Ground-state wave-function optimization for a quantum system of N bosons

Author
Jordi Sala Juárez

Director
Joaquim Casulleras Ambros
Physics department

Codirector
Ferran Mazzanti Castrillejo
Physics department

Defense date 29/06/2020

Degree final project for the
Bachelor Degree in Informatics Engineering

Computer Science
Facultat d'Informàtica de Barcelona
Universitat Politecnica de Catalunya
Spain, Barcelona

Dedication

I would like to thank both Joaquim Casulleras and Ferran Mazzanti for all the effort they have given throughout the development of this project, and for the immense dedication they spent on me, helping me understand all the all physical and mathematical principles needed for the development of the tool. I have really enjoyed the time spend with them and I would like to congratulate them for their fantastic job as the directors and co-directors of this project.

Resum

Aquest treball s'emmarca en el camp de l'estudi i simulació del comportament de sistemes quàntics complexos, les propietats dels quals vénen dictades per les lleis de la física quàntica. Aquestes juguen un paper fonamental en una varietat de sistemes físics, des de matèria condensada, sistemes superfluids i superconductors, líquids quàntics com l'He a molt baixa temperatura o trampes d'àtoms freds formant condensats de Bose - Einstein.

En l'estudi de tots aquests sistemes han adquirit un paper de gran rellevància les tècniques de Quantum Monte Carlo, que es basen en la simulació directa de les partícules que componen el sistema, que es regeixen universalment per l'equació de Schrödinger, que descriu exactament, excepte per efectes relativistes, el comportament quàntic de la matèria.

En aquest projecte ens dedicarem a crear una eina que apliqui mètodes estocàstics actuals, i permeti comparar-los amb la nostra implementació d'un mètode molt recent que aconsegueix extreure informació molt precisa del sistema basant-se en postulats de la mecànica quàntica.

Resumen

Este trabajo se enmarca en el campo del estudio y simulación del comportamiento de sistemas cuánticos complejos, cuyas propiedades vienen dictadas por las leyes de la física cuántica. Estas juegan un papel fundamental en una variedad de sistemas físicos, desde materia condensada, sistemas superfluidos y superconductores, líquidos cuánticos como Helio a muy baja temperatura o trampas de átomos fríos formando condensados de Bose - Einstein.

En el estudio de todos estos sistemas han adquirido un papel de gran relevancia las técnicas de Quantum Monte Carlo, que se basan en la simulación directa de las partículas que componen el sistema, que se rigen universalmente por la ecuación de Schrödinger, que describe exactamente, excepto por efectos relativistas, el comportamiento cuántico de la materia.

En este proyecto nos dedicaremos a crear una herramienta que aplique métodos estocásticos actuales, y permita compararlos con nuestra implementación de un método muy reciente que consigue extraer información muy precisa del sistema basándose en postulados de la mecánica cuántica.

Abstract

This work is aimed to a contribution to the study and simulation of the behaviour of complex quantum systems, whose properties are dictated by laws of quantum physics. These play a fundamental role in a variety of physical systems, ranging from condensed matter, superfluid systems and superconductors, quantum liquids like Helium at very low temperatures, or traps of cold atoms forming Bose - Einstein condensates.

In the description of all these systems, stochastic methods known as Quantum Monte Carlo methods have acquired a role of great relevance. They rely on a direct simulation of the particles that constitute the system, which are universally governed by the Schrödinger's equation which, apart from relativistic effects, describes exactly the behaviour of quantum matter.

In this project we will develop a tool to apply current quantum Monte Carlo methods and compare them with our implementation of a very recent method which, based on a direct stochastic modelling of the Schrödinger equation for the evolution of the quantum state, in accordance to the postulates of quantum mechanics, extracts precise information from the system.

Contents

Chapter 1

Introduction and Context

This project will be developed in collaboration with people from the Barcelona Quantum Monte Carlo group, in the Physics department of the Barcelona School of Informatics (FIB).

