.11}$$

If $P_{\mathrm{S}} < 1$ as in the diagram 4.5 a child walker is tried to be spawned on the selected coupled determinant $|D_i\rangle$. A random number from the interval $[0, 1]$ is generated.

[2] It is possible to skip the enumeration and set N_{C} to the total number of all possible determinants, but this is highly inefficient because the success of the spawning step depends on the matrix element $\langle D_i|\hat{H}|D_j\rangle$ which is zero for uncoupled determinants.

If the spawning probability is larger than this number, spawning is successful and the new walker is created. It may happen, for example when a high timestep $\delta\tau$ is adopted, that the spawning probability will be larger than one. In this case $\lfloor P_{\mathrm{S}} \rfloor$ child walkers are spawned on the selected coupled determinant $|D_i\rangle$ and one additional walker is spawned with the probability $P_{\mathrm{S}} - \lfloor P_{\mathrm{S}} \rfloor$.

A new walker spawned on the determinant $|D_i\rangle$ is a contribution to $\frac{\partial N_i(\tau)}{\partial \tau}$ and its sign σ is determined by the matrix element H_{ij} and the sign of the parent sign(N_j).

$$\begin{gathered} \frac{\partial N_i(\tau)}{\partial \tau} \sim \sum_{j \neq i} -N_j H_{ij}. \\ \sigma = -\mathrm{sign}(N_j H_{ij}) \qquad H_{ij} \begin{cases} < 0 & \text{Same sign as parent.} \\ > 0 & \text{Opposite sign than parent.} \end{cases} \end{gathered} \tag{4.12}$$

2.2. The death/cloning step

The second step is the death/cloning step which simulates the first term on the right side of Eq. (4.8):

$$\frac{\partial N_i(\tau)}{\partial \tau} \sim -(H_{ii} - S) N_i. \tag{4.13}$$

The diagram of this step is presented in the Fig. 4.6.

For each parent walker the probability of dying is computed:

$$P_{\mathrm{D}} = \delta\tau (H_{ii} - S). \tag{4.14}$$

. Based on this probability two things can happen:

$$P_{\mathrm{D}} \begin{cases} > 0 & \text{Walker dies with probability } P_{\mathrm{D}}. \\ < 0 & \text{Walker is cloned with probability } |P_{\mathrm{D}}|. \end{cases} \tag{4.15}$$

The dying is realized randomly. It can happen that $P_D > 1$, but in this case only one walker dies. It can not happen that more than one walker is removed in the death/cloning step for a single walker. On the other side, cloning is done in the same way as the spawning. If the death probability is negative and $|P_{\mathrm{D}}| > 1$, $\lfloor |P_{\mathrm{D}}| \rfloor$ child walkers are created on the determinant $|D_i\rangle$ of the current walker and one additional walker is cloned with the probability $|P_{\mathrm{D}}| - \lfloor |P_{\mathrm{D}}| \rfloor$. The sign of the cloned walkers is the same as the sign of their parents.

It is very important to note that the death/cloning step is done only for parent walkers. Child walkers from the previous step are excluded. The reason is that the spawning and the death/cloning step are computed for a single timestep. They simulate the differential equation for the same discrete time step. They can be exchanged, but from the programming point of view it is more practical to spawn new walkers at first and then remove old walkers. Indeed, if these steps are exchanged, there would be no child walkers in the death/cloning step as the spawning step would yet follow. It would be possible to compute the death/cloning step only for parent walkers as it is done in the current order.

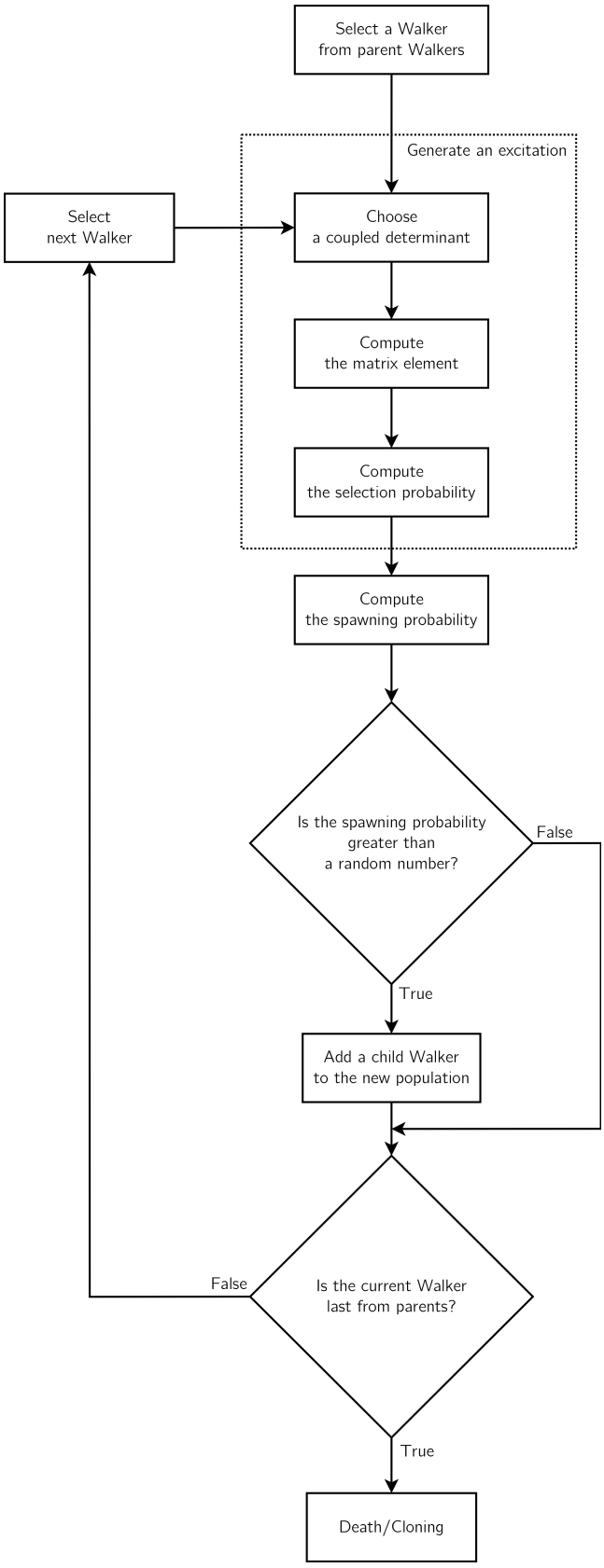


Figure 4.5: Diagram of the spawning step

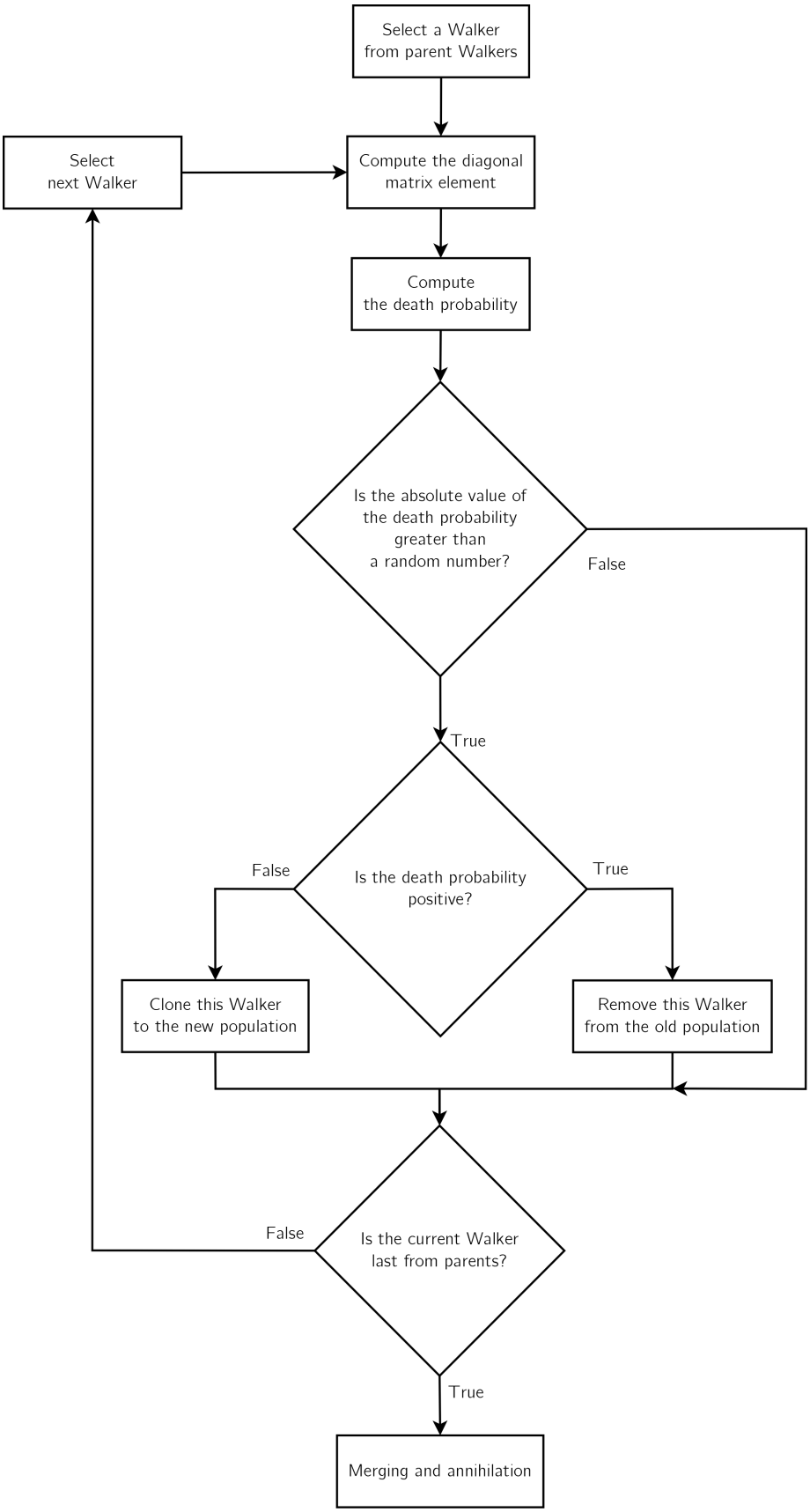


Figure 4.6: Diagram of the death/cloning step

2.3. The annihilation step

The third step is the annihilation, which finalizes the simulation of a single timestep of Eq. (4.8). It was stated in Eq. (4.7) that the amplitude C_i of a wavefunction on a determinant $|D_i\rangle$ is the signed sum of the walkers associated to this determinant. Due to the spawning and the cloning which create new children walkers the population can be composed of walkers which are located on the same determinant and have the same sign, or the opposite sign.

As the signs of the children walkers depend on the signs of the parents through Eq. (4.12), populations with different signs can effectively cancel each other when they spawn new walkers as shown in Fig. 4.7. The CI coefficient of both determinants on the figure is zero even though the populations on the same determinants are nonzero.

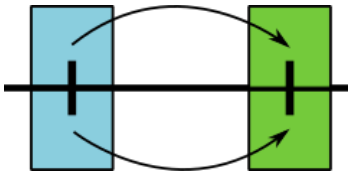


Figure 4.7: Ineffective spawning, populations with different signs (above the axis positive, under the axis negative) cancel each other.

The annihilation step helps mitigate the instability which can arise if one keeps walkers with different signs on a single determinant, a point which is related to the infamous sign problem in Monte Carlo approaches. We present the diagram of the annihilation step in Fig. 4.8. We iterate over all determinants which have a nonzero population of walkers and we subtract populations with different signs from each other on every determinant. After the annihilation step, there remains is a population with uniform sign on each determinant. The equations which describe the evolution of populations of different signs, and their annihilation are derived in detail in Ref. [18]. The annihilation is also the reason why the total population experiences a plateau, but we will discuss this later in Chapter 6.

3. Selection probability

The general formulas for the computation of the selection probability P_{C} are given in Ref. [19]. Here, we follow their approach for determining this quantity. The selection probability P_{C} needs to be computable, nonzero and normalized. When a double excitation $|D_j\rangle = a_r^\dagger a_s^\dagger a_q a_p |D_i\rangle$ is considered, an occupied pair (p, q) and an unoccupied pair (r, s) are selected. The selection probability $P_{\mathrm{C}}(j|i)$ can thus be written as:

$$P_{\mathrm{C}}(j|i) = P(r,s|p,q)P(p,q), \tag{4.16}$$

where $P(p,q)$ is the probability to select particles p and q, and $P(r,s|p,q)$ is the probability to select unoccupied spin-orbitals r and s, given that occupied spin-orbitals p and q have already been chosen.

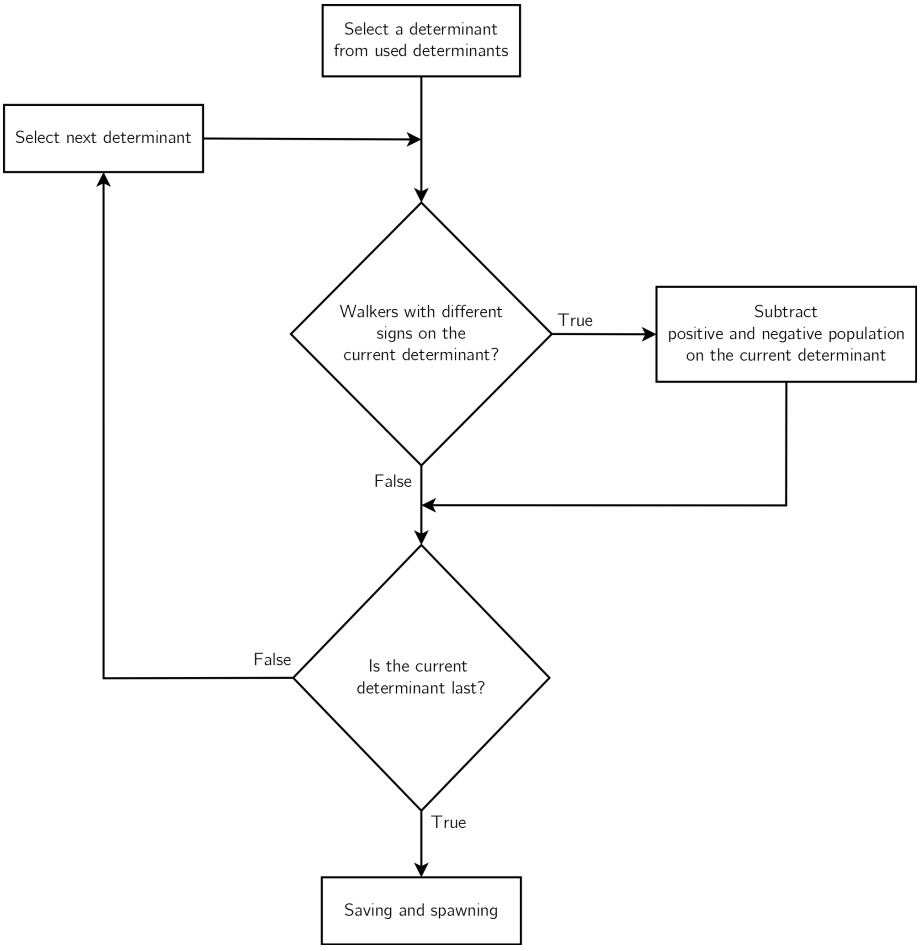


Figure 4.8: Diagram of the annihilation step

For a N_{el} electron system, the occupied pair is selected with uniform probability:

$$P(p,q) = \binom{N_{\mathrm{el}}}{2}^{-1} = \frac{2}{N_{\mathrm{el}}(N_{\mathrm{el}}-1)}, \tag{4.17}$$

and one needs to consider that r can be picked before s, or the other way round, because generally $P(r|s,p,q) \neq P(s|r,p,q)$:

$$P(r,s|p,q) = P(r|s,p,q) \cdot P(s|p,q) + P(s|r,p,q) \cdot P(r|p,q). \tag{4.18}$$

We thus need to determine the probability $P(s|p,q)$ of selecting the unoccupied spin-orbital s, and the probability $P(r|s,p,q)$ to select spin-orbital r, which is model dependent.

The generation of a single excitation $|D_j\rangle = a_q^\dagger a_p |D_i\rangle$ is simpler:

$$P_{\mathrm{C}}(j|i) = P(q|p) \cdot P(p), \tag{4.19}$$

with $P(p) = \frac{1}{N_{\mathrm{el}}}$.

4. The energy shift parameter S

In the FCIQMC algorithm, a second independent way, how to compute the ground state energy of the system, exists. In Eq. (4.9) the energy shift variable S was introduced. It was introduced for the diagonal terms, and we have shown in Eq. (4.13) that it affects the death/cloning step. If the shift S is negative (positive), then more (less) walkers die. The shift is a stochastic variable itself. It is changed until the number of new walkers and the number of dead walkers is the same. This results in the plateau in the graph of the population as a function of time. When the plateau is reached, the shift is equal to the ground state energy of the system. This is easily demonstrated in the eigenvector basis. We recall an Eq. (3.15) at first. When the energy shift is introduced into diagonal terms, and it converges to the value of E_{GS}, the exponent becomes zero and the population stops growing with time:

$$\lim_{\tau\to\infty} \sum_i c_i |\Phi_i\rangle e^{-\tau(E_i - S)} = \lim_{\tau\to\infty} \sum_i c_i |\Phi_i\rangle e^0 = \sum_i c_i |\Phi_i\rangle . \tag{4.20}$$

The shift starts to vary once a certain target level of population is reached. The shift also increases or decreases the speed of growth of the population according to the initial value it is set to.

The shift variable is changed every A steps, according to Eq. (4.21), originally from Ref. [20]:

$$S(\tau) = S(\tau - A\cdot\delta\tau) - \frac{\zeta}{A\cdot\delta\tau} \ln \frac{N_{\mathrm{w}}(\tau)}{N_{\mathrm{w}}(\tau - A\cdot\delta\tau)}, \tag{4.21}$$

where $N_{\mathrm{w}}(\tau)$ is the number of walkers at time τ and ζ is the damping parameter which prevents the shift S from fluctuating too wildly.

5. Obtaining results

When the simulation is finished, it is possible to evaluate the following quantities:

- The ground state wavefunction.
- The ground state energy - computed using the projection approach.
- The ground state energy - computed using the shift variable S.

The ground state wavefunction is obtained by converting the populations on each determinant back into a CI coefficient. For computing the ground state energy, a projection can be used:

$$\begin{aligned} E(\tau) &= \frac{\langle D_0|He^{-\tau H}|D_0\rangle}{\langle D_0|e^{-\tau H}|D_0\rangle} \\ &= \sum_j \langle D_j|H|D_0\rangle \frac{C_j}{C_0} \\ &= \sum_j \langle D_j|H|D_0\rangle \frac{N_j}{N_0}. \end{aligned} \tag{4.22}$$

The shift is obtained as the converged value of $S(\tau)$ as described in the preceding section.

It is useful to estimate the error associated to the obtained value of the ground state energy. It is possible to proceed in two different ways:

- Compute the energy for the last x timesteps, average it and calculate the standard deviation.
- Run more simulations with same parameters but a different random seed and average results from them.

The first way is not very computationally demanding. When the system converges to the ground state there is no problem with computing additional timesteps. In addition, the more walkers we have, the lower the error. On the other side, the data from a single simulation are correlated: two consecutive timesteps do not differ much. This problem can be avoided by averaging only every y-th value or using more complex estimates like jackknife.

The second way is computationally demanding as it is needed to run more simulations. The advantage is that the data from different simulations are not correlated. Ideally, more simulations are run and in every simulation many additional timesteps are computed after the system converges to the ground state and a high number of walkers is reached. Then the average energy for every simulation is computed and afterwards these results are averaged together. This way the stochastic error is minimized.

6. Complex amplitudes

Up to this point, all quantities have been assumed real. If we allow the CI coefficients to be complex, Ref. [21], they have to be represented by two populations of walkers instead of one (one population for the real part, another for the imaginary part). The walkers acquire a new attribute, indicating their nature, real or imaginary. The steps of the algorithm also need to be modified. Additionally, a new rotation step has to be added.

6.1. The spawning step

In contrary to the spawning step in the algorithm for real CI coefficients, two spawn attempts have to be realized for each walker.

For real parent walkers, the probability of spawning a real child walker $P_{\mathrm{S}}^{\mathcal{R}\to\mathcal{R}}$ and the probability of spawning an imaginary child walker $P_{\mathrm{S}}^{\mathcal{R}\to\mathcal{I}}$ are computed with corresponding signs:

$$\begin{aligned} P_{\mathrm{S}}^{\mathcal{R}\to\mathcal{R}} &= \frac{\delta\tau|\operatorname{Re}[H_{ij}]|}{P_{\mathrm{C}}} \qquad & \sigma^{\mathcal{R}\to\mathcal{R}} &= -\operatorname{sign}(\operatorname{Re}[N_i]\operatorname{Re}[H_{ij}]), \\ P_{\mathrm{S}}^{\mathcal{R}\to\mathcal{I}} &= \frac{\delta\tau|\operatorname{Im}[H_{ij}]|}{P_{\mathrm{C}}} \qquad & \sigma^{\mathcal{R}\to\mathcal{I}} &= -\operatorname{sign}(\operatorname{Re}[N_i]\operatorname{Im}[H_{ij}]). \end{aligned} \tag{4.23}$$

For imaginary parent walkers probability $P_{\mathrm{S}}^{\mathcal{I}\to\mathcal{R}}$ to spawn real walker and probability $P_{\mathrm{S}}^{\mathcal{I}\to\mathcal{I}}$ to spawn imaginary walker are computed as well:

$$\begin{aligned} P_{\mathrm{S}}^{\mathcal{I}\to\mathcal{R}} &= \frac{\delta\tau|\operatorname{Im}[H_{ij}]|}{P_{\mathrm{C}}} \qquad & \sigma^{\mathcal{I}\to\mathcal{R}} &= +\operatorname{sign}(\operatorname{Im}[N_i]\operatorname{Im}[H_{ij}]), \\ P_{\mathrm{S}}^{\mathcal{I}\to\mathcal{I}} &= \frac{\delta\tau|\operatorname{Re}[H_{ij}]|}{P_{\mathrm{C}}} \qquad & \sigma^{\mathcal{I}\to\mathcal{I}} &= -\operatorname{sign}(\operatorname{Im}[N_i]\operatorname{Re}[H_{ij}]). \end{aligned} \tag{4.24}$$

Probabilities are then handled in the same way as Sec. 4.2.1. describes and an according number of walkers is spawned.

Special care is needed when spawning real walkers from imaginary walkers. Spawning is governed by Eq. (4.8). When real walkers are spawned from imaginary walkers both H_{ij} and N_j are imaginary and multiplying two pure imaginary numbers yields a factor -1. This means that the signs of the new real walkers spawned from imaginary parent walkers are given by a different formula.

6.2. The death/cloning step

There are no differences in the death/cloning step in comparison with the algorithm for real CI coefficients. This step depends only on the diagonal matrix elements (4.13), which are real for a Hermitian matrix. Each walker is considered separately, regardless of its real or imaginary nature, and the death probability is computed by the Eq. (4.14).

6.3. The annihilation step

The annihilation step is similar for complex and real wavefunctions as well. The only difference is that there are two populations on each determinant and they have to be considered separately in this step.

6.4. The rotation step

The number of imaginary walkers on a given determinant in the complex version of algorithm is related to the phase of the CI coefficient of this determinant in the wavefunction. The estimator of the energy is based on the evaluation of a matrix element between a candidate ground state $|D_0\rangle$, and the state represented by the active population of the FCIQMC algorithm. The phase of $|D_0\rangle$ is chosen arbitrarily at the start of the run, and is not connected with the phase of the ground state which emerges from the FCIQMC procedure. This means that the estimator of the energy can easily possess a finite, unphysical, imaginary part, if the phases $|D_0\rangle$ and the phase of the current population in the FCIQMC calculation are mismatched. This can be fixed by modifying the phases of the populations on each determinant, so that the active population on $|D_0\rangle$ is purely real. This is equivalent to multiplying the CI coefficients by a complex number, and is executed via an additional step, added right after the annihilation step: the rotation step. The goal of this step is to ensure that the population of walkers on $|D_0\rangle$ is purely real, so that the main contribution to the energy estimator is also purely real.