The objective of this project is to build a tool, based on Monte Carlo propagation in imaginary time, that helps to improve the variational description of a quantum many-body system by direct optimization of its struture. A tool capable of analyzing the description, behaviour and properties of quantum many-body systems at very low temperature.

Evaluating their properties requires to deal with the many-body Schrödinger equation. Finding an analytical exact solution is impossible, and one must rely on numerical computation techniques such as Quantum Monte Carlo methods in order to gain some insight into the properties of those systems.

Quantum Monte Carlo methods can be classified in two categories, exact methods (which rely on huge amounts of computing resources, but do not offer an exact analytical insight, and can be worked out only in some specific cases), and the Variational Quantum Monte Carlo method, whose main characteristic is that it requires from the user the input of an approximate wave function, from which it is able to produce reliable predictions concerning the properties of the system.

It must be stated that variational methods follow the principle of zero variance for the energy, meaning that as we approach the ground-state wave function, the local energy converges to the ground-state energy and hence becomes independent of the configuration of the particles, ,and therefore it has zero variance. This implies that a lower bound for variational methods exists and can be reached.

Recently, a new line of progress has been proposed which could provide the best of both approaches: the Time-dependent variational Monte Carlo method [1] originally derived in [2, 3],which bears the promise to be able to get arbitrarily close to the

analytical exact, true ground-state of the system from first principles, for any system whose Hamiltonian is known (the Hamiltonian is a mathematical function describing the full characteristics of the physical system, including the atomic interactions).

Steps in that direction are most interesting, but they involve extremely complex and computationally demanding calculations. Any attempt in this line must provide the ability to link highly complex symbolic calculations entering in the ground-state wave function, with hundreds of parameters whose evolution is determined within the framework of the recently proposed time dependent Variational Quantum Monte Carlo method.

The project is aimed at continuing this line of research and provide a tool capable of analyzing the ground-state properties of a many body system and capable of defining the physical descriptors of the functional form of the wave function used in the simulation to validate the correctness of the descriptors.

The software would be used in the first place by the research group in many-body quantum systems at the Physics department, and it could make a significant contribution to the knowledge of the behaviour of quantum matter, which undoubtedly will transform, in the upcoming decades, the materials and the world we live in. Furthermore, this project may open new lines of research.

Chapter 2

Justification

In recent years, in the field of low temperature quantum systems simulation, the development and refinement of stochastic methods which go beyond the prediction capacity of the Variational Monte Carlo method has been an active area. These refinements have allowed to obtain exact (partial) information of the system, based on a stochastic modelling of the Schrodinger equation.

An even more ambitious idea would be to achieve, from a direct stochastic modeling of the Schrodinger equation, the improvement of the wave function itself using the time-dependent Variational Monte Carlo [1], with the aim of extracting the exact wave function with arbitrary accuracy. This would open up the ability to extract accurate information of all the properties of the system, in accordance with the postulates of quantum mechanics.

Although in traditional VMC it is already a common practice to look for, as far as possible, a good test function, this idea that has recently come out to achieve a complete description of the exact wave function requires to develop and systematize all the Variational Monte Carlo procedures for functions of arbitrary complexity.

Even if the conceptual procedure is clearly understood, the manual development of programs that implement the convergence towards the exact wave-function is tremendously hard.

Ideally, a perfect solution would be to have a mechanism with the ability to automate the whole set of tasks required for a such a research program. A tool capable of implementing this ideas does not exist to date, and the work of designing and implementing it, taking elements from theoretical physics to software development, is an interdisciplinary task which requires the confluence of knowledge of the two fields.

Defining the requirements of such automation and building a solution to these requirements is the goal of our project.

Chapter 3

Scope

3.1 Objectives and Sub-objectives

The project will provide solutions to the different requirements that the group needs to satisfy:

(1) Create a graphical interface that lets the user define the problem, specifying the Hamiltonian of the system, as well as:

- The number of atoms used in the simulation.
- The operators used in the Variatonal Time-dependent formalism.