A diagram of the rotation step is presented in Fig. 4.9. For applying a phase the magnitude and the angle of the starting determinant need to be computed, and then these quantities are calculated for all other determinants. Both quantities are shown in Fig. 4.10 and are computed as follows:

$$\begin{aligned} M_0 &= |N_0| = \sqrt{\mathrm{Re}[N_0]^2 + \mathrm{Im}[N_0]^2}, \\ \theta_0 &= \arctan\left(\frac{\mathrm{Im}[N_0]}{\mathrm{Re}[N_0]}\right). \end{aligned} \tag{4.25}$$

The rotation of walkers on the starting determinant is simple. This coefficient is needed to be pure real, so it is simply set to the number of walkers $\lfloor M_0 \rfloor$ here. One additional walker is then spawned with probability $M_0 - \lfloor M_0 \rfloor$.

In order to keep the wavefunction consistent, the same phase θ_0 needs to be applied on all CI coefficients. The magnitude and phase are calculated for the other walkers:

$$\begin{aligned} M_i &= |N_i| = \sqrt{\mathrm{Re}[N_i]^2 + \mathrm{Im}[N_i]^2}, \\ \theta_i &= \arctan\left(\frac{\mathrm{Im}[N_i]}{\mathrm{Re}[N_i]}\right). \end{aligned} \tag{4.26}$$

Afterwards a new angle is computed after applying the phase shift:

$$\theta_i' = \theta_i - \theta_0. \tag{4.27}$$

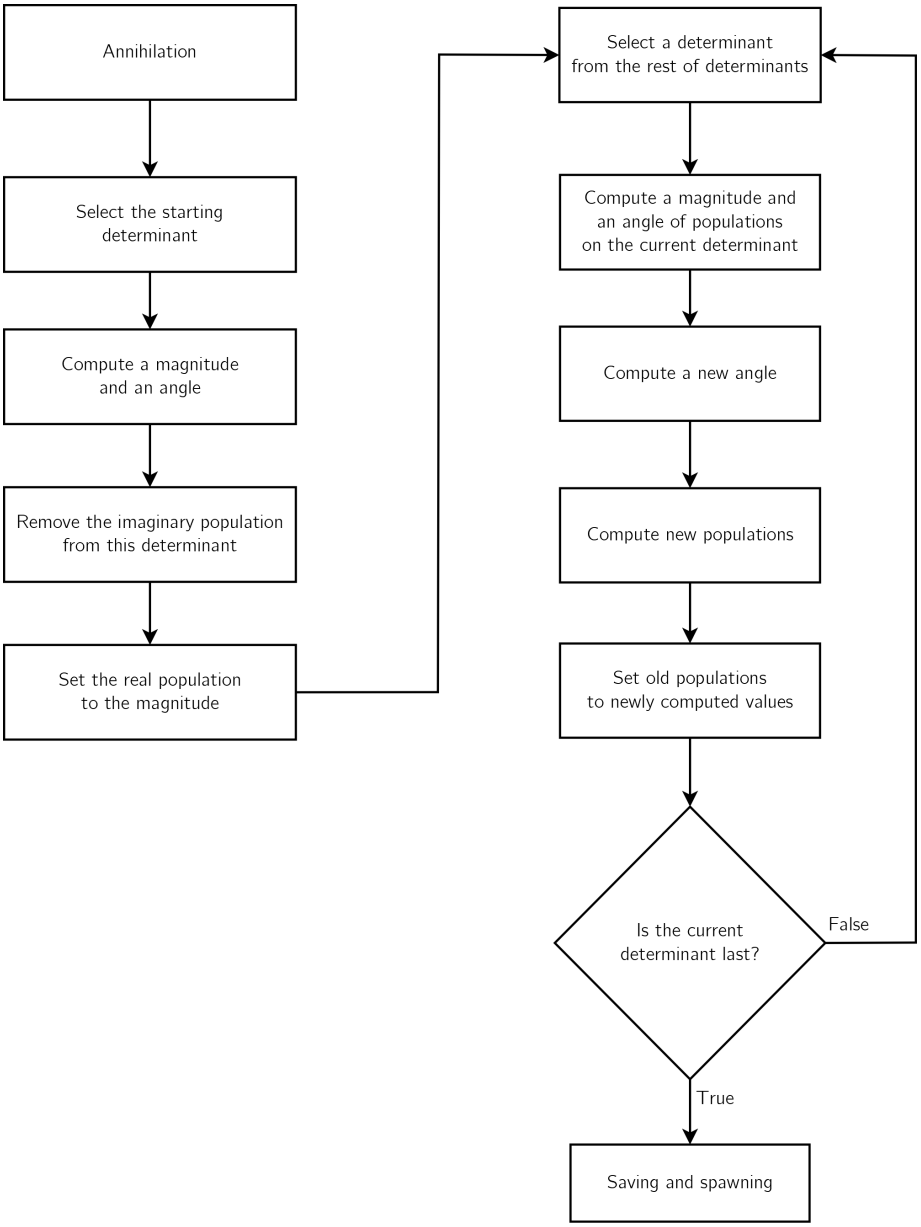


Figure 4.9: Diagram of the rotation step

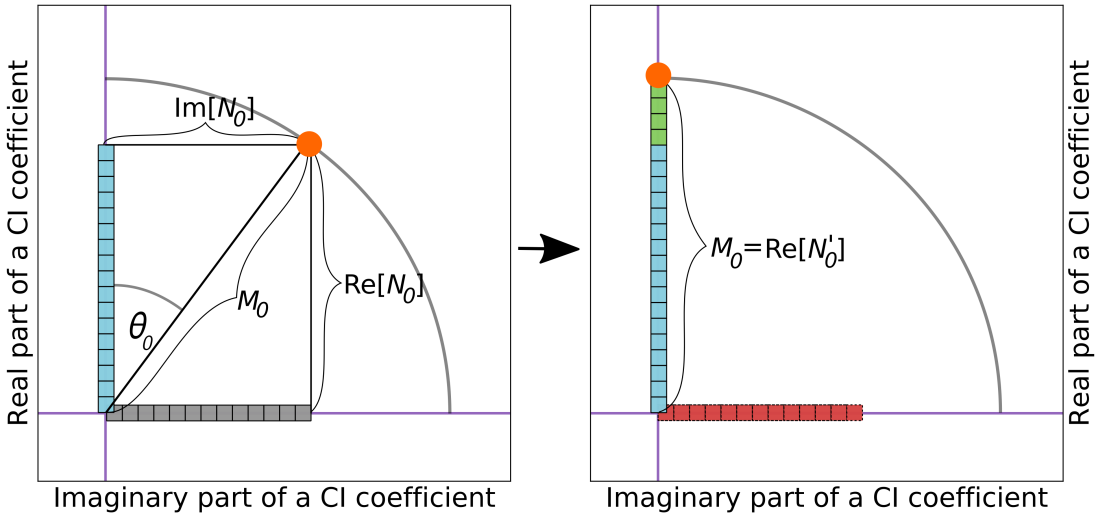


Figure 4.10: Rotation of the starting state - Real (blue) and imaginary (grey) populations living on the single determinant are plotted on the left side. Variables which can be calculated from these populations are in the figure as well. Populations after the rotation step are on the right side. Walkers which die during the rotation step have red color. Walkers which are added to the simulation during the rotation step are green.

Finally, the new size of the population is obtained with Eq. (4.28):

$$\begin{aligned} \mathrm{Re}[N_i'] &= M_i \cos(\theta_i') \qquad \text{Real walkers,} \\ \mathrm{Im}[N_i'] &= M_i \sin(\theta_i') \qquad \text{Imaginary walkers.} \end{aligned} \tag{4.28}$$

After computing the new populations, the old populations do not need to be stored anymore. As numbers computed by (4.28) are real rather than integer, we spawn $\lfloor \mathrm{Re}[N_i'] \rfloor$ new real walkers, and one real walker with probability $\mathrm{Re}[N_i'] - \lfloor \mathrm{Re}[N_i'] \rfloor$. Similarly we spawn $\lfloor \mathrm{Im}[N_i'] \rfloor$ new imaginary walkers and one with probability $\mathrm{Im}[N_i'] - \lfloor \mathrm{Im}[N_i'] \rfloor$. The rotation of a determinant with all variables is shown in the Fig. 4.11.

6.5. Energy evaluation

The energy of the system is given by Eq. (4.22), considering all variables including the energy are now complex. The energy needs to be real in order to be physically meaningful. Thus, the complex value of the energy is split into its real and imaginary components *real energy* and *imaginary energy*. The contribution of the populations

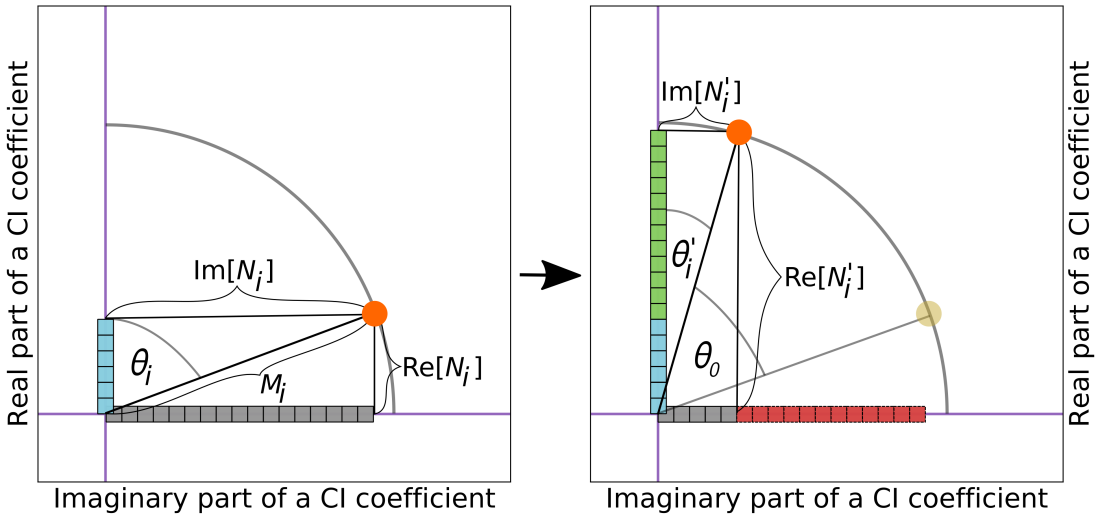


Figure 4.11: Rotation step - Real (blue) and imaginary (grey) populations living on the single determinant are plotted on the left side. On the right side populations after the rotation step are plotted. Walkers which die during the rotation step have red color. Walkers which are added to the simulation during the rotation step are green. Variables which can be calculated from these populations are in the figure as well. Orange points mark CI coefficient. The light brown point marks CI coefficient before the rotation step.

on the determinant $|D_i\rangle$ to the total energy is derived in the following:

$$
\begin{aligned}
\langle D_0|\hat{H}|D_i\rangle \frac{N_i}{N_0} &= (\mathrm{Re}[H_i] + i\,\mathrm{Im}[H_i]) \cdot \frac{\mathrm{Re}[N_i] + i\,\mathrm{Im}[N_i]}{\mathrm{Re}[N_0] + i\,\mathrm{Im}[N_0]} \\
&= (\mathrm{Re}[H_i] + i\,\mathrm{Im}[H_i]) \cdot (\mathrm{Re}[N_i] + i\,\mathrm{Im}[N_i]) \cdot \frac{\mathrm{Re}[N_0] - i\,\mathrm{Im}[N_0]}{\mathrm{Re}[N_0]^2 + \mathrm{Im}[N_0]^2} \\
&= ((\mathrm{Re}[H_i]\,\mathrm{Re}[N_i]\,\mathrm{Re}[N_0] - \mathrm{Im}[H_i]\,\mathrm{Im}[N_i]\,\mathrm{Re}[N_0] \\
&\quad + \mathrm{Re}[H_i]\,\mathrm{Im}[N_i]\,\mathrm{Im}[N_0] + \mathrm{Im}[H_i]\,\mathrm{Re}[N_i]\,\mathrm{Im}[N_0]) \\
&\quad + i\,(\mathrm{Re}[H_i]\,\mathrm{Im}[N_i]\,\mathrm{Re}[N_0] + \mathrm{Im}[H_i]\,\mathrm{Re}[N_i]\,\mathrm{Re}[N_0] \\
&\quad - \mathrm{Re}[H_i]\,\mathrm{Re}[N_i]\,\mathrm{Im}[N_0] + \mathrm{Im}[H_i]\,\mathrm{Im}[N_i]\,\mathrm{Im}[N_0])) \\
&\quad / \left(\mathrm{Re}[N_0]^2 + \mathrm{Im}[N_0]^2\right) \\
&= \frac{R_i + I_i}{\mathrm{Re}[N_0]^2 + \mathrm{Im}[N_0]^2}, \\
R_i &= (\mathrm{Re}[H_i]\,\mathrm{Re}[N_i]\,\mathrm{Re}[N_0] - \mathrm{Im}[H_i]\,\mathrm{Im}[N_i]\,\mathrm{Re}[N_0] \\
&\quad + \mathrm{Re}[H_i]\,\mathrm{Im}[N_i]\,\mathrm{Im}[N_0] + \mathrm{Im}[H_i]\,\mathrm{Re}[N_i]\,\mathrm{Im}[N_0]) \\
&\quad / \left(\mathrm{Re}[N_0]^2 + \mathrm{Im}[N_0]^2\right), \\
I_i &= i\,(\mathrm{Re}[H_i]\,\mathrm{Im}[N_i]\,\mathrm{Re}[N_0] + \mathrm{Im}[H_i]\,\mathrm{Re}[N_i]\,\mathrm{Re}[N_0] \\
&\quad - \mathrm{Re}[H_i]\,\mathrm{Re}[N_i]\,\mathrm{Im}[N_0] + \mathrm{Im}[H_i]\,\mathrm{Im}[N_i]\,\mathrm{Im}[N_0]) \\
&\quad / \left(\mathrm{Re}[N_0]^2 + \mathrm{Im}[N_0]^2\right).
\end{aligned}
\tag{4.29}
$$

The real and imaginary components of the energy are obtained as:

$$\begin{aligned} \mathrm{Re}[E] = E_{\mathrm{R}} &= \frac{\sum_i R_i}{\mathrm{Re}[N_0]^2 + \mathrm{Im}[N_0]^2}, \\ \mathrm{Im}[E] = E_{\mathrm{I}} &= \frac{\sum_i I_i}{\mathrm{Re}[N_0]^2 + \mathrm{Im}[N_0]^2}. \end{aligned} \tag{4.30}$$

The imaginary component of the energy is important for monitoring the convergence of a simulation, as its convergence to zero indicates that the relevant physical final state is within reach.

7. Further references

An introduction to FCIQMC can be found in Ref. [22] and Ref. [23]. The method was first introduced in 2009 in Ref. [19], an in 2010 [24] it was found that adding an additional condition to the algorithm reduces the number of walkers needed for convergence, thus reducing the overall computational effort. In this improvement, all determinants are split into initiator and noninitiator determinants. Initiator determinants are those with a population of walkers higher than a given threshold. Walkers on noninitiator determinants can spawn new walkers only on determinants which already have nonzero population: only "initiator" determinants can actually create a new population from zero. This mechanism improves the efficiency of the algorithm. The initiator approximation is further studied in Ref. [25].

In Ref. [18], the fermion sign problem is analyzed. Differential equations that describe the growth of populations of different signs are derived and the importance of the annihilation step is emphasized. The population plateau is the result of derived equations. One of results is that the height of the plateau depends on the intensity of coupling and this is more studied in Ref. [26].

In terms of implementations, only one open source code is currently available for this algorithm: the NECI code [27] is written in Fortran. It can be run, however, from the Python interface, using the library PySCF [28].

Model quantum dots are studied using FCI and FCIQMC in Refs. [29, 30].

Chapter 5

Models

FCIQMC is a general algorithm that can be used to handle various models. This chapter describes shortly the models which are currently implemented in our code with an emphasis on the computation of the model-specific selection probability P_{C} previously described in Sec. 4.3.. This is the most subtle part of the implementation of the FCIQMC algorithm.

1. Electron gas

Our study of interacting systems has focused on the electron gas [31–33]. In order to be able to address this problem, we have based our modeling on the work of Shepherd [33], which we have adapted to a solid state physics framework. Their approach is based on a set of electrons enclosed in a simulation-cell, and they use an energy cutoff in order to make the basis set finite. Our description of the electron gas is based on a more usual approach in solid state physics, that of a solid made up of a finite number of unit cells of parameter a. The system contains a number of electrons N_{el}. In this work, we consider a simple cubic unit cell, containing a single orbital at each vertex. The full system contains N_{k}^3 such unit cells, i.e. its length in each direction is $a \cdot N_{\mathrm{k}} \equiv L$. Periodic boundary conditions are applied. In the following, we take $a = 1$ without loss of generality. N_{k} is assumed even for numerical purposes (so that the Γ point is reachable in the simulation). The Hamiltonian for this system can be written [33]

$$\hat{H} = \sum_{i}^{N_{\mathrm{el}}} \frac{\hat{p}_i^2}{2m_{\mathrm{el}}} + \frac{1}{2} \sum_{i}^{N_{\mathrm{el}}} \sum_{j \neq i}^{N_{\mathrm{el}}} \frac{e^2}{|\hat{r}_i - \hat{r}_j|} \tag{5.1}$$

We look for the ground state using Slater determinants based on plane waves indexed by momentum and spin:

$$\psi_j(\mathbf{x}) = \psi_j(\mathbf{r}, \sigma) = \sqrt{\frac{1}{\Omega}} e^{i\mathbf{k}_j \cdot \mathbf{r}} \delta_{\sigma_j, \sigma}, \tag{5.2}$$

where Ω is the unit cell volume. The allowed values for the wavevectors $\mathbf{k}_j$ are determined by the total size of the system:

$$\mathbf{k}_j = \frac{2\pi}{L}(n, m, l), \tag{5.3}$$

with (n, m, l) integers. We define the first Brillouin zone as the volume defined by $-N_{\mathrm{k}}/2 < (n, m, l) \leq N_{\mathrm{k}}/2$. It is illustrated in Fig. 5.1. It should be noted that

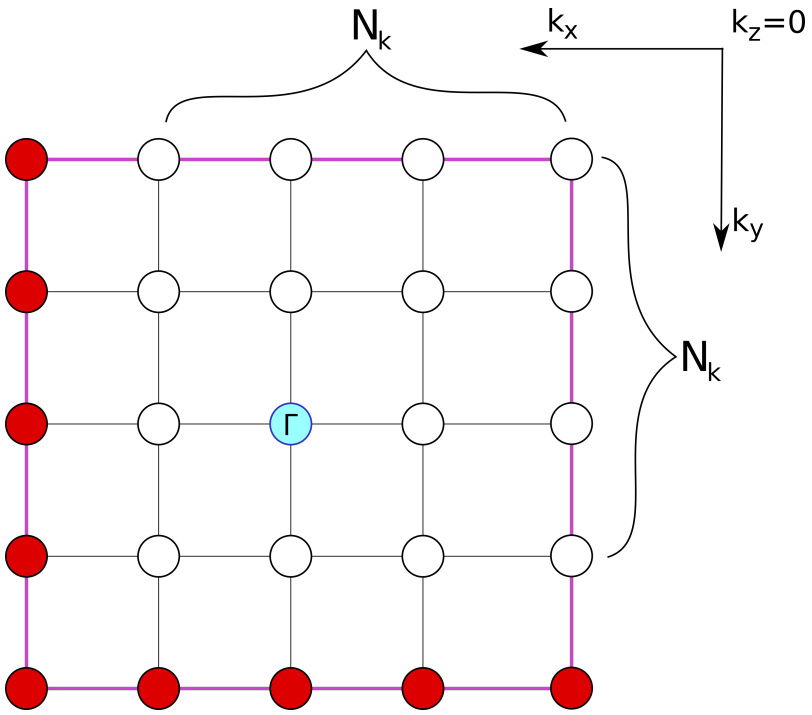


Figure 5.1: A cut through the first Brillouin zone of the electron gas model, plane $k_z = 0$. The number of lattice points per a single axis, N_k, is chosen even so that the Γ-point is accessible (blue circle). The red circles lying on edges of the allowed domain are not considered as they are equivalent to other points in the allowed domain by a reciprocal in order to avoid double counting (they are equivalent of first Brillouin zone (violet line) but belonging to the second Brillouin zone.

this scheme differs slightly from what is usually employed in solid state physics. It is indeed customary to work with a fixed number of electrons per unit cell (or equivalently, with a fixed density of electrons n), in order to describe a solid. FCIQMC is a method which simulates the full system, and in which the number of simulated particles is a critical parameter. Therefore, our description of the system relies not on its density, but on the total number of electrons $N_\mathrm{el} = n \cdot N_\mathrm{k}^3$. One must keep in mind that as a consequence, any study which would consider the thermodynamic limit $N_\mathrm{k} \to \infty$ for fixed density must impose $N_\mathrm{el} \to \infty$ as well. This is one of the majors challenges in developing a FCIQMC engine: the study of the thermodynamic limit requires that a very large value of N_el be supported by the model.

The second quantized form of the Hamiltonian is:

$$\hat{H} = \sum_{ij} t_i^j a_i^\dagger a_j + \frac{1}{2} \sum_{ijkl} v_{ij}^{kl} a_i^\dagger a_j^\dagger a_l a_k, \tag{5.4}$$

where t_i^j is the kinetic energy operator,

$$t_i^j = -\frac{1}{2} \langle i | \nabla^2 | j \rangle = \frac{1}{2} \mathbf{k}_i^2 \delta_{ij}, \tag{5.5}$$

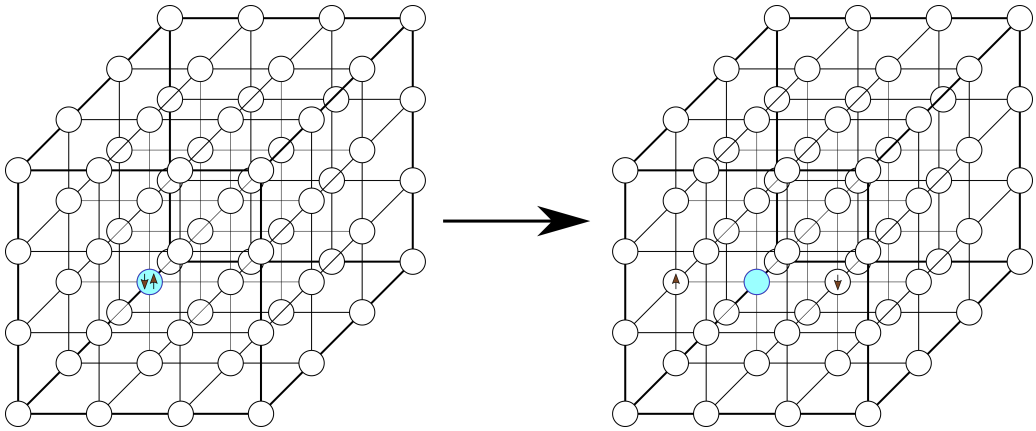


Figure 5.2: Representation of the Hartree-Fock determinant (left) with two electrons (brown arrows) at the centre of the reciprocal space (the Γ-point is highlighted in blue). A doubly excited determinant is shown on the right: the two electrons are sitting on different single particle wavefunctions, with total momentum conserved.

and v_{ij}^{ab} is a two-particle operator which represents the electron-electron interaction:

$$\begin{aligned} v_{ij}^{ab} &= v_{\mathbf{g}} \delta_{\mathbf{g},\mathbf{k}_a - \mathbf{k}_i} \delta_{\mathbf{g},\mathbf{k}_j - \mathbf{k}_b}, \\ v_{\mathbf{g}} &= \begin{cases} \frac{1}{\Omega} \frac{4\pi}{\mathbf{g}^2} & \mathbf{g} \neq 0 \\ 0 & \mathbf{g} = 0, \end{cases} \end{aligned} \tag{5.6}$$

where $\mathbf{g}$ is the change of the momentum due to an excitation.

For the spawning step only nondiagonal matrix elements $\langle D_i | \hat{H} | D_{j \neq i} \rangle$ need to be considered. In this model, the kinetic energy operator is diagonal, so it does not participate in the spawning. It is the operator v_{ij}^{ab} which couples determinants. To be nonzero, this operator needs two interacting electrons, it causes double excitations. As there are no other operators in this model, determinants are always doubly excited: they differ by two spin-orbitals, and there are no single excitations.

In order to generate the excitations, we use the conservation of total momentum. First, this implies that any excited determinant with a non-zero population has the same total momentum as the starting determinant. Second, when generating a double excitation, we pick one electron with momentum k_1, and transfer it to another single particle state with moment k_1'. This implies that the second electron is transferred from a state k_2 to a state k_2', such that $k_1 + k_2 = k_1' + k_2'$. This considerably restricts the amount of possibilities we need to consider: translational symmetry helps us in this case. An example of a determinant and an excitation is given in the Fig. 5.2.

A relevant parameter in the study of interacting systems is the intensity of the coupling (often referred to as the ratio U/t in models such as the Hubbard model). In the electron gas model, the corresponding ratio can be introduced, if we introduce scaling for the kinetic and potential energy as follows:

$$\begin{aligned} t_i^j &= -\frac{T}{2} \langle i | \nabla^2 | j \rangle = \frac{T}{2} \mathbf{k}_i^2 \delta_{ij}, \\ v_{ij}^{ab} &= U \cdot v_{\mathbf{g}} \delta_{\mathbf{g},\mathbf{k}_a - \mathbf{k}_i} \delta_{\mathbf{g},\mathbf{k}_j - \mathbf{k}_b}. \end{aligned} \tag{5.7}$$

1.1. Selection probability

The calculation of the selection probability for this model is a complex task. One needs to determine, which electrons to process, and which empty spin-orbitals such electrons will be moved to. For both decisions, there is a selection probability which is a part of the total selection probability. We label P_1 the selection probability of electrons and P_2 the selection probability of spin-orbitals.

At first, one of N_{el} electrons is chosen and labeled p. Any excitation in this model is the double excitation, so a second electron is selected from $N_{\mathrm{el}} - 1$ electrons and labeled q. These two electrons can be selected in two ways. Electron p can be chosen at first and then q or vice versa. Because of this, the selection probability has to be multiplied by a factor of 2. The selection probability of electrons is:

$$P_1(j|i) = P(p,q) = \frac{2}{N_{\mathrm{el}} \cdot (N_{\mathrm{el}} - 1)}. \tag{5.8}$$

Once this is done, two empty spin-orbitals need to be picked. The allowed k-domain is a cubic grid with a N_{k} allowed positions in each direction (N_{k} is assumed to be even for simplicity). Given our convention for the Brillouin zone, and considering that the particle q needs to compensate any change in the momentum of particle p, the change in momentum for particle p along the x direction is constrained as follows:

$$\begin{aligned} \delta x^+ &= \min(-p_x - 1, q_x) + \frac{N_{\mathrm{k}}}{2}, \\ \delta x^- &= \min(-q_x - 1, p_x) + \frac{N_{\mathrm{k}}}{2}, \end{aligned} \tag{5.9}$$

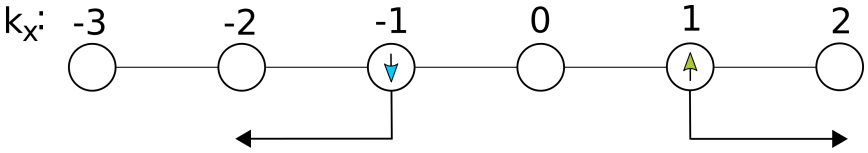
where δx^+ is the maximal change of the momentum of the electron p in the positive direction along the x-axis so that electron q can balance the change of momentum and move by the same amount in the opposite direction. p_x and q_x are the x-coordinates of electrons p and q. The computation of both variables is represented in Fig. 5.3.

δy^+, δy^-, δz^+, δz^- are determined similarly. This allows the evaluation of the volume of k-space into which particle p may move. For two electrons, this volume is given by:

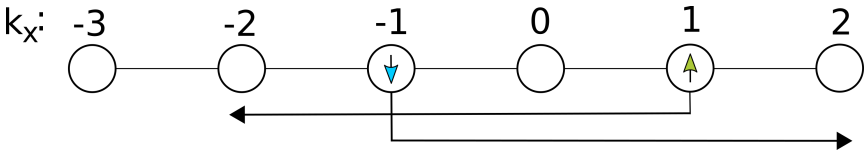
$$N_A = (1 + \delta x^+ + \delta x^-)(1 + \delta y^+ + \delta y^-)(1 + \delta z^+ + \delta z^-). \tag{5.10}$$

The minimal volume is 1, because the p electron always has the possibility to remain on the starting spot.

In this volume, some spin-orbitals may actually be forbidden (their number is noted N_{F}), because a move there would not respect the Pauli principle. This is illustrated in Fig. 5.4: In Subfig. 5.4a, there is a forbidden orbital at the top which is fully occupied. This means that the other orbital pictured in red at the bottom is forbidden too, even though it is empty. If any of the selected electrons were to be transferred to the forbidden orbital at the bottom, the second electron would have to move to the fully occupied orbital at the top in order to conserve total momentum. In Subfig. 5.4b, the central orbital can not be selected. A single electron can move here, but the other electron would have to balance the momentum change and move



(a) Example of computation δx^+ - electron p (green arrow) can move only by a single lattice point in the positive direction of x-axis ($\delta x^+ = 1$), because there are no more points in the first Brillouin zone in this direction.



(b) Example of computation δx^- - electron p (green arrow) can move maximally by three lattice points in the negative direction of x-axis ($\delta x^- = 3$). Electron q (blue arrow) has to move by the same amount in the opposite direction and if $\delta x^- > 3$, it would move out of the first Brillouin zone.

Figure 5.3: x-coordinate of two selected electrons p (green arrow) and q (blue arrow) are plotted into a single axis. $N_\mathrm{k} = 6$, so there are totally six orbitals (white circles) on the axis. In the Subfigs. 5.3a and 5.3b the computation of δx^+ and δx^- from Eq. (5.9) is graphically represented.

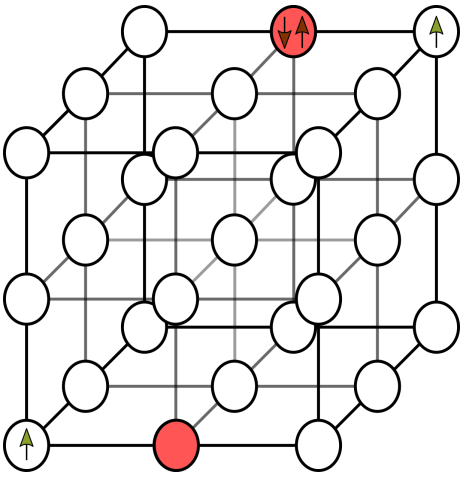
there too, however the orbital is already partially occupied and three electrons can not fit there.

From N_A allowed spin-orbitals we select an empty spin-orbital r. The other spin-orbital (s) does not require a selection, because of spin and momentum conservation. As we can choose r and then s or vice versa we gain again the factor of 2 in the second part of the selection probability:

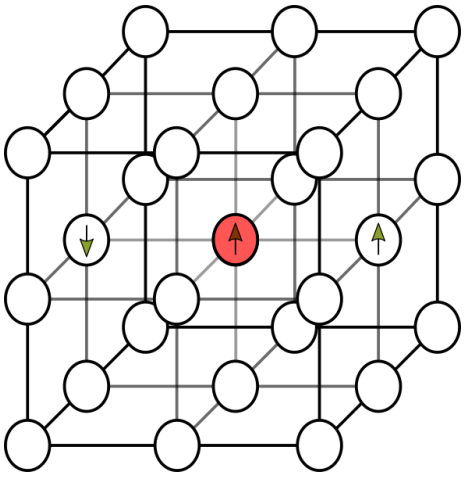
$$\begin{aligned} P_2(j|i) &= P(r,s|p,q) \\ &= \frac{2}{(1+\delta x^+ + \delta x^-)(1+\delta y^+ + \delta y^-)(1+\delta z^+ + \delta z^-) - N_\mathrm{F}} \\ &= \frac{2}{N_\mathrm{A} - N_\mathrm{F}}. \end{aligned} \tag{5.11}$$

The final selection probability is then:

$$\begin{aligned} P_C(j|i) &= P_1(j|i) P_2(j|i) \\ &= P(r,s|p,q) P(p,q) \\ &= \frac{4}{N_\mathrm{el} \cdot (N_\mathrm{el}-1)(N_\mathrm{A} - N_\mathrm{F})}. \end{aligned} \tag{5.12}$$



(a) Two electrons are selected (green arrows). Red orbitals are forbidden for selection as unoccupied. The orbital at the top of the figure is forbidden because it is fully occupied. The one at the bottom can not be selected even though it is empty, the second electron would have to move to the fully occupied orbital at the top in order to conserve the momentum.



(b) Two selected electrons are pictured as green arrows. The orbital in the middle (red) can not be selected. It contains a single electron (brown arrow) so it is partially occupied and both selected electrons would have to move here in order to conserve the momentum. Three electrons can not fit into a single orbital so it is forbidden.

Figure 5.4: Examples of forbidden orbitals

2. Matrix

The second model is a different, but complementary approach, in which we assume that only the matrix representing the Hamiltonian in some basis is provided as input (*Matrix* model). It is a simpler context for the FCIQMC algorithm, because all necessary information is readily available in memory.

This *Matrix* model was developed in order to compute the model of quantum dots where the Hamiltonian is not given in a second quantized form, only the Hamiltonian matrix is available. This is not, however, the sole purpose of this model, it can be used to test and verify results of existing models whose Hamiltonians were generated. The complex approach had to been implemented, because it is necessary for the computation of the model of quantum dots which use complex CI coefficients.

2.1. Selection probability

When a walker is given a chance to spawn another walker, all that is needed is to identify the row corresponding to initial walker's determinant, and determine the number of nonzero elements in this row $N_{\neq 0}$. We assume that each of these connected determinants is equally probably, so that the selection probability is given by

$$p_C(j|i) = \frac{1}{N_{\neq 0} - 1}. \tag{5.13}$$

The minus one comes from the fact that we do not wish to consider 'self-spawning' moves in the Monte Carlo steps.

Eq. (5.14) shows a simple example of this approach, with some random Hamiltonian matrix, and the corresponding selection probabilities to choose the state in column j as a target for spawning from a walker in the state associated with line i:

$$\begin{pmatrix} 0.3 & -0.1 & -0.2 & 0.0 \\ -0.1 & 0.4 & 0.3 & -0.4 \\ -0.2 & 0.3 & 0.2 & 0.0 \\ 0.0 & -0.4 & 0.0 & 0.1 \end{pmatrix} \rightarrow \begin{pmatrix} 0.0 & 1/2 & 1/2 & 0.0 \\ 1/3 & 0.0 & 1/3 & 1/3 \\ 1/2 & 1/2 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 & 0.0 \end{pmatrix} \tag{5.14}$$

Chapter 6

Results

1. Homogeneous electron gas

As a first system, we have studied the homogeneous electron gas, with two electrons and antiparallel spins. Parameters in Eq. (5.7) were set to $T = 16$ and $U = 1/16$ so the interaction of electrons was suppressed. The starting determinant $|D_0\rangle$ needs to be chosen carefully: if it does not have a substantial overlap with the real ground state of the system, then the algorithm cannot converge towards the correct result. In this case, we chose $|D_0\rangle$ as the state with two electrons with the lowest kinetic energy, i.e. located at the centre of the reciprocal space as pictured in Fig. 6.1. The centre of the reciprocal space has a total zero momentum and because of that the lowest kinetic energy of all possible determinants. We can expect the largest CI coefficient here, at least as long as the Coulomb interaction is not dominant, which should always be the case, considering that there are only two particles in the system. It is also the Hartree-Fock determinant, which is usually a good place to start.

When the lowest possible number of points in k-space is used, cube with four points per edge, the system can be described by only twenty seven determinants. This is very easily computable by exact diagonalization, which we used in order to check the correctness of all the FCIQMC results.

The computed ground state and its energy are on the graphs 6.2 and 6.3.

In Fig. 6.2, it is possible to see that the CI coefficient of the starting determinant is the highest, as expected. Next, there are six determinants with same value of CI coefficients. These are the states where each electron has a momentum equal to one. Their reciprocal representation is given in Fig. 6.4.

2. Matrix approach to the electron gas system

Our second model *Matrix* can handle any system whose Hamiltonian matrix is given. The system from the previous section uses only twenty seven determinants, so that its matrix is small enough to be easily computed and stored, and fed into the *Matrix* model. Our code is able to evaluate the matrix elements between all determinants of the system, using the Slater-Condon rules described in 2.3.2.. The technical limit to this approach is the size of the Hamiltonian matrix, which increases exponentially with the size of the considered system.

The results computed with the *Matrix* model should match exactly the results obtained using the *ElectronGas* approach, as the Hamiltonian matrix and the Hamiltonian expressed using second quantization describe the same system.

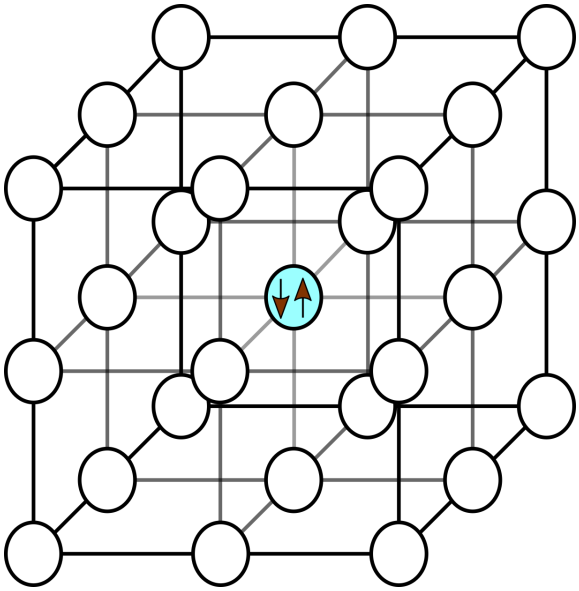


Figure 6.1: Starting state for the electron gas - two electrons (brown arrows) in the centre of reciprocal space (blue circle). In the figure is only a subset of the first Brillouin zone centered around the Γ-point.

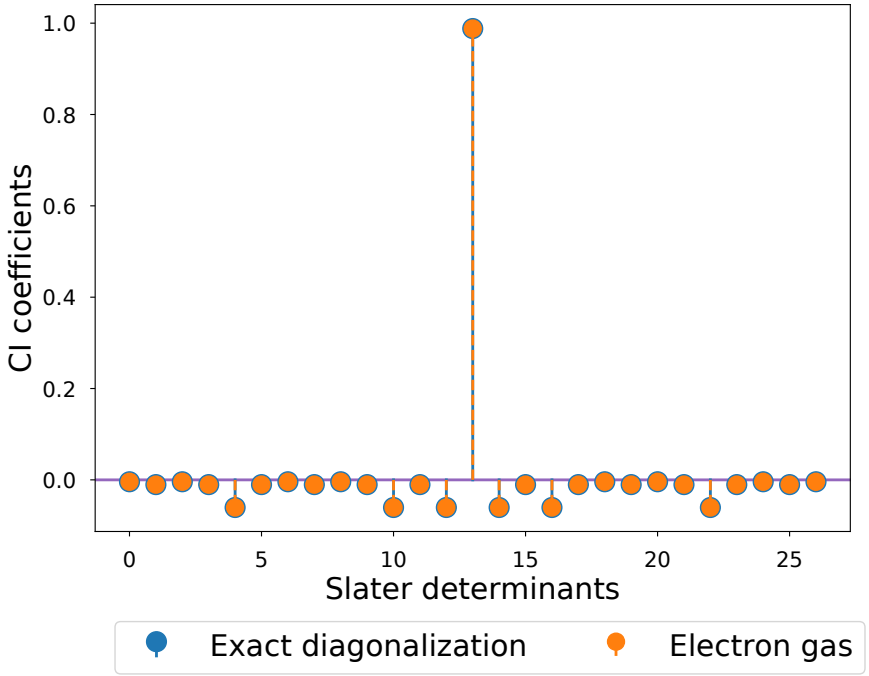


Figure 6.2: The ground state of the electron gas with suppressed interactions between electrons. x-axis is used to index Slater determinants, y-axis is used for their CI coefficients. Orange points computed by FCIQMC are plotted over blue points computed by exact diagonalization.

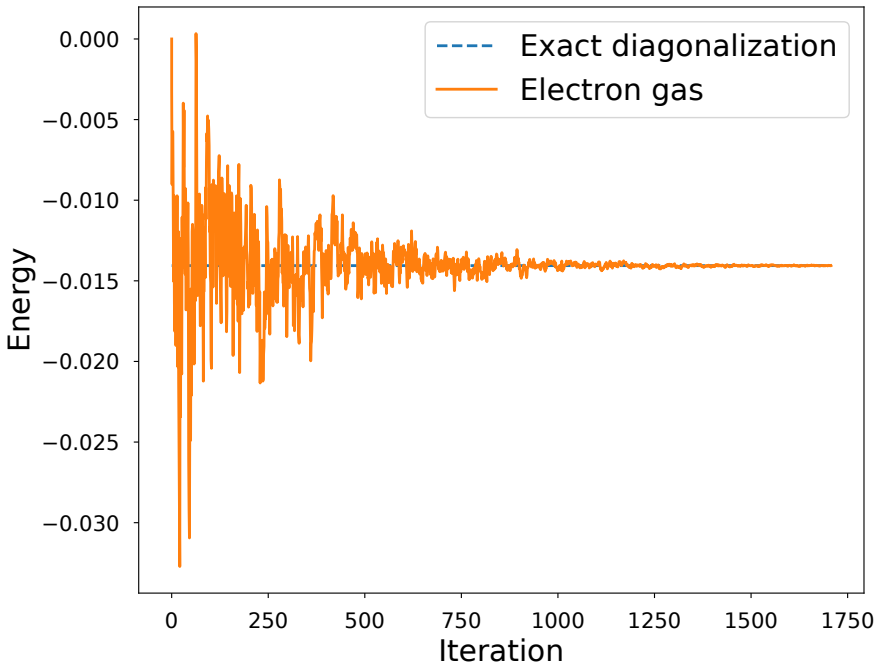


Figure 6.3: The energy of the electron gas calculated using FCIQMC (orange line) compared with the correct ground state energy computed using exact diagonalization (blue dashed line).

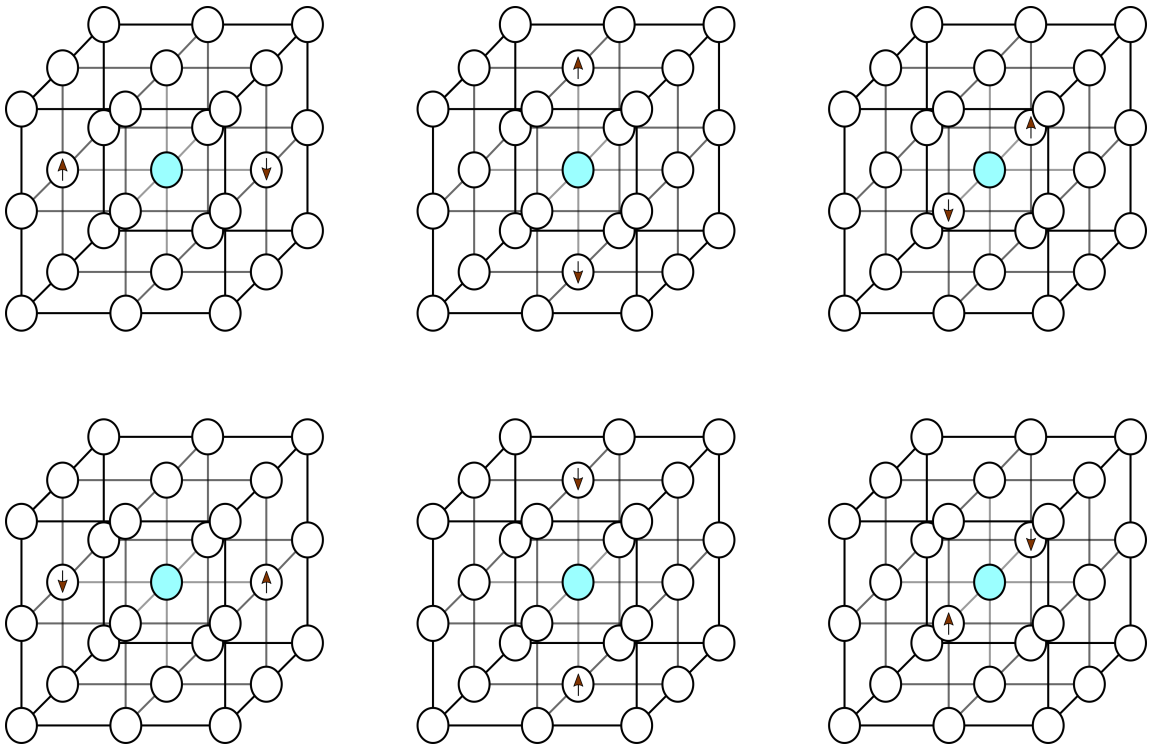


Figure 6.4: Determinants with second highest absolute value of CI coefficients in Fig. 6.2 - every cube is only a subset of the first Brillouin zone centered around the Γ-point (blue circle).

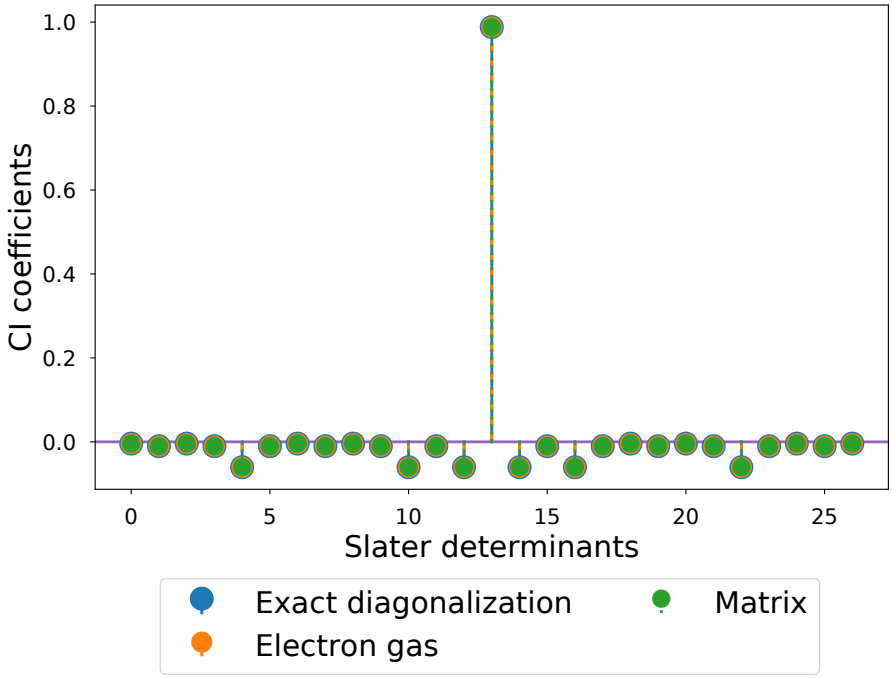


Figure 6.5: The same as in Fig. 6.2, but with new results (green points) computed by the *Matrix* model.

Once the matrix is generated, we can run a simulation using the *Matrix* model. The obtained results are then compared with those obtained with the *ElectronGas* model from the previous section.

The results are indistinguishable. Both approaches match the values provided by the exact diagonalization as well, as expected.

3. Number of lattice points

One of the parameters of any CI approach is the number of basis states used in the calculation. The higher the number of those states, the better the CI approximation. In our case, the number of single-particle basis states is controlled by the number N_k. It represents the number of unit cells in the direct space in a single direction (we assume, that it is uniform for all directions) and the number of lattice points in the reciprocal space in a single direction.

A system with identical parameters as in Sec. 6.1. but with $N_\mathrm{k} = 6$ was computed. Its ground state is in Fig. 6.7. It can not, however, be compared with Fig. 6.2 where $N_\mathrm{k} = 4$ in a meaningful way: as mentioned in Sec. 5.1., an increase in N_k, together with a fixed number of electrons N_{el}, generates a decrease in the density n of the system. Therefore, interactions effects, which are the main point of our study, are qualitatively modified by such a transformation. As soon as the code will support the sufficient number of electrons, it will be possible to keep the density $n = \frac{N_\mathrm{el}}{(N_\mathrm{k}a)^3}$ constant and to study finite size effects.

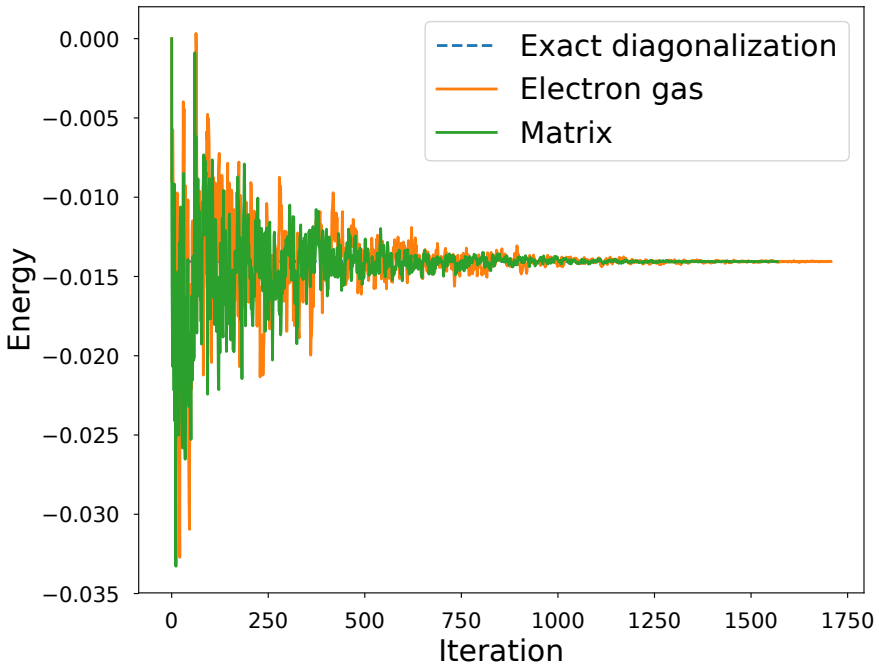


Figure 6.6: The same as in Fig. 6.2 with new result (green line) calculated using the *Matrix* model.

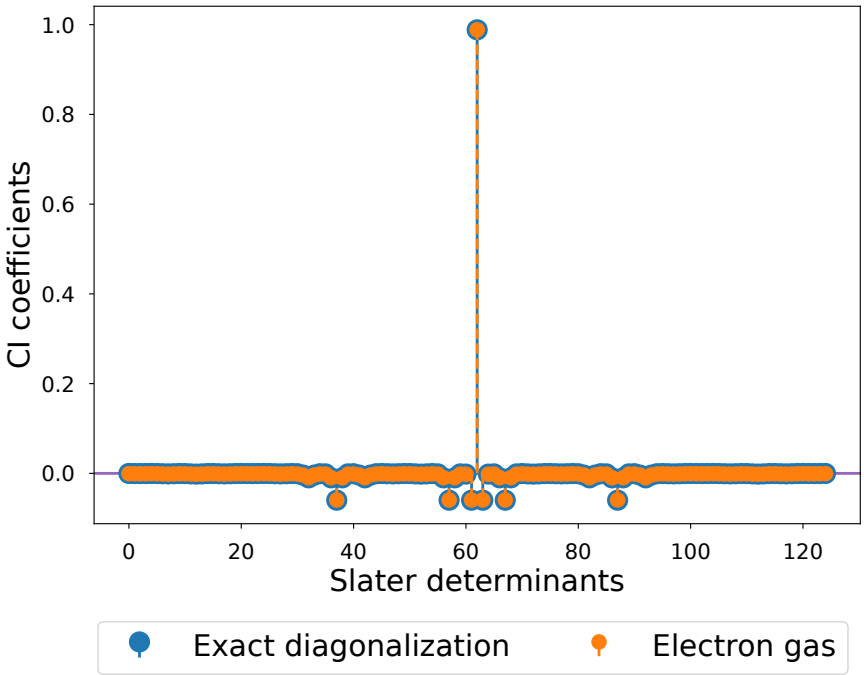


Figure 6.7: The ground state of weakly interacting electron gas as in Fig. 6.2 but with six unit cells in the direct space in every direction.

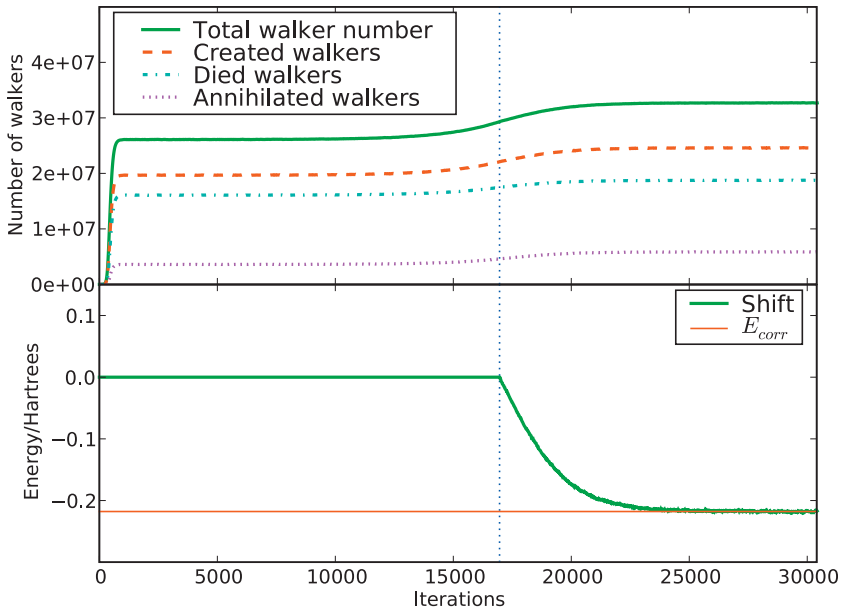


Figure 6.8: An example of results of FCIQMC from Ref. [19]; presented with permission from the author.

4. Population size: plateau

The population of walkers grows exponentially with imaginary time. In general, the population stops growing at some point, exhibiting a plateau. The value, and emergence of this plateau is system specific, with the height of the plateau being related to the computational difficulty of the problem. This phenomenon is illustrated in Fig. 6.8 taken from Ref. [19]:

In Fig. 6.8 we can see two plateaus. The first one, at lower values of imaginary time, is the one just described. The second plateau is created via the shift control variable, which will be examined in more detail in Section 6.5..

Now and in the rest of this section, we are interested in the first plateau, which is not caused by changing the energy shift. We can see that the shift is constant from the lower part of the figure. The algorithm needed high number of walkers to reach the first plateau, in total 26×10^6. On the other hand their system, the water molecule, is described by 451×10^6 determinants.

In Fig. 6.9, showing our results, we can not identify any plateau. This is due to the tiny size of the FCIQMC space in this case. The visibility of such plateau can depend on various factors [26], and we examine such possibilities in the next sections.

5. Shift parameter S

In this section we consider the behavior of the energy shift described in Chap. **??**. The energy shift is also an estimator of the ground state energy of the system. In

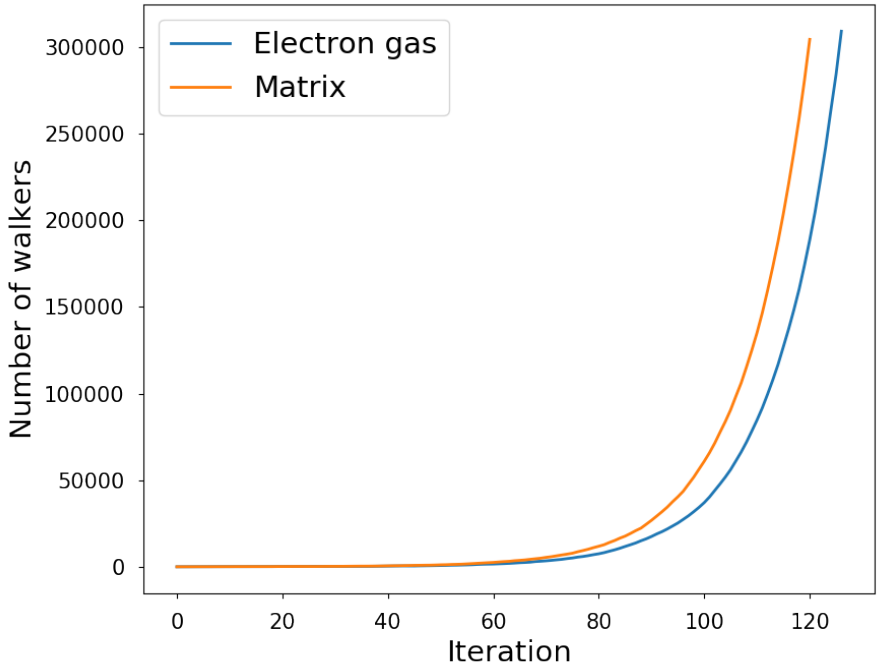


Figure 6.9: The population growth in simulations leading to Figs. 6.5 and 6.6.

Fig. 6.10 a comparison of the energy computed by the projection method, and via the energy shift, is presented. The number of walkers is represented on the secondary axis. It can be seen that both estimators of the energy converge to the same value.

The energy shift changes the death probability (see Eq. (4.14)). If the value of the energy shift is positive, then less walkers die and there is a possibility that they will be cloned if $S > H_{ii}$. If the energy shift is negative, then the death probability is increased. The shift is varied until the number of new walkers is the same as the number of dead walkers in a single timestep, which allows us to keep the population constant when the shift converges to the ground state energy, as illustrated in Fig. 6.10. The shift is not usually varied right from the start of the simulation but only once the population is large enough.

The initial value of the shift can be chosen arbitrary. A positive shift can also be used to decrease the number of deaths and increase the growth of the population at the beginning of the simulation.

We ran a series of simulations where the initial value of the energy shift is set to different values, while the other parameters are kept at the same value for all simulations:

- Model = Electron gas
- Number of starting walkers = 10
- Damping parameter $\zeta = 0.2$
- Number of walkers needed for adjusting shift = 300000

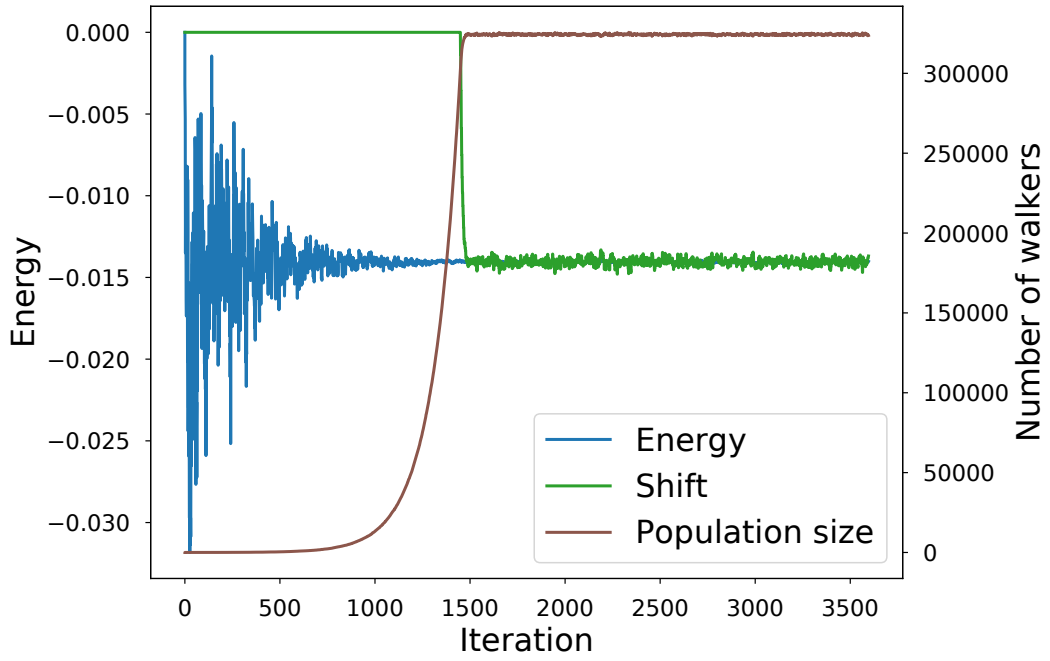


Figure 6.10: The convergence of the projected energy (blue line) and the energy shift (green line) to the ground state energy. The total number of walkers (brown line), creating a plateau once shift is enabled, is represented on the secondary axis.

- The energy shift is varied every 2 timesteps
- Timestep $\delta\tau = 0.5$
- Lattice constant $a = 10$
- Number of real unit cells in every direction $N_{\mathrm{k}} = 4$
- Number of electrons $N_{\mathrm{el}} = 2$
- Potential factor $U = \frac{1}{16}$
- Kinetic factor $T = 16$

The first thing which need to be verified and which we present in Fig. 6.11 is that the shift does not affect the projected energy. The energy shift changes the death probability but it is changed on all determinants by the same amount, so it should not affect the computation of the projected energy.

As we can see from Fig. 6.11, even though every simulation uses a different initial value of the energy shift, all converge correctly to the same value of the ground state energy.

The second thing we need to check is that the shift converges to the ground state energy. We present this in Fig. 6.12:

The ground state energy is represented by a dashed line in Fig. 6.12 . We can see that all simulations modulate the shift correctly, and that after a number of

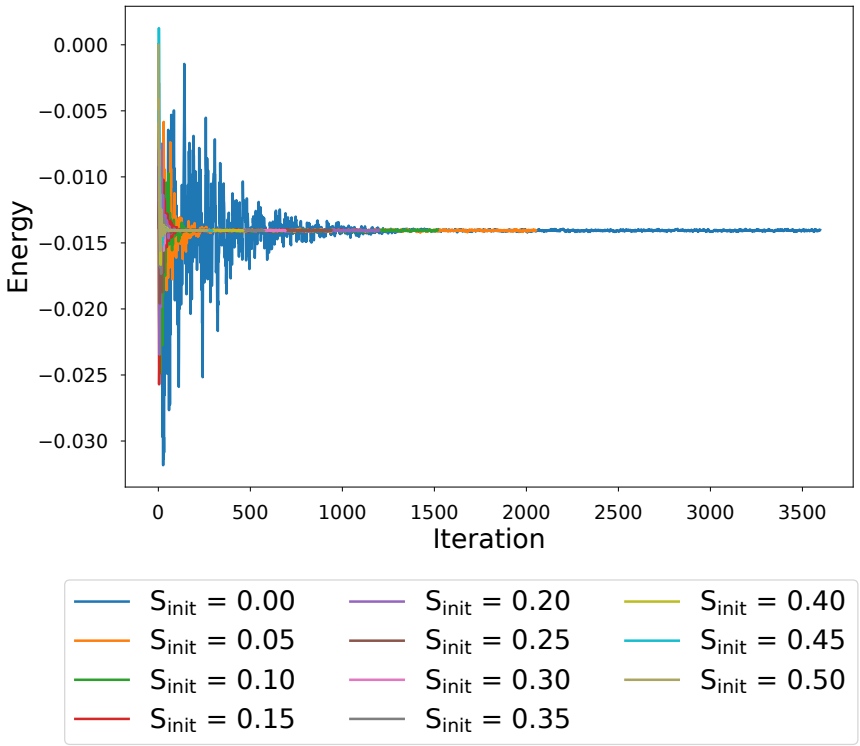


Figure 6.11: Profiles of the projected energy, for different initial values of the shift.

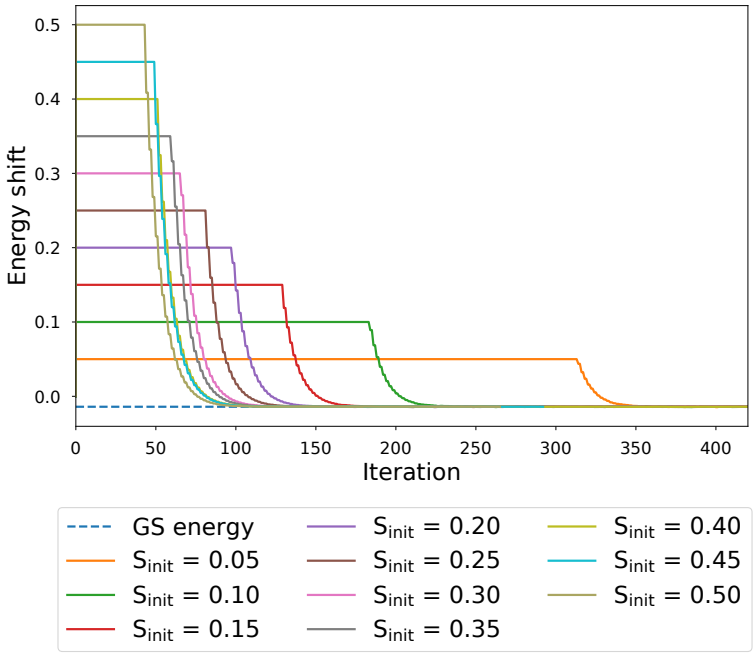


Figure 6.12: The convergence of the energy shift (solid lines) to the ground state energy (blue dashed line) is presented for simulations with different initial values of the energy shift.

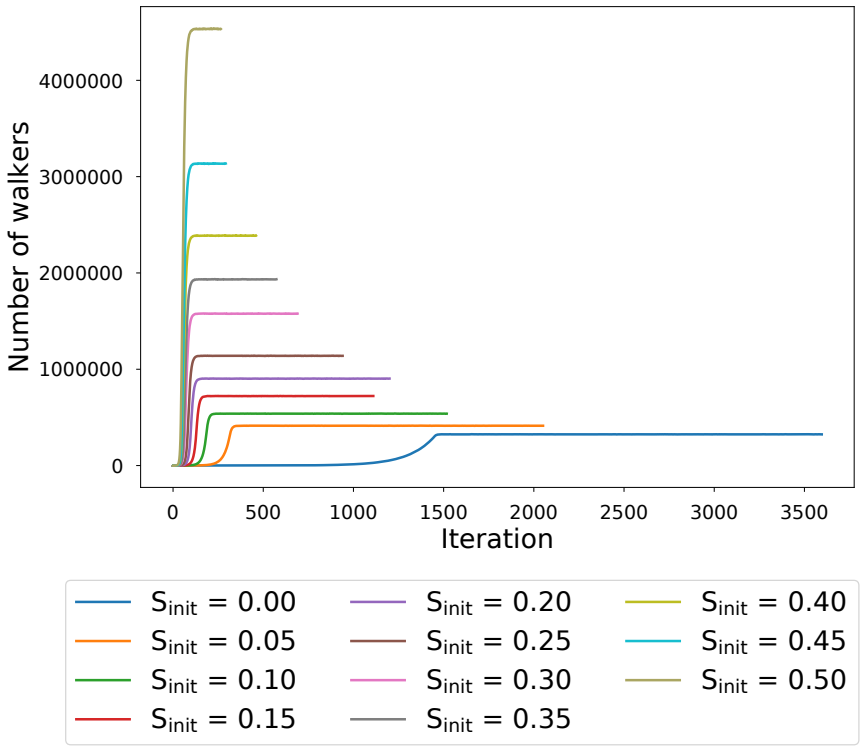


Figure 6.13: The growth of a population size (solid lines) is shown for various values of the energy shift.

iterations the shift converges to the ground state energy. This is an important result, as it provides us with an independent way of computing the ground state energy, in addition to the projected energy.

Another thing we can notice in Fig. 6.12 is that the shift in every simulation converges to the ground state energy after a different number of iterations. This is a direct consequence of the used parameter values. The shift starts to vary after the total population consists at least of three hundred thousands of walkers. The higher and positive the initial value of the energy shift is, the lower the probability of death is and the faster the population grows. It follows that for high initial values of the shift the condition of three hundred thousands walkers is fulfilled earlier. This trend is presented in Fig. 6.13. In Fig. 6.14 we zoom to the first five hundred iterations to show the exponential growth more clearly.

It can be observed that with high initial values of the energy shift the plateau is achieved in a dozens iterations. The number of walkers is higher so the precision, with which the projected energy is computed, is higher. But it turns out that adopting high initial values of the shift is not always beneficial. All simulations were run for the same amount of time, six CPU hours, and it can be seen that for the highest initial value of the shift $S_{\text{init}} = 0.5$, only less than forty timesteps were computed, because the higher the number of walkers, the higher the computational time. A larger number of walkers increases the precision of the computation, but at the cost of CPU time. In Table 6.1 the results are summed up, the initial value of the energy

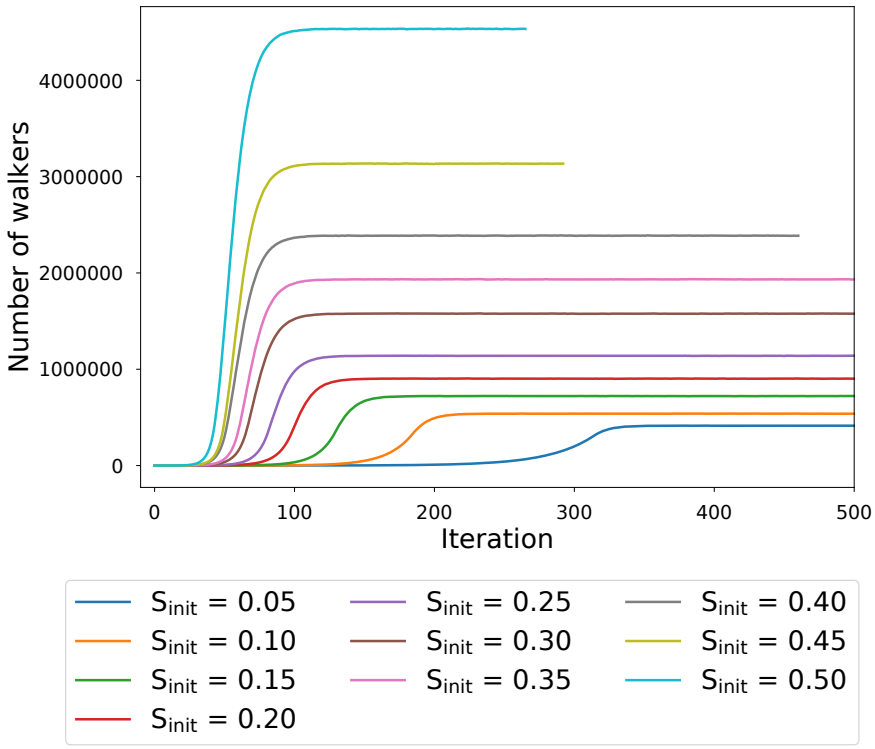


Figure 6.14: Zoom of Fig. 6.13 to the exponential growth

shift S_{init}, the number of iterations needed to achieve the plateau T_{P} and the height of plateau N_{P}.

The last thing presented in this section is the profile of the size of the population for an initial value of the energy shift $S_{\text{init}} = 0.5$ in Fig. 6.15. In this figure we also present the number of spawned, cloned, dead and annihilated walkers in every situation. The brown line in Fig. 6.15 representing the total population is thus a cumulative line of other plotted data - spawned and cloned walkers are summed up and dead and annihilated walkers are subtracted.

We can notice a small peak of cloned walkers at the beginning of the simulation. As the number of walkers grow, more walkers can be cloned, but at the same time the shift is lowered from the high initial value to match the ground state energy and the lower the shift, the lower the probability of cloning.

Once the plateau is reached, the walkers still undergo the process of spawning, dying and the annihilation, but the number of spawned walkers becomes same as the sum of dead and annihilated walkers and the total population is stable.

6. Damping parameter

In this section we study the damping parameter ζ used in the Eq. (4.21). It controls the way the energy shift is modified from one time step to the next. The damping parameter prevents the energy shift S from oscillating too wildly around the exact

Table 6.1: Data from simulations with different initial value of shift S_{init}. T_{P} is the number of iterations needed to achieve the plateau and N_{P} is the height of the plateau.

S_{init}	T_{P}	N_{P}
0.00	1468	324344
0.05	346	413549
0.10	223	539024
0.15	170	721571
0.20	140	902652
0.25	126	1139512
0.30	113	1577238
0.35	108	1933351
0.40	101	2387012
0.45	99	3134734
0.50	95	4533601

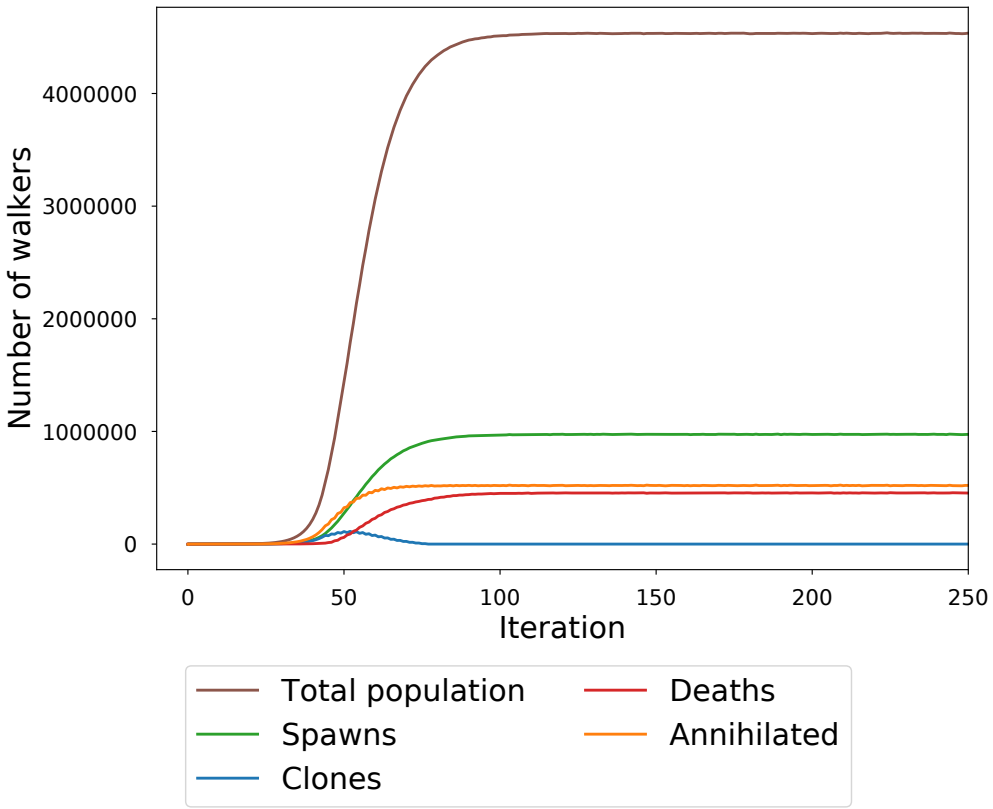


Figure 6.15: Size of the population as a function of the number of iterations, using an initial shift $S_{\text{init}} = 0.5$. The number of spawned, cloned, dead and annihilated walkers per iteration is also shown.

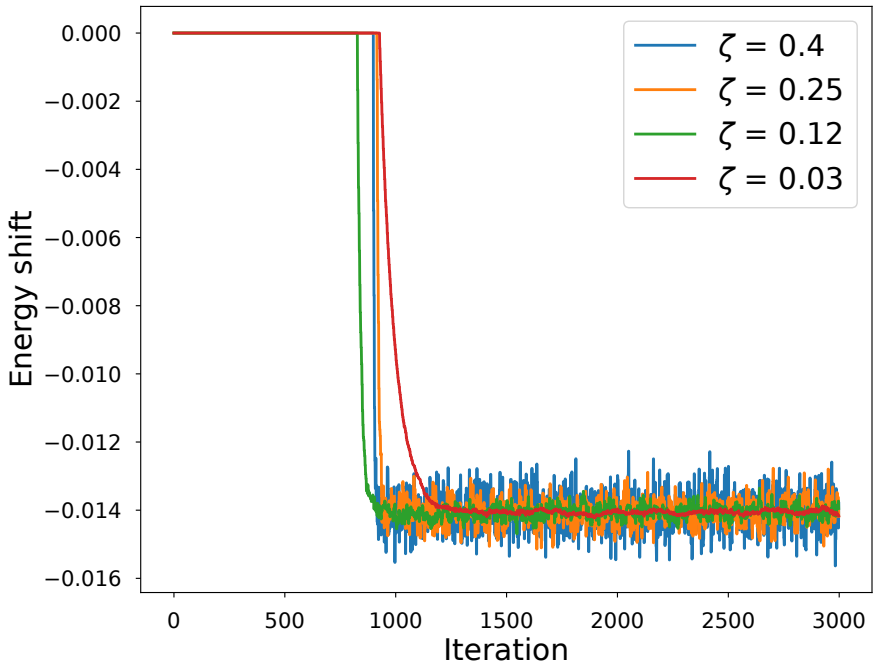


Figure 6.16: The impact of the damping parameter ζ on the energy shift S.

value of the energy. We ran a series of simulation with identical parameters, varying only the damping parameter:

- model = Electron gas
- Lattice constant $a = 10$
- Number of electrons $N_{\mathrm{el}} = 2$
- Number of real unit cells in every direction $N_{\mathrm{k}} = 4$
- Timestep $\delta\tau = 0.7$
- Walkers needed for adjusting shift = 100000
- Potential factor $U = \frac{1}{16}$
- Kinetic factor $T = 16$

Fig. 6.16 shows that that for larger values of the damping parameter, fluctuations of the energy shift are larger. This causes that the standard deviation of the energy shift is higher too.

When populations of simulations with different damping parameters are plotted, as shown in Fig. 6.17, it is possible to see that the population needed to achieve a plateau is inversely proportional to the damping parameter.

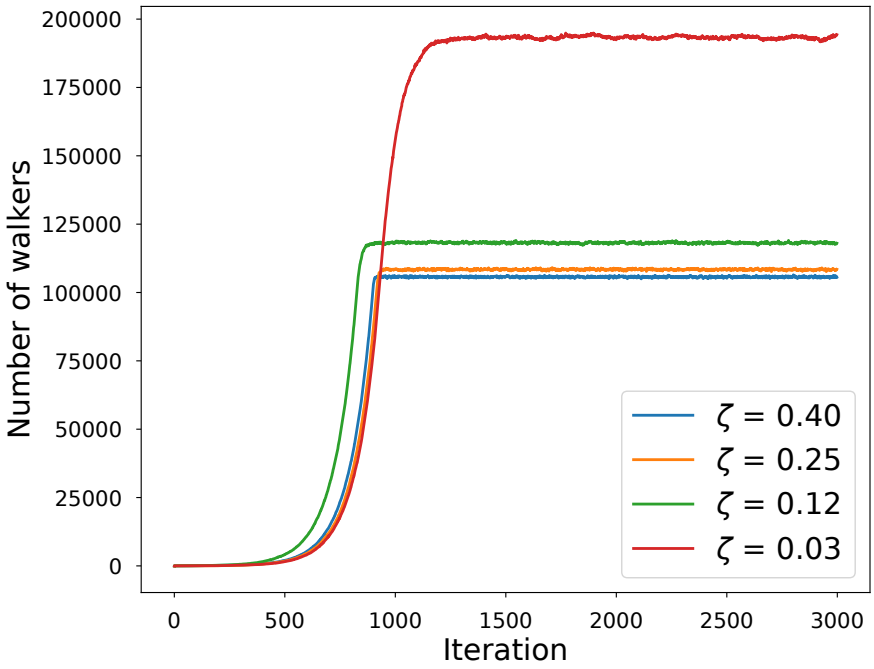


Figure 6.17: The effect of the damping parameter ζ on the growth and the size of a population.

It is difficult to estimate the number of iteration where the plateau is achieved. In order to describe this, we define a variable $T_{X\%}$.:

$$|S(T_{X\%}) - E_{\text{exact}}| < |E_{\text{exact}}| \cdot \frac{X}{100}, \tag{6.1}$$

such that $T_{X\%}$ is the lowest number of iteration since the variation of shift is started where the difference between shift and exact energy is lower than X percent of the absolute value of the exact energy.

The dependence of the standard deviation of the energy shift on the damping parameter, together with the values of $T_{10\%}$ and $T_{1\%}$ are presented in Table 6.2. $T_{10\%}$ can be considered as a rough estimate of the number of iterations needed to achieve the plateau with the shift enabled. $T_{1\%}$ is a finer estimate. The corresponding data are plotted in Figs. 6.18 and 6.19.

Table 6.2: Various values of the damping parameter ζ and studied variables: the standard deviation of the energy shift $\sigma(S)$, the rough estimate of the number of iterations needed to achieve the plateau $T_{10\%}$ and the finer estimate $T_{1\%}$.

ζ	$\sigma(S) \cdot 10^3$	$T_{10\%}$	$T_{1\%}$
0.010	0.008	456	928
0.012	0.016	384	752
0.015	0.025	304	586
0.018	0.039	258	480
0.020	0.026	230	460
0.030	0.054	150	266
0.050	0.072	90	180
0.070	0.126	62	126
0.080	0.146	58	110
0.090	0.149	48	86
0.120	0.184	34	74
0.150	0.218	26	40
0.200	0.305	20	34
0.250	0.370	20	30
0.350	0.515	10	18
0.400	0.591	8	28
0.500	0.741	6	18
0.600	0.851	6	18
0.700	1.110	2	18
0.800	1.286	4	38
0.900	1.545	4	30
1.000	1.609	2	16

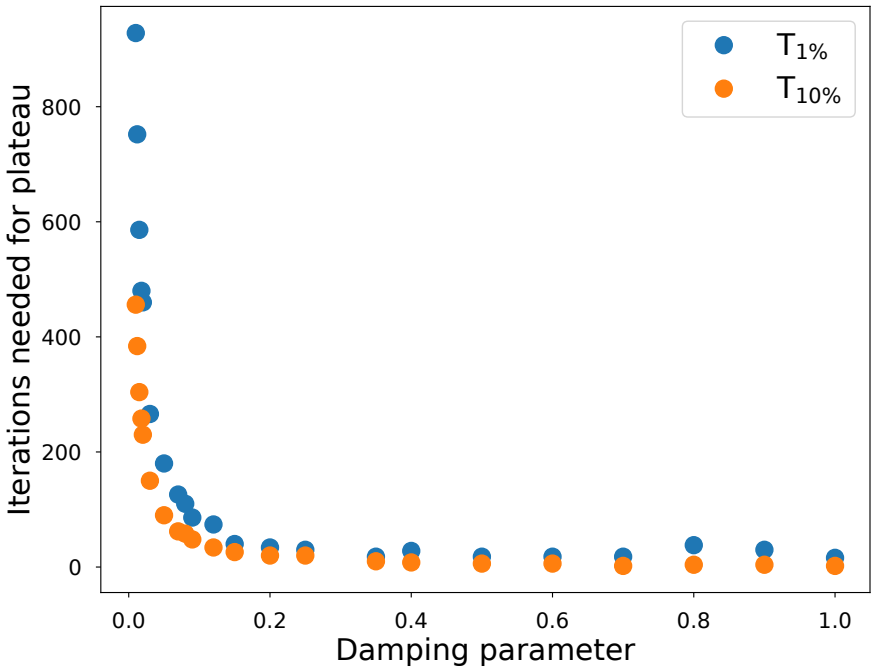


Figure 6.18: The number of iterations needed to achieve a plateau, versus the damping parameter ζ. Orange points show the rough estimate, blue points show more finer estimate.

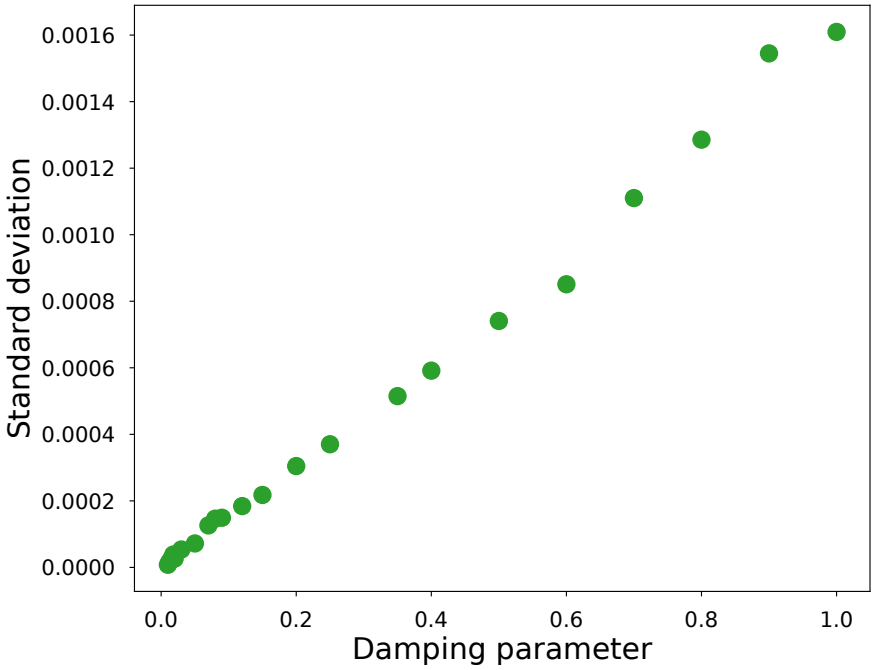


Figure 6.19: The standard deviation of the energy shift versus the damping parameter.

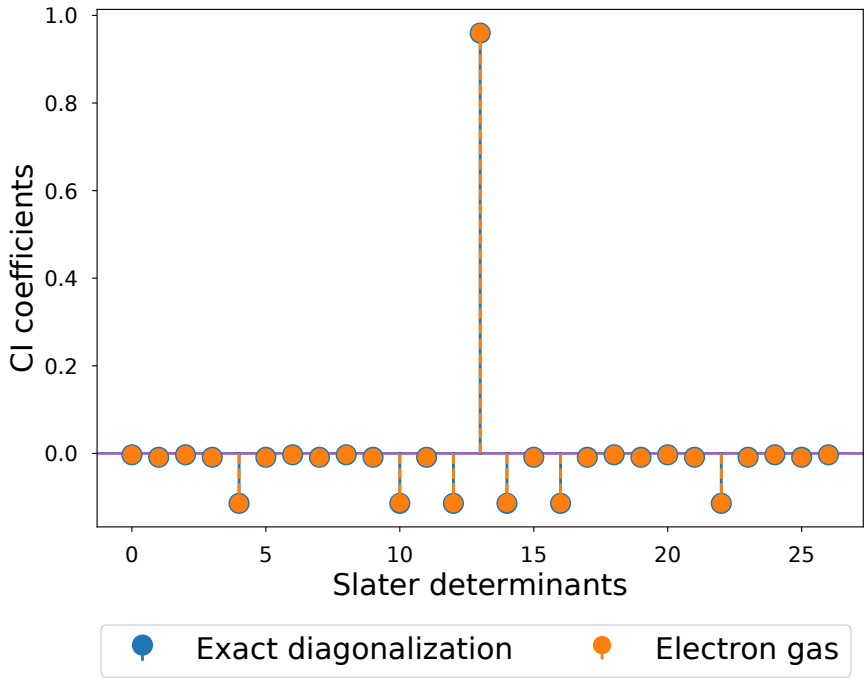


Figure 6.20: The ground state of the electron gas with the factors from Eq. 5.7 set to $U = 0.01$ and $T = 1$. x-axis is used to index Slater determinants, y-axis is used for CI coefficients. Orange points computed by FCIQMC are plotted over blue points computed by exact diagonalization.

To conclude this section, it can be said that the lower the damping parameter, the lower the fluctuations of the shift around the exact energy. On the other hand, the number of iterations needed to achieve the plateau with shift enabled grows exponentially with the inverse of the damping parameter. If the shift is used, then the damping parameter must be carefully set.

7. Interaction strength

In the following section, parameters U and T which scale kinetic and potential energy in Eq. (5.7) are studied. The ratio of these energies is the only relevant factor, we can therefore keep $T = 1$, and study the behavior of the system for different values of U. The decomposition of the ground state is presented in Figs. 6.20, 6.21, 6.22 and 6.23.

We can observe that the amplitude of the CI coefficient on the HF determinant decreases with increasing U, the corresponding weight being transferred to determinants with higher momentum. Second highest values of CI coefficients are still the nearest neighbours shown in Fig. 6.4. We therefore observe a finite effect of higher interaction strength, even if it is quite challenging to interpret it in real space terms. Moreover, the density of our system is so low that correlation effects remain quite minor, even in the limit $U \to \infty$.

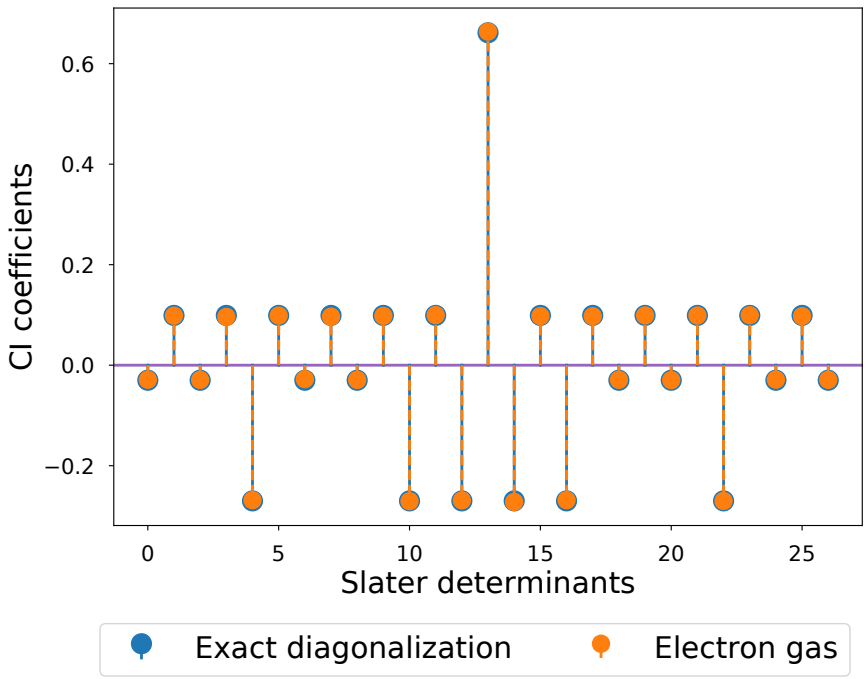


Figure 6.21: The same as in Fig. 6.20, but with factor $U = 0.1$.

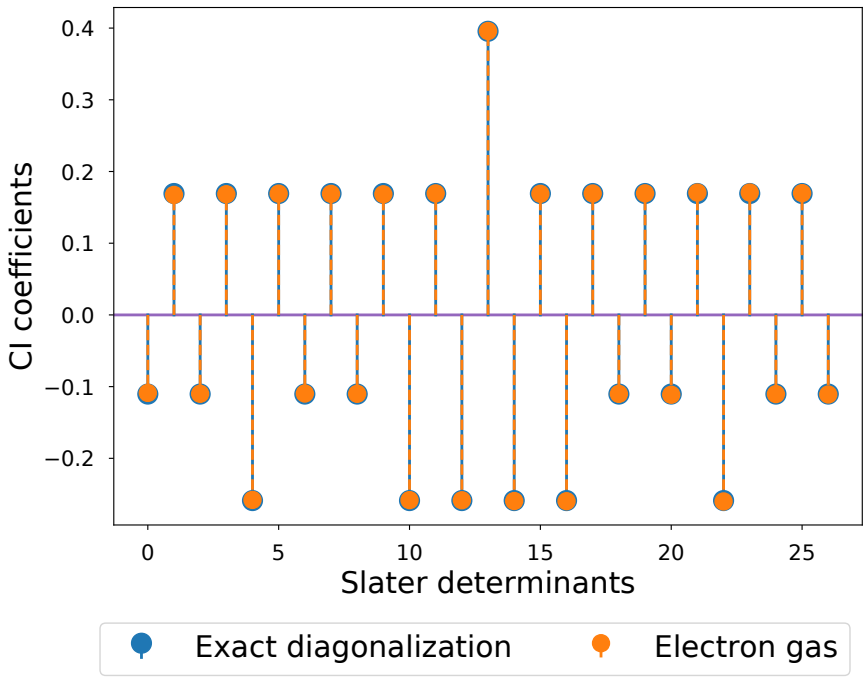


Figure 6.22: The same as in Figs. 6.20 and 6.21, but with factor $U = 1$.

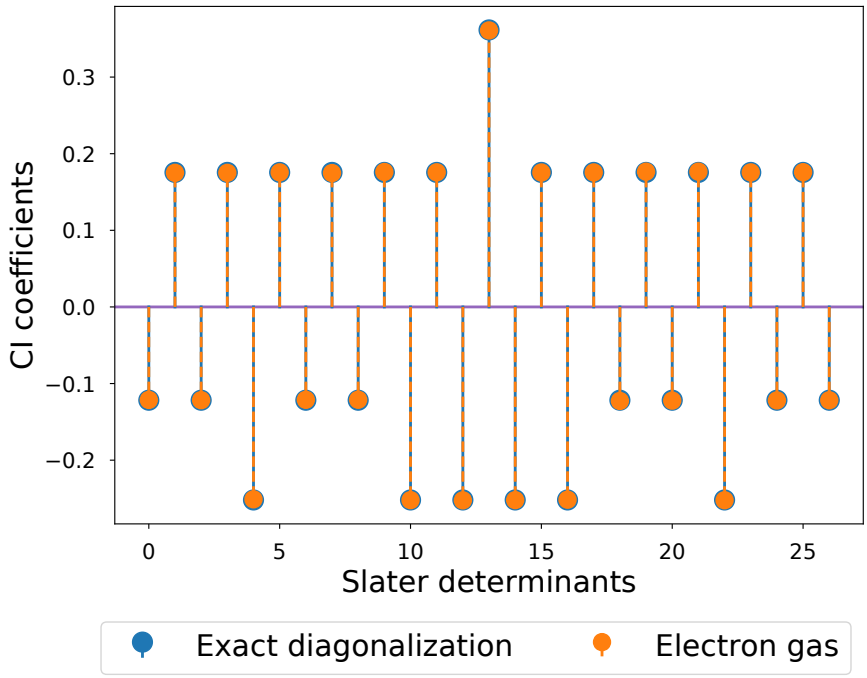


Figure 6.23: The same as in Figs. 6.20, 6.21 and 6.22, but with factor $U = 10$.

Another quantity of interest is the growth of the population as a function of U, illustrated in Fig. 6.24. No plateau appears, even for large values of U. We were not able to observe the emergence of a plateau in our simulations, which is probably due, once again, to the weakness of correlation effects in this low density limit.

8. Symmetry considerations

It is a general fact that if G is the group of symmetries of the Hamiltonian $\hat{H}$ of a system, then, states which transform according to different representations of G are not connected by $\hat{H}$, i.e. the matrix element of $\hat{H}$ between them is zero.

This fact has direct consequences in the case of FCIQMC: If the simulation is started with a single Slater determinant belonging to some representation, then all states generated by the Monte Carlo simulation will have finite CI weights only on determinants belonging to that same representation. Indeed, at the start of the simulation all walkers are located on the same determinant. They can spawn new walkers only on coupled determinants and as determinants with different symmetries are not coupled, the population of walkers and the FCIQMC algorithm are locked in the symmetry of the starting determinant.

It follows that only a subset of the full matrix is visited during a simulation, but more importantly, that if the starting determinant does not belong to the same representation as the true ground state, then the algorithm will converge only to the lowest eigenvalue in a subspace that does not contain the true ground state. The converged energy estimator will be higher than the ground state energy. This

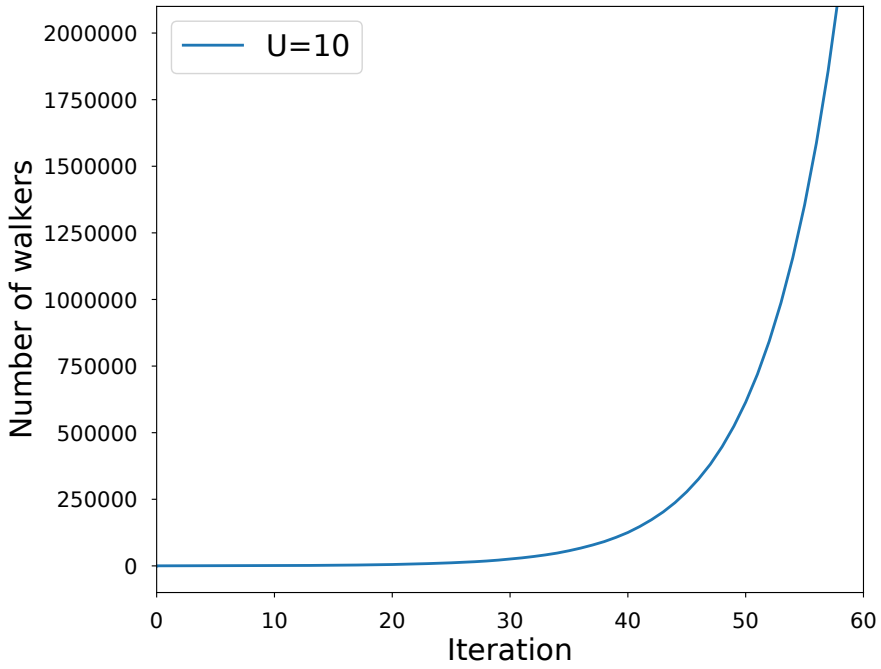


Figure 6.24: The population growth in the simulation with potential factor set to $U = 10$.

justifies why, as was stated earlier, the choice of starting determinant D_0 is crucial, and it should have a 'finite overlap' with the true ground state. In other terms, it should belong to the same representation as the ground state.

For $U \to 0$, the ground state should have the lowest possible momentum to minimize the kinetic energy. We ran a series of simulations and then compared them with the results of exact diagonalization of the full $\hat{H}$ matrix. The variable in the simulation was the total momentum of the system:

$$\mathbf{k}_{\mathrm{T}} = \sum_{i}^{N_{\mathrm{el}}} \mathbf{k}_i \tag{6.2}$$

The values of the parameters for the simulations were the following:

- model = Electron gas
- Lattice constant $a = 10$
- Number of electrons $N_{\mathrm{el}} = 2$
- Number of real unit cells in every direction $N_{\mathrm{k}} = 4$
- Potential factor $U = \frac{5}{8}$
- Kinetic factor $T = 16$
- Number of starting walkers $N_{\mathrm{start}} = 10$

The starting determinants of the corresponding simulations are shown in Figs. 6.25.

The computed ground states are shown in Figs. 6.26-6.29. Each result is compared with the result of the exact diagonalization for the block matrix corresponding to the restriction of $\hat{H}$ to the subspace with the symmetry of the initial state of the simulation (here, the subspace of $\hat{H}$ with a given value of momentum which corresponds to a given eigenvalue of the momentum operator, the momentum being conserved in our model as a consequence of translational invariance). The true ground state of the system has zero total momentum and is therefore shown on the first figure 6.26.

The energies computed from simulations are also compared with the results of exact diagonalization of matrices of $\hat{H}$ restricted to the subspaces with the proper symmetry. The results are shown in Figs. 6.30, 6.31. The lowest eigenvalue plotted with dashed line in figures is computed using the exact diagonalization of the full matrix.

It is apparent that the simulations with different symmetry do not converge to the true ground state energy of the system, as expected and as discussed above.

9. Complex matrix

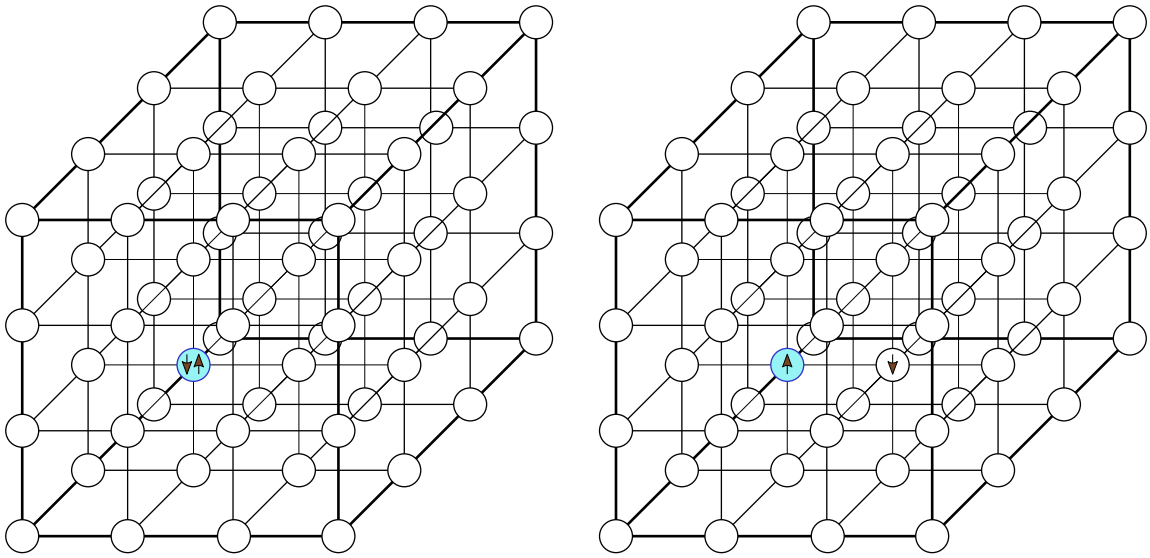
In this section, we explore the *Matrix* model with complex CI coefficients which could be applied for the case of quantum dots for instance. We present some results using a small matrix, but they demonstrate the correctness of the implementation, and can be scaled up easily (contrary to the electron gas model, where handling more electrons is a complex task). A small random complex Hermitian matrix, see Eq. (6.3), easily computable using exact diagonalization, is generated for this illustration.

$$\begin{pmatrix} 0.1+0.0i & -0.4-0.5i & -0.2-0.2i & -0.4+0.7i \\ -0.4+0.5i & 0.2+0.0i & -0.4-0.5i & -0.2-0.3i \\ -0.2+0.2i & -0.4+0.5i & 0.4+0.0i & -0.8-0.1i \\ -0.4-0.7i & -0.2+0.3i & -0.8+0.1i & 0.3+0.0i \end{pmatrix} \tag{6.3}$$

Three different simulation results are shown. They all use:

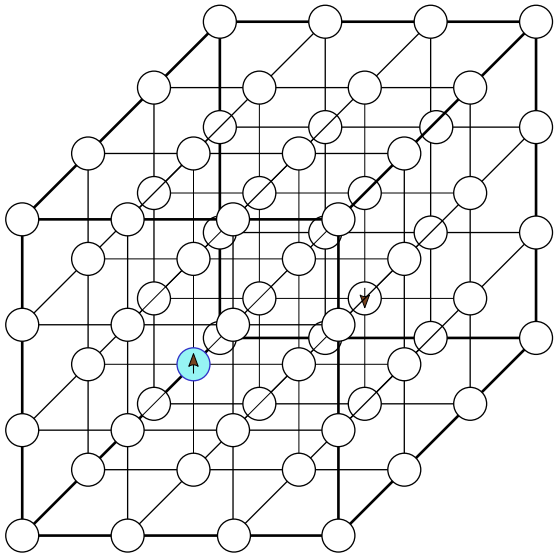
- Model = Matrix
- Timestep $\tau = 0.008$
- Initial population $N_{\text{start}} = 100$

On the other hand, two parameters are varied in each simulation: the target population level N_{R}, after which the rotation step is added to the algorithm; and the number of iterations between two rotation steps T_{R}. The values for these parameters are shown in Table 6.3.

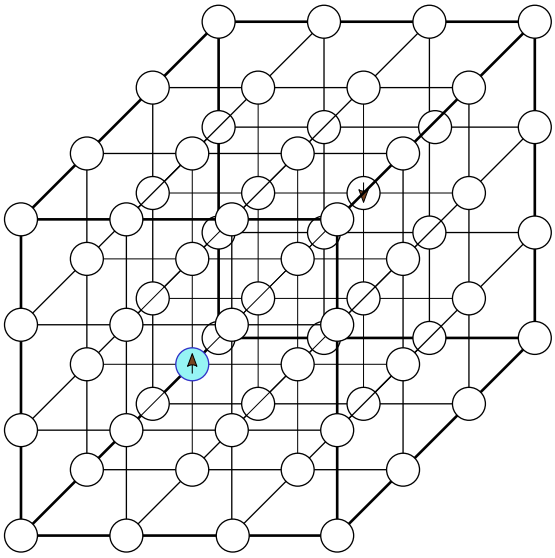


(a) $k_T = (0, 0, 0)$

(b) $k_T = (1, 0, 0)$



(c) $k_T = (1, 1, 0)$



(d) $k_T = (1, 1, 1)$

Figure 6.25: Starting states for simulations with different momenta. The Γ point is highlighted by the blue color.

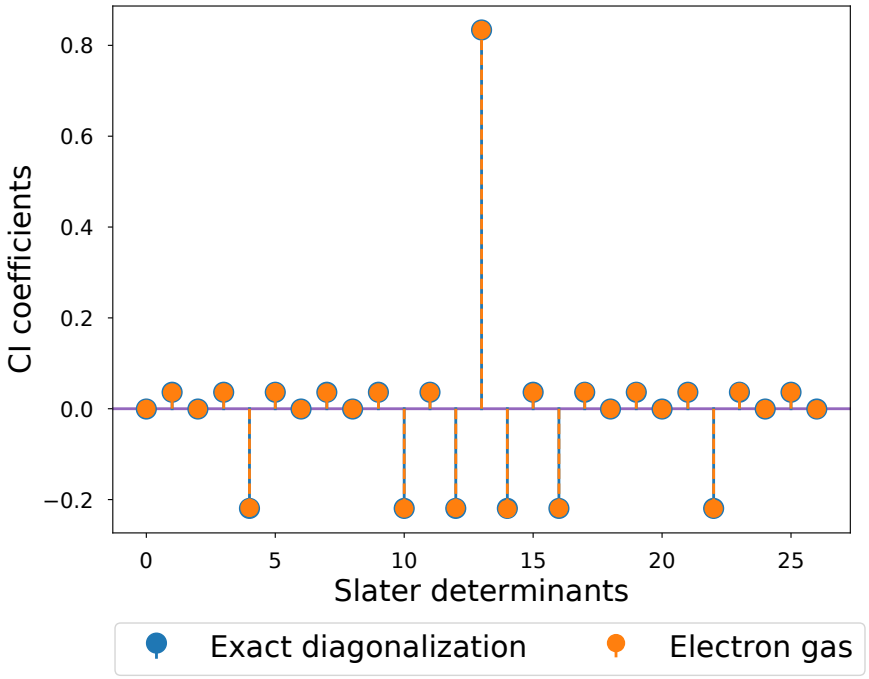


Figure 6.26: The ground state of the electron gas with total momentum $k_T = (0, 0, 0)$.

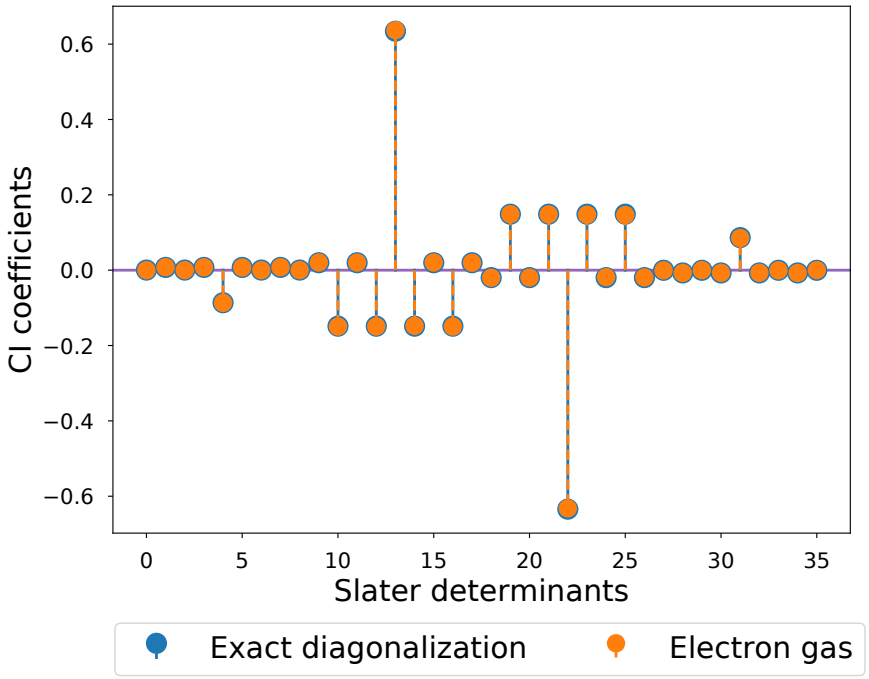


Figure 6.27: The ground state of electron gas as in Fig. 6.26, but with total momentum $k_T = (1, 0, 0)$

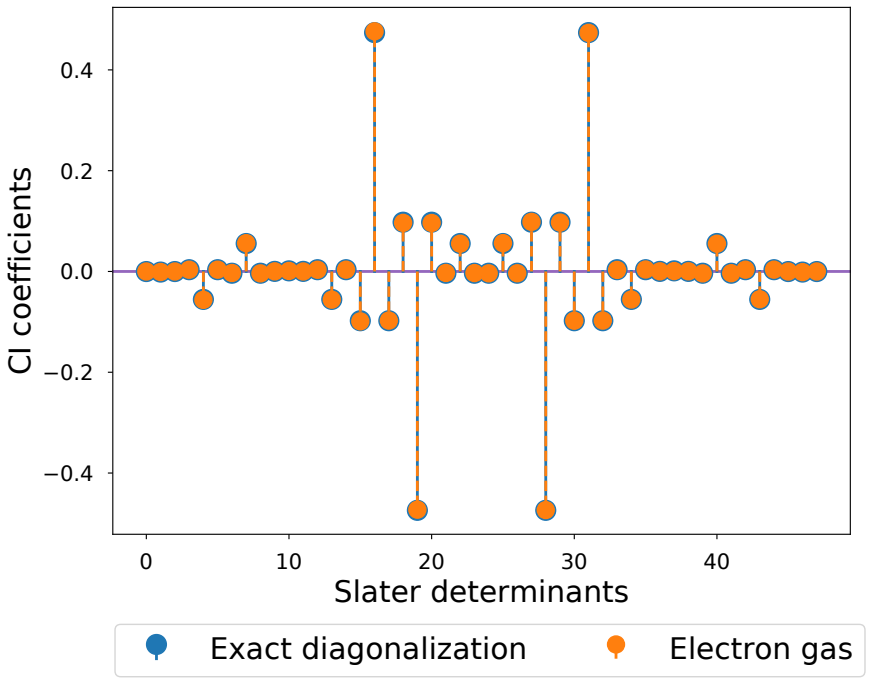


Figure 6.28: The ground state of electron gas as in Figs. 6.26 and 6.27, but with total momentum $k_T = (1, 1, 0)$

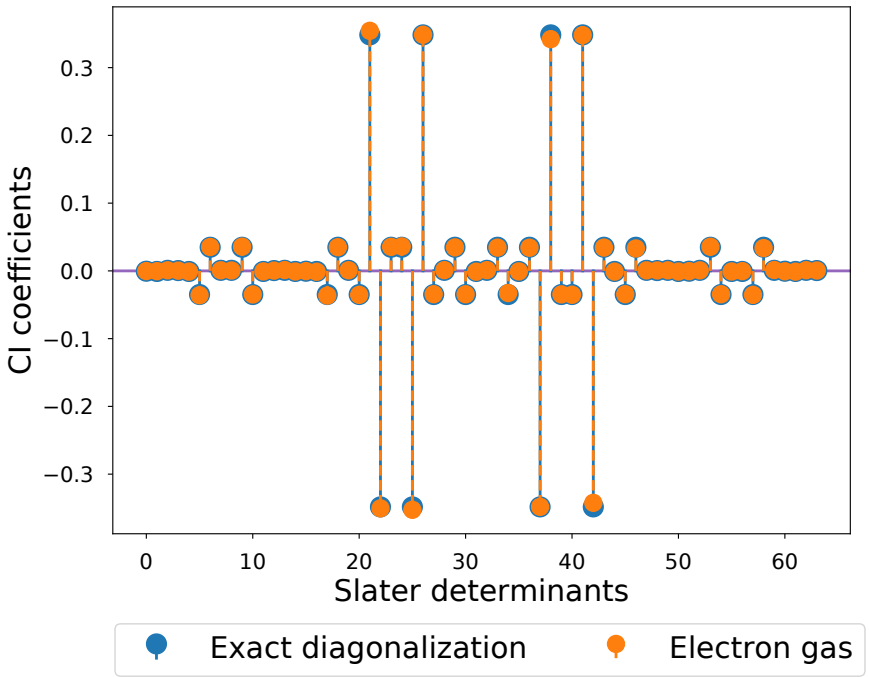


Figure 6.29: The ground state of electron gas as in Figs. 6.26, 6.27 and 6.28, but with total momentum $k_T = (1, 1, 1)$

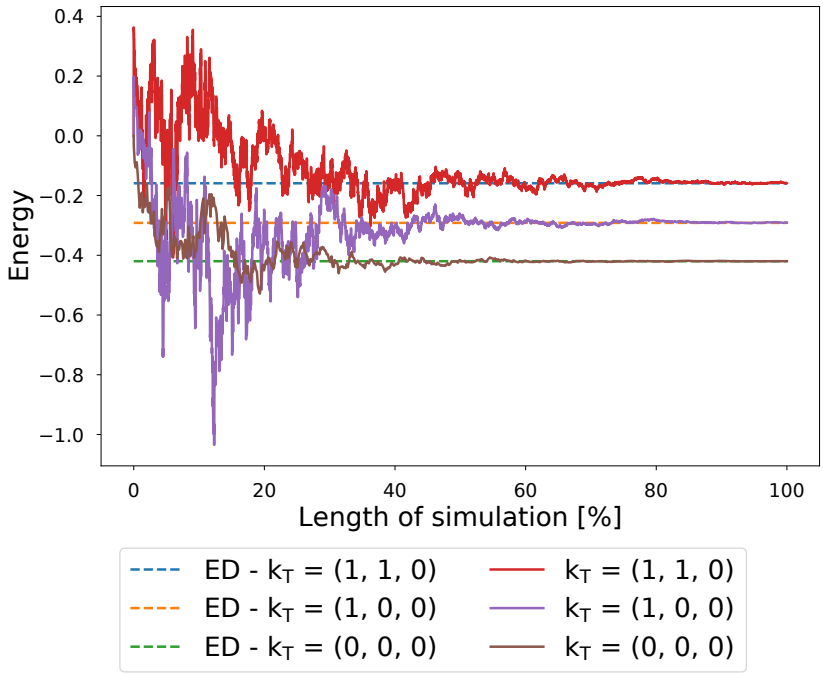


Figure 6.30: Energies of simulations with different momenta. Solid lines represent the energy computed using FCIQMC, the energy computed using exact diagonalization is plotted with dashed lines (the lowest corresponds to exact diagonalization of the full matrix).

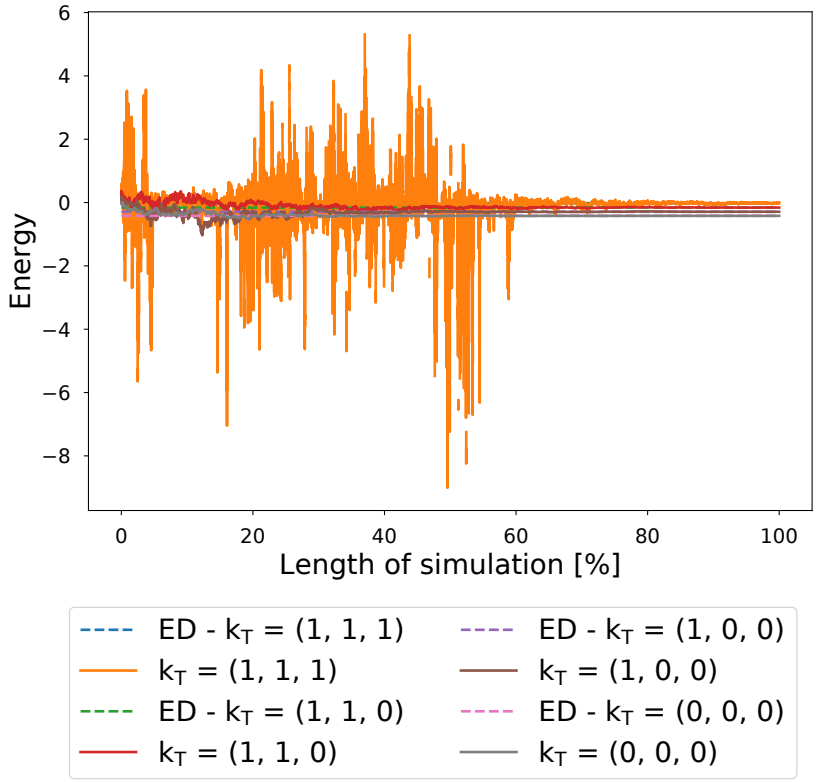


Figure 6.31: The same as in Fig. 6.30 with the data from an additional simulation.

Table 6.3: Parameters which are different for each simulation of complex matrix (6.3). N_{R} is the target level of population after which the rotation step is added to the algorithm. The rotation step is triggered every T_{R} timesteps.

	N_{R}	T_{R}
A)	200000	20
B)	∞	0
C)	0	1

The first simulation proceeds in the same way as standard simulations with a shift variable from Sec. 6.5., a certain number of walkers is reached and then the phase shift is applied every T_{R} timesteps. The second simulation is without the rotation step, in order to demonstrate its impact. In the third simulation, the phase shift is applied every timestep from the start of the simulation.

The real and the imaginary components of the energy are shown in Fig. 6.32. This figure clearly shows that if the rotation step is not executed, then the energy estimator possesses a finite imaginary component, which is unphysical. Conversely, the rotation step totally suppresses this component and allows the simulation to converge to the physical result. It can be seen that both simulations with enabled

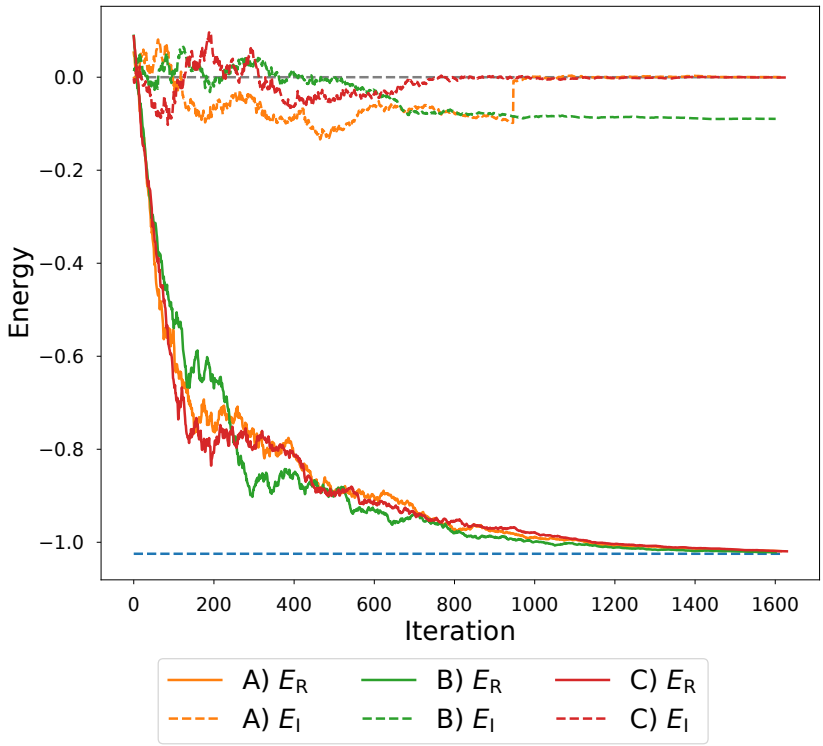


Figure 6.32: Real (solid lines) and imaginary (dashed lines) energies computed for simulations of complex Hermitian matrix (6.3). The variant A) enables rotation step after the population of 200 000 walkers is reached. The variant B) does not use the rotation step at all. The variant C) uses the rotation step since the beginning of the simulation. The dashed blue line is the ground state energy computed using exact diagonalization. Zero energy is shown as dashed grey horizontal line.

rotation converge to the correct values using a similar number of iterations. It follows that it is not necessary to trigger the rotation step every timestep and thus to save some CPU time.

The ground state of the first simulation with rotation step is displayed in Fig. 6.33, and can be compared with that obtained from the second simulation, where the rotation step is disabled, Fig. 6.34. The final ground state without the rotation step possesses a finite imaginary population on the Hartree-Fock determinant, which is the direct cause of the unphysical imaginary component of the energy estimator, shown in Fig. 6.32. It can be, however, rotated once in the end of the simulation, yielding correct ground state with zero imaginary population on the Hartree-Fock determinant, see Fig. 6.35.

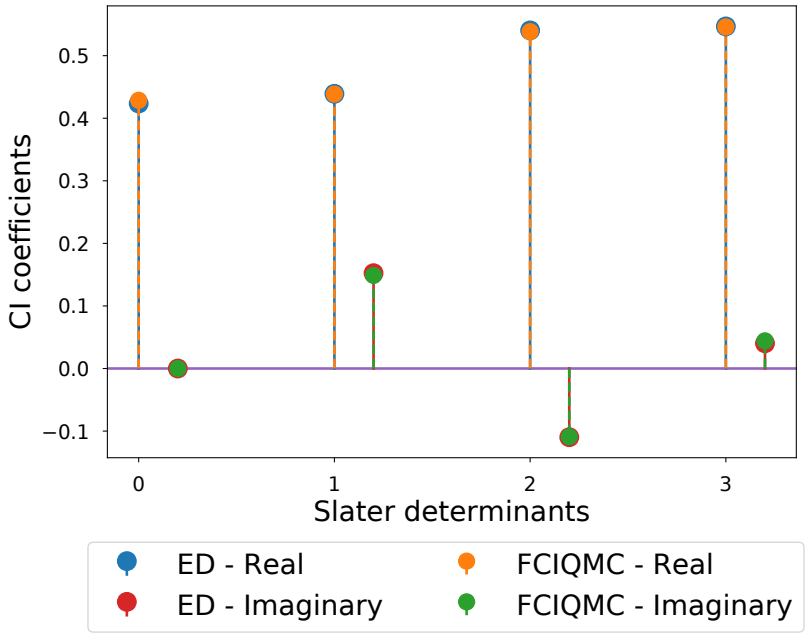


Figure 6.33: CI coefficients of the ground state of the complex matrix (6.3) computed using exact diagonalization (ED) and full configuration interaction quantum Monte Carlo (FCIQMC), including the rotation step. The x-axis is used as an index for the Slater determinants included in the basis. The real and imaginary parts of each coefficient are shown side by side. The imaginary part is shifted to the right for readability.

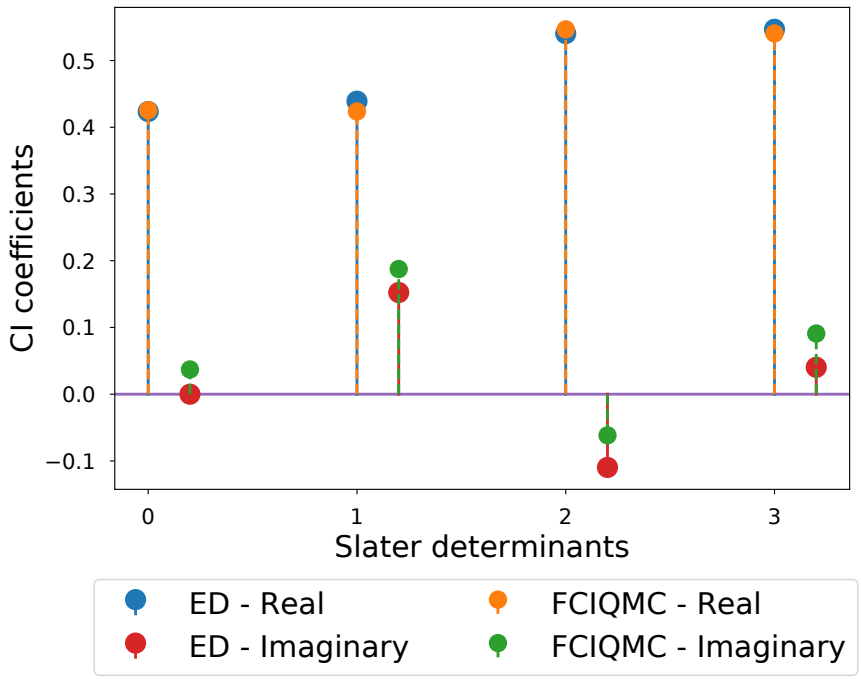


Figure 6.34: The same as Fig. 6.33, but without the rotation step in the FCIQMC procedure.

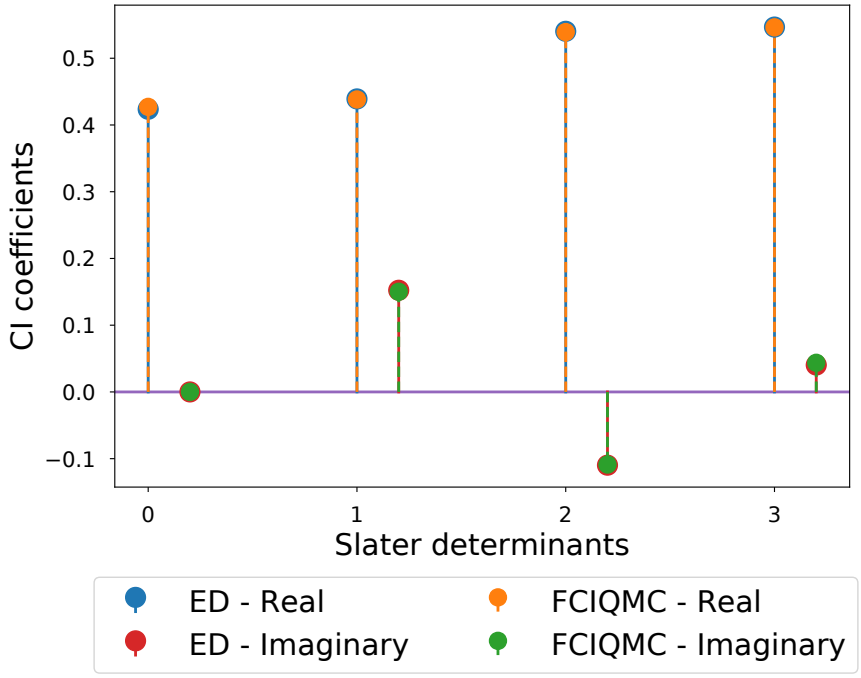


Figure 6.35: The population from Fig. 6.34, rotated after the end of simulation.

Chapter 7

Program

1. Program

As far as computational physics is concerned, we often witness a tradeoff between computation time, and memory consumption. One can often be converted into the other. The recent technological developments have been such in the computing industry, that time is getting cheaper than memory. This is one of the attracting features of the FCIQMC approach, which relies on parallelization, i.e. high computational time, while needing relatively little memory. In particular, FCIQMC does not need to store the full Hamiltonian matrix. Instead, when a matrix element is needed it is computed on the fly. The algorithm belongs to the 'embarrassingly parallel' category, which can be easily parallelized.

2. Classes

The program is written in C++ , and the data post-processing is done using Python and Jupyter notebooks. The code is divided into classes, themselves split into core classes, and model classes. Core classes deal with the model-agnostic parts of the algorithm: the main cycle, management of the population of walkers, walker and data saving/loading. The second category of classes deals with the specific models. Two models are implemented (homogeneous electron gas and matrix). Inheritance and virtual classes are used in order to structure this part of the code, using the factory design pattern.

All in all, the code implements nine classes:

- Main - starts simulation, reads configuration file and contains whole population dynamics cycle.
- Walker - defines basic object of the simulation, the walker and its attributes like determinant and sign.
- Population - structure where walkers are stored and ease work with large number of walkers.
- H5saver - definitions of tables in h5 file and saving procedures.
- ModelFactory - factory design pattern connecting *model* class with specific models.
- Model - virtual class, overridden by specific models.

- ElectronGas - model of homogeneous electron gas described in Sec. 5.1..
- Matrix - model of Matrix described in Sec. 5.2..

2.1. The Main class

The program takes an INI configuration file as input (*config.ini*), where all necessary parameters are specified. A sample input file is provided as code snippet 7.1. The Model, H5saver and Population classes are instantiated, including a few starting Walkers on a chosen determinant. The user can then choose whether to run a Monte Carlo simulation, or dump the full Hamiltonian for different purposes.

2.2. The Walker class

The Walker class has the following attributes:

- determinant,
- stacks,
- is_real,
- zero_excitation_matrix_element,
- energy.

The bitset class is used for storing the determinant. It physically corresponds to the occupation number formalism in memory. A set (unset) bit, i.e. a bit equal to one (zero), represents an occupied (unoccupied) spin-orbital. A one-to-one mapping between a bit in the bitset and the spin-orbitals allows the direct interpretation of a bitset in terms of Slater determinants.

An important trick in order to reduce the memory footprint of the program it to stack the Walkers. If Walkers are stacked we need to store only single Walker on each determinant and its integer attribute *stacks*. A single Walker stored in the memory then represents *stacks* Walkers on a determinant. We can compare it with with saving every single Walker separately. We recall the Fig. 4.2. The system is based on twenty seven different Slater determinants. Some CI coefficients are equal to zero and do not need to be saved, as no Walkers live on them. Thus only twenty one determinants are 'active'. Using stacks, we can represent each active determinant by one stack, and thus need to store twenty-one bitsets (determinants) only, together with twenty-one integers (stacks) indicating the size of the stack. This is much more efficient than storing ninety nine bitsets, each one corresponding to a single Walker: a simulation can easily reach a population of several millions of walkers, in which case the gain in memory provided by stacks is huge.

The *is_real* attribute is used to distinguish real Walkers from imaginary Walkers.

zero_excitation_matrix_element and *energy* are frequently used constants: *zero_excitation_matrix_element* is the diagonal Hamiltonian matrix element $\langle D_i|\hat{H}|D_i\rangle$ for the determinant associated to the walker; *energy* is matrix element $\langle D_i|\hat{H}|D_0\rangle$

between the determinant associated to the Walker, and the starting determinant. It is computationally efficient to store them, instead of computing them at each timestep.

2.3. The Population class

The Population class is a container for all active Walkers. Its attributes are following:

- is_complex - switch used for adding certain function like rotation if we work with complex numbers.
- R_walkers - container for real Walkers.
- R_new_walkers - container for newly spawned Walkers, which will not go through the death/cloning step and will be merged with R_walkers afterwards.
- R_ground_state_occupation - the number of Walkers living on the starting state.
- size - total number of Walkers stored only for information about simulation.
- spawned_walkers - number of Walkers spawned in the current timestep.
- dead_walkers - number of Walkers which died in the current timestep.
- cloned_walkers - number of Walkers which were cloned in the current timestep.
- total_energy - energy of the current state.
- R_shift - energy shift for real Walkers.
- damping_parameter - parameter used for adjusting energy shift, prevents energy shift from large oscillations.
- adjust_shift_each - number of iterations between two shift adjustments.
- last_R_size - for adjusting shift is needed the size of population from last timestep.

Usage of complex coefficients requires additional attributes: the R (I) prefix relates to the real (imaginary) part of a quantity. The attributes related to imaginary components are not listed here for the sake of conciseness.

The class implements the following methods:

- constructor - initializes the class, loads the necessary parameters such as the starting number of Walkers etc.
- spawn - iterate through the list of walkers, and trigger the spawning for each; new Walkers are added to separate container which is merged with current after death/cloning step.

- die - iterate through the list of parent Walkers, and give each walker a chance to die; dead Walkers are immediately removed from the container.
- merge - merge container of new Walkers with old ones.
- rotate - iterate through used determinants and apply the phase shift to populations on determinants so that the starting determinant has only real population after.
- compute_ground_state_occupation - this function is triggered when we want to be sure that the number of Walkers in the starting state is correct; it iterates through the list of Walkers and counts how many of them live on the starting state.
- adjust_shift - computes a new value of the shift according to Eq. (4.21) for following timesteps.
- compute_total_energy - computes the energy of the system using Eq. (4.22).
- statistics - function used mainly for debugging; it prints out all available information about the simulation, such as the number of walkers, the value of the shift, the occupation of each determinant etc.

2.4. The H5saver class

The algorithm manipulates large amounts of data, which need to be saved to disk. The modern standard for this purpose is the HDF5 format [34], which stands for 'Hierarchical Data Format', version 5. Every HDF5 file has its own hierarchical data structure with folders and subfolders, called groups. Inside them, datasets are stored. Any file size is allowed and the data are by default stored in binary format. HDF5 also offers the option to attach metadata to any element. HDF5 files are easily read using Python, which we use for data post-processing.

The H5saver class is implemented in order to add the functionality of saving into the HDF5 file format, which is not native in C++ . The data file for storage has the structure from Fig. 7.1.

The configuration file is copied into the *parameters* dataset. This helps for tracking problems, identifying results, or reproducing them. A dataset called *population* is used for saving all the dynamics of the population throughout the simulation. A group (folder inside the HDF5 file) called *walkers* contains all the fine structure details of the population: the determinant bitsets, the number of walkers living on them etc.

2.5. The Model class

The *Model* class is a virtual class. It declares all functions and attributes which a model needs to implement. All models inherit this class. Currently, the methods contained in the Model class are:

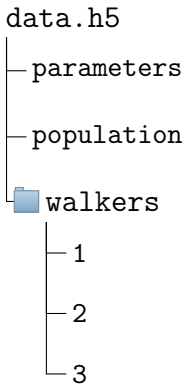


Figure 7.1: Structure of HDF5 file - The HDF5 file is named *data.h5*. Two datasets are stored directly in the root folder of HDF5 file, *parameters* and *population*. A single group *walkers* is in the root folder too. In this group datasets are named by the number of an iteration.

- generate_excitation - function used to generate an excitations: the input is a given determinant, the output is the target excited determinant, its selection probability, and the value of the matrix element of $\hat{H}$ between input and output determinants.
- compute_energy - computation of $\langle D_i|\hat{H}|D_0\rangle$.
- zero_excitation_matrix_element - computation of $\langle D_0|\hat{H}|D_0\rangle$.

2.6. The ElectronGas class

This class implements the model of electron gas described in 5.1.. Its attributes are:

- a - the size of a unit cell in the direct space, lattice constant, assuming a cube.
- N_{el} - number of electrons in the system.
- N_{k} - number of lattice points on a single axis in the reciprocal space.
- Potential factor U - multiplication constant increasing/decreasing the interaction strength, see Eq. (5.7).
- Kinetic factor T - multiplication constant scaling the kinetic energy, see Eq. (5.7).

2.7. The Matrix class

This class implements the *Matrix* model from Sec. 5.2., with the following attributes:

- Input type - Files with Hamiltonian matrices can be of different types, here the type of the file is needed to be specified.
- Input name - Name of the file containing the Hamiltonian matrix.

- Size - dimension of Hamiltonian matrix.

Currently, two input types are supported. The first type is a standard matrix format where all matrix elements $\langle D_i|\hat{H}|D_j\rangle$ are provided. The second type is used for the model of quantum dots, where the Hamiltonian matrix has to be constructed from two files containing direct and exchange interaction of particles in determinants.

2.8. The ModelFactory class

Design patterns are an advanced concept of the C++ language, which allow the development of standard use cases. For instance, we wish to be able to specify the model and parameters of a simulation at runtime, as opposed to doing it at compilation time. The *Factory pattern* [35] helps us achieve this, and is used in the *ModelFactory* class. Here, the *Model* class is overridden by the *ElectronGas* or *Matrix* class, as specified by the user at runtime.

3. Configuration file

A sample INI configuration file is provided here:

Listing 7.1: Example of a configuration file

```
[Model]
model=Matrix

[Cycle]
generate_hamiltonian=0
N_runs=1000000000
adjust_shift_each=1
rotate_each=1
walkers_needed_for_adjusting_shift=1000000000
walkers_needed_for_rotation=1000000000

[Population]
is_complex=1
N_start_walkers=10
R_shift=0.0
I_shift=0.0
dumping_parameter=0.4e-1
iteration_save_interval=1
timestep=7.0e-1

[ElectronGas]
a=10
N_el=2
N_k=4
potential_factor=2.5e1
```

```
kinetic_factor=2.0e1

[Matrix]
input_type=1
input_name=A
size=27
```

The configuration file is divided into sections, each separated by a label between square brackets. A total of four sections are needed for a simulation to be run: Model, Cycle, Population and one of the defined models. For example in the example configuration file 7.1 the electron gas section would be ignored, because the first parameter specified in the configuration file is the type of the model. It indicates which model will be run and selects the appropriate section of parameters in order to instantiate this specific model.

4. Implementing a new model

The code is designed in such a way that adding a new model to the framework is straightforward. The following things need to be implemented in order to manage a new model:

- Write a new class with following conditions
  - Has different name than current classes.
  - Uses a bitset to express a determinant.
  - Functions *generate_excitation*, *compute_energy* and *zero_excitation_matrix_element* are defined.
- Add this class as an option to the *ModelFactory*.

It is reasonable to expect that a new model would need new parameters. In such case, a new ad hoc section can be created in the configuration file, similar to the sections for *ElectronGas* and *Matrix* models (see 7.1).

Chapter 8

Conclusion

The goal of this work was to develop an implementation of the FCIQMC algorithm from scratch, and use it to solve a multi-particle problem. The FCIQMC algorithm has been implemented in C++, using modern source control tools (hosted by Bitbucket). Our main source for this work was Ref. [19]. The code is portable, and was used on various personal workstations, as well as on the Czech academic computation grid Metacentrum. Two models are currently implemented, the electron gas and the Matrix model.

In this thesis, we have described the FCIQMC approach in detail, including some related quantum chemical methods, on which FCIQMC is based. We have also described the implemented models, and emphasized the critical role of the step where the selection probability is calculated.

The ground state of the electron gas with two electrons has been successfully reproduced by this new method, using different numbers of lattice points in reciprocal space. We have demonstrated that FCIQMC can be used to estimate the ground state and its energy, without the storing the full Hamiltonian matrix. The obtained results were successfully compared with the results of exact diagonalization. The homogeneous electron gas was studied using three different methods: exact diagonalization, FCIQMC without knowledge of the full Hamiltonian matrix and FCIQMC based on the full Hamiltonian matrix.

The emergence of a plateau in the population dynamics was studied, with no success so far. The energy of the ground state was computed successfully, using the standard projection technique and also the independent energy shift variable S. Finally, we have presented the results of the Matrix model, in which the amplitudes of the CI coefficients can be complex. Work on this feature is ongoing.

1. Further work

The list of improvements and new features to implement is hosted on Bitbucket *Issues*: from the programming point of view, it is desirable to implement MPI parallelization. From the point of view of physics, our study barely scratched the surface of the homogeneous electron gas model. In order to tackle more interesting problem, it will be necessary to implement a more general way to compute probabilities for this model, so that more electrons are supported. It is interesting to note that in this respect, going from two to three particles is a very difficult step, but we expect that going from three particles to higher number will not be very challenging, once three particles are supported.

More models could be implemented, such as the Hubbard model, or matrix models

which are stored on disk, but not in RAM.

2. Summary

Code development:

- FCIQMC implemented from scratch in C++
- Input from INI configuration file
- Data saved in HDF5 format
- Data processing in Python, Jupyter notebooks
- Additional models easy to implement
- Portable (works on Metacentrum)
- Energy shift implemented

Study of physical systems:

- Ground state of the homogeneous electron gas composed of two electrons was successfully computed using three different methods (projection technique, energy shift variable, matrix).
- Number of lattice points in homogeneous electron gas model can be set arbitrarily.
- FCIQMC effect of plateau was not observed.

Further work:

- Complex CI coefficients
- General computation of probabilities for electron gas
- Parallelization
- More models
- Initiator
- Non-integer weights

Appendix A

Graphical example of the algorithm

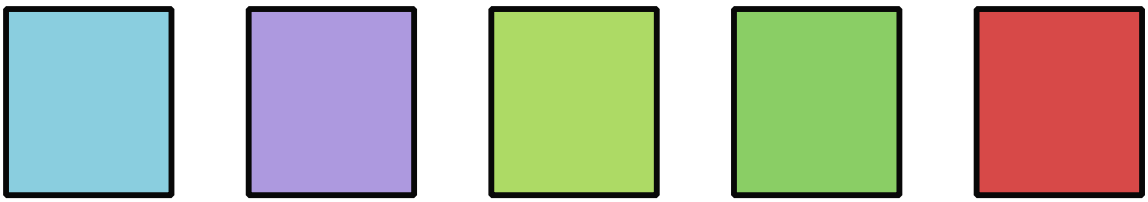
The algorithm for the dynamics of the population can be illustrated graphically. The phase space, represented by the basis of Slater determinants, is represented by a horizontal line in Fig. A.2. On this horizontal line, three ticks represent the three allowed Slater determinants. Walkers are represented by colored squares. The meaning of the colors is described in Fig. A.1. Walkers are blue by default. When a walker is selected during the algorithm its color is changed to violet. When a new walker is spawned from currently selected walker, light green is used for it. If it was spawned in the current timestep, but not from currently selected violet walker, dark green is used instead. Finally, walkers which die are red. The sign of walkers is determined by their position under (negative sign) or above (positive sign) the horizontal line.

The example proceeds in this simplified phase space consisting of only three Slater determinants. We do not specify the starting determinant, as we will not compute the energy of the system. We also use a population which consists only of four walkers at the initial timestep. The walkers are spread through the configuration space for the sake of clarity. The starting state is shown in Fig. A.2.

The first action is the spawning step. We iterate through all walkers, and each walker is given a chance to spawn a new walker on a different Slater determinant, which has to be determined. The spawning step is represented in Figs. A.3, A.4, A.5 and A.6.

The first walker from the population of parent walkers is selected and, is given a chance to spawn a new walker in Fig. A.3.

In Fig. A.3, a successful spawning event is shown. A different target determinant is determined: in Fig. A.3, the third determinant is selected from the first one. The



(a) Walker (b) Selected (c) Child (d) Child (e) Dying

Figure A.1: The meaning of walker's colors. We generally use blue A.1a for all walkers. For new walkers we use dark green A.1d. If the walker is currently selected, we highlight it with violet A.1b. If a new walker is spawned from the currently selected violet walker we use light green A.1c in the current figure and dark green A.1d furthermore. If a walker dies, we use the red color A.1e.

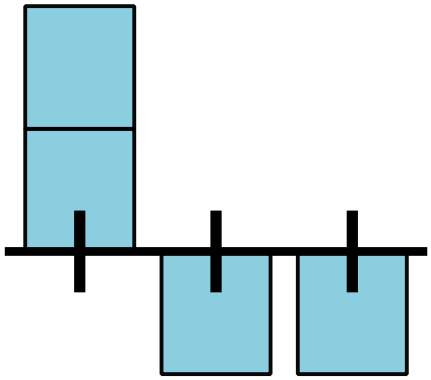


Figure A.2: The initial population - The configuration space is represented by a line consisting of three Slater determinants represented by three ticks. There are four walkers represented by squares at the start of the timestep.

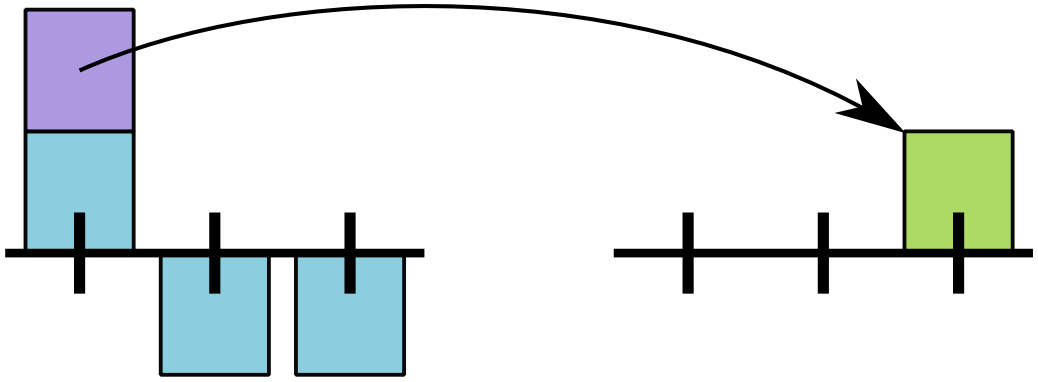


Figure A.3: Spawning step - A first walker (violet) from the parent population (on the left) located on the first determinant selects a third determinant to spawn on. The spawning event is successful and the new walker (light green) is added to the population of child walkers (on the right).

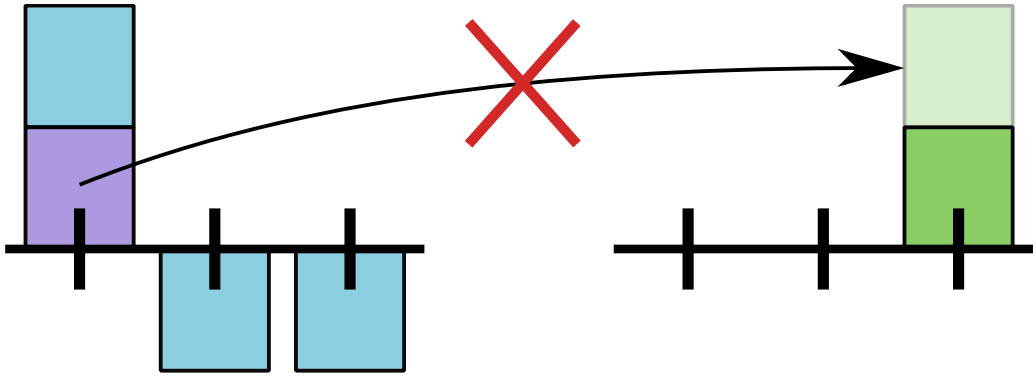


Figure A.4: Spawning step - A second walker is selected and tries to spawn a walker on a third determinant. The spawning event is unsuccessful and no walker is added to the child population.

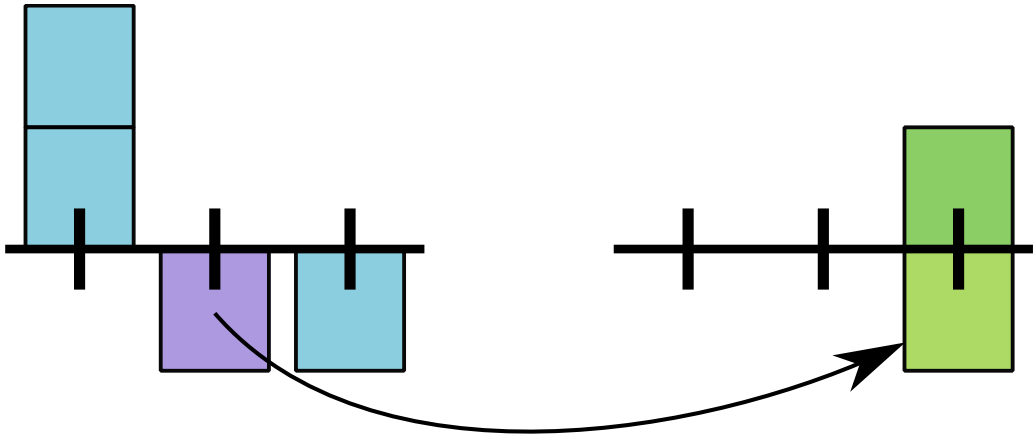


Figure A.5: Spawning step - The third walker located on the second determinant chooses the third determinant as a spawning place. It successfully spawns a new walker there.

matrix element $\langle D_1|\hat{H}|D_3\rangle$ is computed, and the probability of spawning is evaluated using Eq. (4.11). Here we assume that the spawning attempt is successful and a new walker is added to the population of children walkers.

The sign of the matrix element $\langle D_1|\hat{H}|D_3\rangle$ is indicated in Fig. A.3 as well. If the matrix element is positive (negative), the child walker has a sign opposite (equal) to that of the parent. In Fig. A.4 we show an unsuccessful spawning event from the second walker.

Figs. A.5 and A.6, show the spawning attempts of the remaining two walkers. Both events are successful, so that new walkers are added to the population of child walkers. We can notice that the matrix elements $\langle D_2|\hat{H}|D_3\rangle$ and $\langle D_3|\hat{H}|D_1\rangle$ are negative, so that the child walkers have the same sign as their parents.

The death/cloning step follows. Individual death events are shown in Figs. A.7, A.8, A.9 and A.10. In the death/cloning step we iterate over the population of parent walkers. Child walkers created in the spawning step are not considered here. The diagonal matrix element $\langle D_i|\hat{H}|D_i\rangle$ and the probability of death are evaluated for

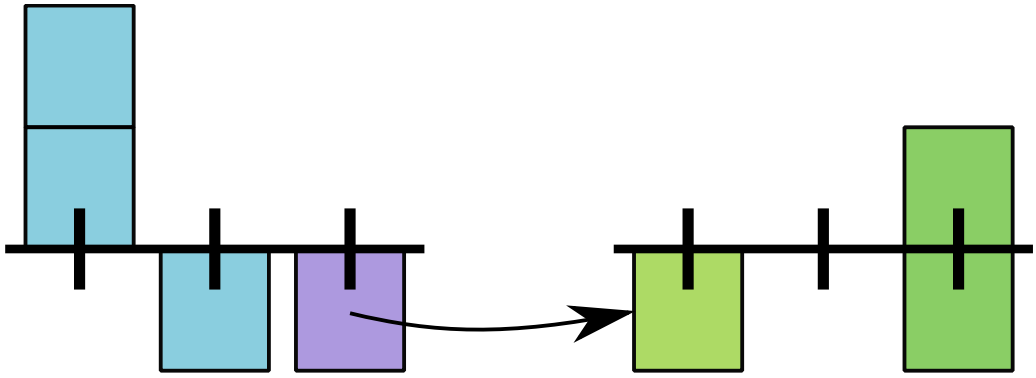


Figure A.6: Spawning step - The third walker located on the second determinant chooses the third determinant as a spawning place. It successfully spawns a new walker there.

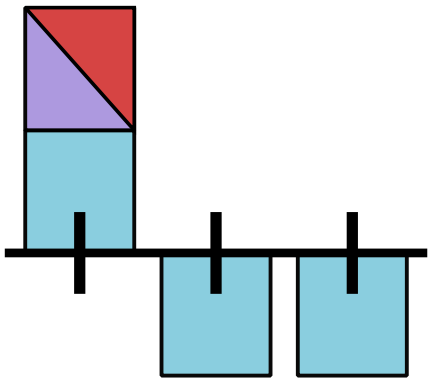


Figure A.7: Death/cloning step - The first walker is selected (marked with violet). The death/cloning event is successful and it dies (it is marked with red too). It is immediately removed from the simulation.

each walker according to the (see Eq. (4.14)). It may happen that the probability of death is negative. In this case walkers are cloned rather than die, with this same probability. In the illustrative figures, in the case where a walker dies, their numbering from the previous figures is conserved.

Figs. A.7 and A.8 are representative of the death event for two walkers on the first determinant. The first walker dies and is removed from the simulation immediately. The second walker does not die and remains in the population.

In Fig. A.9, the death/cloning step is shown for the third walker. The event is successful but the walker has positive diagonal matrix element $\langle D_2|\hat{H}|D_2\rangle$ and instead of dying, it is cloned and a new walker with the same sign is added to the population of child walkers.

At the end of the death/cloning step, the last walker is selected and its death/cloning event is not successful.

When the spawning and the death/cloning steps are finished, we proceed to the annihilation and merging steps. We present three figures showing these steps, A.11, A.12 and A.13.

The annihilation of the child population is shown in Fig. A.11, while Fig. A.12

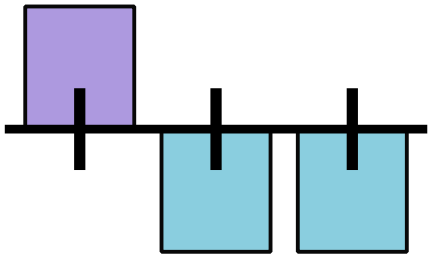


Figure A.8: Death/cloning step - The second walker is selected for the death/cloning event. It is unsuccessful and walker remains in the population of parents.

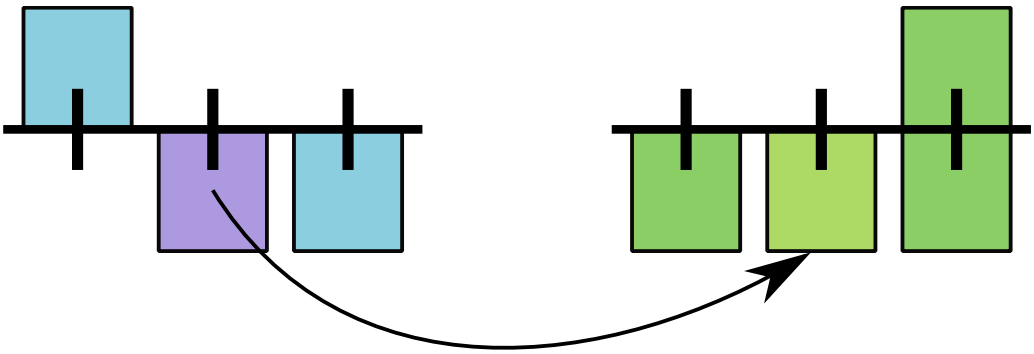


Figure A.9: Death/cloning step - The third walker is selected and its death/cloning event is successful. Its diagonal matrix element is negative so it clones itself into the child population (on the right) instead of dying.

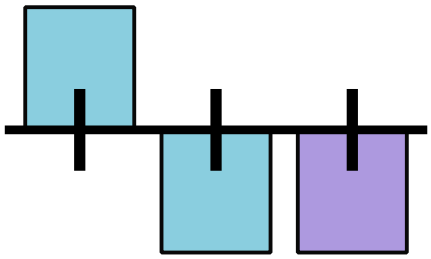


Figure A.10: Death/cloning step - The last walker is selected, its death/cloning event is unsuccessful and it remains in the population.

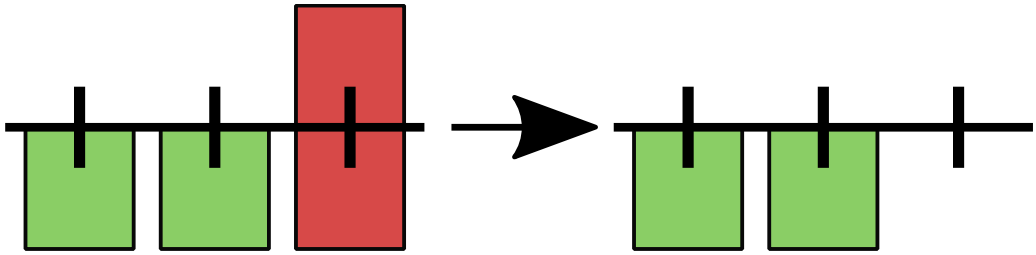


Figure A.11: Annihilation step - The annihilation step is run over the population of child walkers on the left side of the figure. On the third determinant, two walkers of opposite signs are present. They are both removed from the simulation and we can see the population of child walkers after the annihilation step on the right side of the figure.

illustrates the merge operation between the annihilated child population and the population of parents. When old and new populations are merged together all become parents so they can participate in spawning and death/cloning steps in the future timesteps. In Fig. A.13 new merged population is annihilated because at the start of a new timestep walkers on every determinant have to have a uniform sign. As a conclusion, Fig. A.14 recapitulates the complete timestep. We can see that the total population was decreased during this specific timestep.

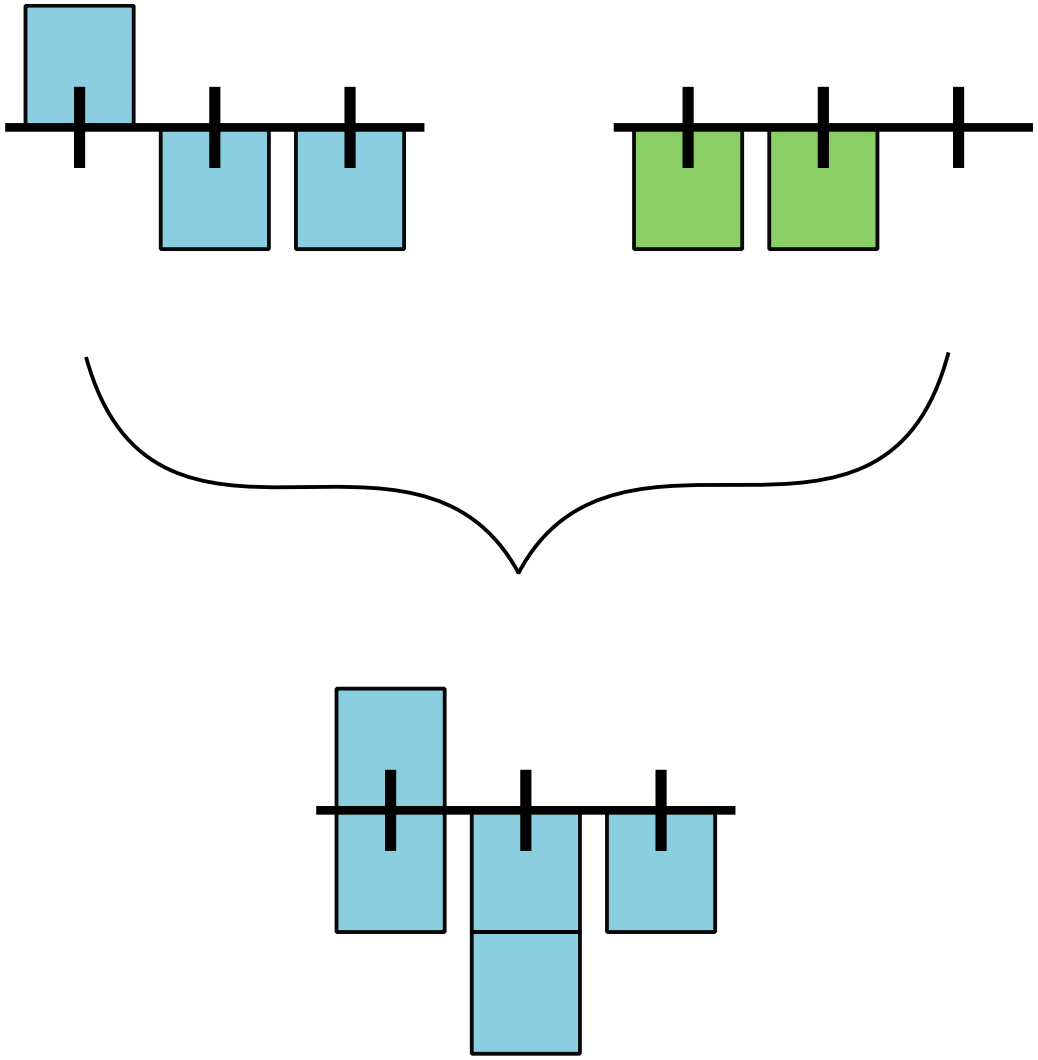


Figure A.12: Merging step - Old population of parent walkers (blue) is merged with the new population of child walkers (green). After the merge all walkers become parent walkers (blue).

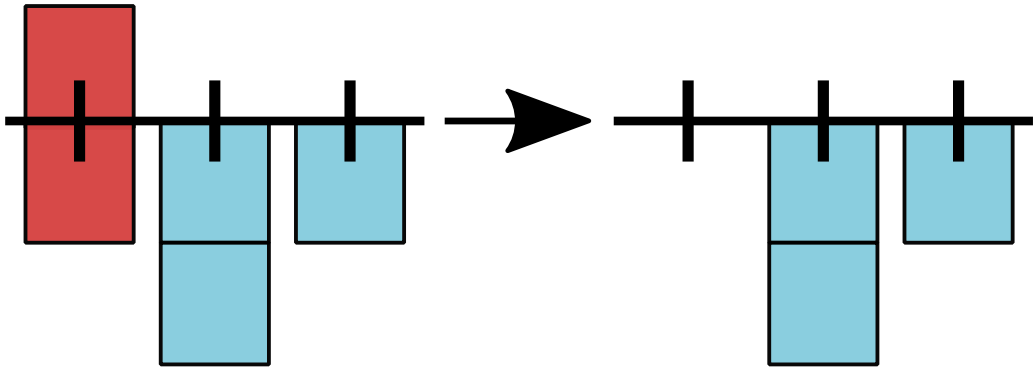


Figure A.13: Annihilation step - Merged population is annihilated. On the first determinant there are two walkers with different sign, one from old and one from new population. Both are annihilated and determinants of the final population (on the right) have populations of a uniform sign.

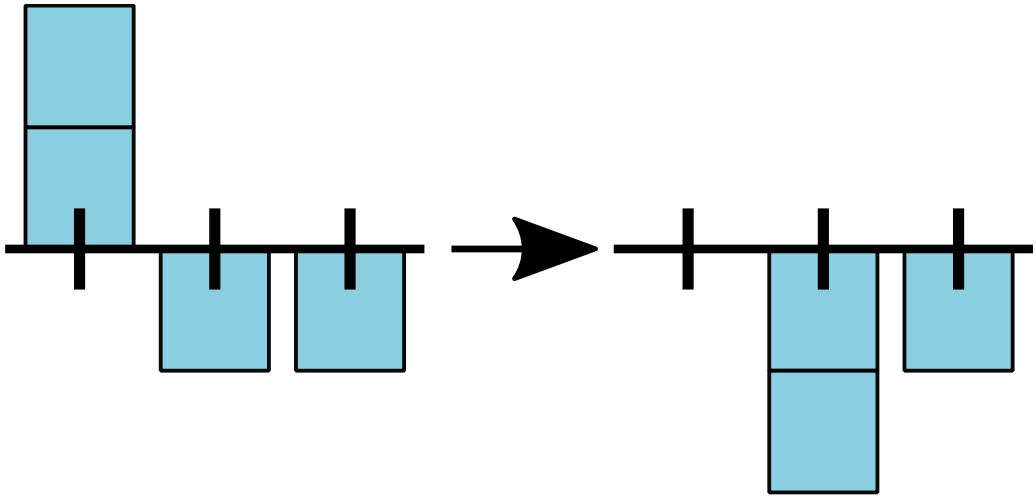


Figure A.14: Recapitulation of the timestep - The population on the first determinant died or was annihilated. The population on the second determinant increased by a single walker and the population on the third determinant did not change.

Bibliography

1. Zettili, N. *Quantum mechanics: concepts and applications* 2nd ed. ISBN: 978-0-470-02678-6 (John Wiley & Sons, 2009).
2. Ballentine, L. E. *Quantum Mechanics: A Modern Development* 2nd ed. ISBN: 981-02-4105-4 (World Scientific Publishing Company, 1998).
3. Dirac, P. A. M. A new notation for quantum mechanics. *Mathematical Proceedings of the Cambridge Philosophical Society* **35,** 416–418 (1939).
4. Wick, G. C. Properties of Bethe-Salpeter Wave Functions. *Physical Review* **96,** 1124–1134 (Nov. 1954).
5. Levine, I. N., Busch, D. H. & Shull, H. *Quantum Chemistry* 7th ed. ISBN: 978-0-321-80345-0 (Pearson, 2013).
6. Hartree, D. R. The Wave Mechanics of an Atom with a Non-Coulomb Central Field. Part I. Theory and Methods. *Mathematical Proceedings of the Cambridge Philosophical Society* **24,** 89–110 (1928).
7. Gross, E. K. U. *Many-Particle Theory* ISBN: 0-7503-0155-4 (CRC Press, 1991).
8. Szabo, A. & Ostlund, N. S. *Modern quantum chemistry: introduction to advanced electronic structure theory* ISBN: 0-486-69186-1 (General Publishing Company, 1996).
9. McWeeny, R. *Methods of Molecular Quantum Mechanics* 573. ISBN: 978-0124865525 (Academic Press, 1992).
10. Shavitt, I. & Bartlett, R. J. *Many-Body Methods in Chemistry and Physics: MBPT and Coupled-Cluster Theory* ISBN: 978-0-521-81832-2 (Cambridge University Press, 2009).
11. Lowe, J. P. & Peterson, K. A. *Quantum Chemistry* 3rd ed. ISBN: 978-0-12-457551-6 (Academic Press, 2005).
12. Condon, E. The theory of complex spectra. *Physical Review* **36,** 1121 (1930).
13. Landau, D. P. & Binder, K. *A Guide to Monte Carlo Simulations in Statistical Physics* 4th ed. ISBN: 978-1-107-07402-6 (Cambridge University Press, 2015).
14. Krauth, W. *Statistical mechanics: Algorithms and Computations* ISBN: 978-0-19-851536-4 (Oxford University Press, 2006).
15. Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. & Teller, E. Equation of state calculations by fast computing machines. *The journal of chemical physics* **21,** 1087–1092 (1953).
16. Anderson, J. B. A random-walk simulation of the Schrödinger equation: H+3. *The Journal of Chemical Physics* **63,** 1499–1503 (1975).

17. Foulkes, W. M. C., Mitas, L., Needs, R. J. & Rajagopal, G. Quantum Monte Carlo simulations of solids. *Rev. Mod. Phys.* **73,** 33–83 (1 Jan. 2001).

18. Spencer, J. S., Blunt, N. S. & Foulkes, W. M. The sign problem and population dynamics in the full configuration interaction quantum Monte Carlo method. *The Journal of Chemical Physics* **136,** 054110 (2012).

19. Booth, G. H., Thom, A. J. W. & Alavi, A. Fermion Monte Carlo without fixed nodes: A game of life, death, and annihilation in Slater determinant space. *The Journal of Chemical Physics* **131,** 054106 (2009).

20. Umrigar, C. J., Nightingale, M. P. & Runge, K. J. A diffusion Monte Carlo algorithm with very small time-step errors. *The Journal of Chemical Physics* **99,** 2865–2890 (1993).

21. Booth, G. H., Grüneis, A., Kresse, G. & Alavi, A. Towards an exact description of electronic wavefunctions in real solids. *Nature* **493,** 365–370 (2013).

22. Pavarini, E., Koch, E., van den Brink, J. & Sawatzky, G. *Quantum Materials: Experiments and Theory* 420 p. ISBN: 978-3-95806-159-0. `http://juser.fz-juelich.de/record/819465` (Forschungszentrum Jülich, Jülich, 12th Sept. 2016).

23. Alavi, A. *Introduction to Full CI Quantum Monte Carlo (with applications to the Hubbard Model)* Presentation. 2015. `https://www.cond-mat.de/events/correl16/talks/alavi.pdf`.

24. Cleland, D., Booth, G. H. & Alavi, A. Communications: Survival of the fittest: Accelerating convergence in full configuration-interaction quantum Monte Carlo. *The Journal of Chemical Physics* **132,** 041103 (2010).

25. Blunt, N. S. An efficient and accurate perturbative correction to initiator full configuration interaction quantum Monte Carlo. *ArXiv e-prints.* arXiv: `1804.09528 [physics.chem-ph]` (Apr. 2018).

26. Shepherd, J. J., Scuseria, G. E. & Spencer, J. S. Sign problem in full configuration interaction quantum Monte Carlo: Linear and sublinear representation regimes for the exact wave function. *Phys. Rev. B* **90,** 155130 (15 Oct. 2014).

27. Booth, G. & Alavi, A. *NECI* `https://github.com/ghb24/NECI_STABLE`.

28. Sun, Q. *et al.* PySCF: the Python-based simulations of chemistry framework. *Wiley Interdisciplinary Reviews: Computational Molecular Science* **8** (2018).

29. Leikanger, K. R. *Full Configuration Interaction Monte Carlo Studies of Quantum Dots* MA thesis (Faculty of Mathematics and Natural Sciences, Department of Physics, Universit of Oslo, June 2013).

30. Olsen, V. K. B. *Full Configuration Interaction Simulation of Quantum Dots* MA thesis (Faculty of Mathematics and Natural Sciences, Department of Physics, Universit of Oslo, Dec. 2012).

31. Giuliani, G. & Vignale, G. *Quantum Theory of the Electron Liquid* ISBN: 978-0-511-82112-4 (Cambridge University Press, 2005).

32. Shepherd, J. J., Booth, G., Grüneis, A. & Alavi, A. Full configuration interaction perspective on the homogeneous electron gas. *Phys. Rev. B* **85,** 081103 (8 Feb. 2012).

33. Shepherd, J. J., Booth, G. H. & Alavi, A. Investigation of the full configuration interaction quantum Monte Carlo method using homogeneous electron gas models. *The Journal of Chemical Physics* **136,** 244101 (2012).

34. The HDF Group. *Hierarchical Data Format, version 5* `http://www.hdfgroup.org/HDF5/`.

35. Helm, R., Johnson, R. E., Gamma, E. & Vlissides, J. *Design patterns: elements of reusable object-oriented software* ISBN: 0-201-63361-2 (Addison-Wesley, 2009).