(2) The interface lets the user specify the functional space of the wave functions to consider:

- One-body functions.
- Two-body functions for each pair of atoms.

(3) From the specified points 1 and 2, develop a Variational program for the simulation of the system by means of Quantum Monte Carlo.

(1) Analyse values obtained from the simulation: variational energy, local energy, standard error.

(2) Compute in real time (while performing the simulation) the evolution of the wave function (the parameters specified at point (2)), given by the Variatonal Time-dependent formalism.

(3) Compare with the results of an optimization using the method of Simulated Annealing.

(4) Graphical representation of the energy dependence of the system based on the descriptor parameters that define the functional space and the considered wave functions.

(5) Graphical representation of the Simulated annealing performance throughout a run on Variational Quantum Monte Carlo.

(4) The program has to have a high performance in computing for a given architecture, given the long compute times it can have for simulations with a large amount of atoms.

3.2 Identification and definition of potential obstacles and risks

No potential obstacles are likely to appear, other than the fact that the time-dependent Variational method to be explored could suffer from unexpected pitfalls. In that case, the product developed in this project would likely be used to characterize them and to point out the root causes.

Chapter 4

Methodology

4.1 Agile software development

This methodology is based on short iterations that last from one to four weeks with objectives for each iteration that are achievable in this short frame of time, making it easier to adapt to possible difficulties or changes of criteria throughout the development.

In our case we have weekly or biweekly meetings with Joaquim Casulleras Ambros and Ferran Mazzanti Castrillejo in the UPC FIB B4 building, these meetings mostly address doubts related to computational physics and features that are needed for the graphical interface, followed up by the definition of the next week objectives based on feedback of the results. Most of the development of the tool is done at home and meetings days are used to get a thorough analysis of the results of the simulations.

4.2 Test driven development

This methodology is used to prove the correctness of a software based on its granularity. Pieces that compose the software are tested (checked for known cases that their outputs are right) in order to prove the correctness of the hole system.

Although finding an analytical exact solution is impossible for a large amount of atoms, and one must rely on techniques such as Quantum Monte Carlo methods that may not give exact solutions, some experiments like the harmonic oscillator or

the hydrogen atom have known analytical exact solutions, therefore we have we have a way to prove that or tool follows the anallitically correct and known solutions.

4.3 Development Tools

- **Version control:** a Github repository has been created to hold the project and Git is used as the version control tool.
- **Diagrams:** Gantt project is used in the making of the Gantt diagram for the project planning.
- **Task management:** a trello is used to orginize the main tasks of the project.

4.4 Technologies

4.4.1 Python

Given the fast prototyping and needs of change this thesis needs, we have decided to use tools meant to be used for fast development and testing purposes. For this reason, our best choice ia Python and the Jupyter Notebook environment. Python has a strong standard library and all our needs are covered by either the standard library or the vast amount of third party libraries. Jupyter provides the perfect balance between code and visualization of data we were searching for, and given all these tools are open source there are no drawbacks if the tool were to be distributed or made public.

Jupyter Lab

Like Jupyter Notebook, Jupyter Lab is a web-based interactive enviroment that is mostly known in the development of python applications although it has other kernels for languages like Julia, R and Haskell. Jupyter Lab was thought as a tool that eases the development of scientific applications by giving a fast prototyping enviroment with great and interactive visualization tools, profiling and debugging capabilities for a great REPL. Many people think of it as and IDE for python.

Voila

One of the problem Jupyter notebooks have is that code is always present so unless you want to release a tool with visible code, the normally notebook code gets ported

to a more presentable format. Voila solves this by rendering the same notebook without code and better and enhanced visuals for the purpose wanted.

Bqplot and Plotly

There are many tools for making plots and interactive widgets in python but Bqplot has a clean, easy and enhanced features to be used in Jupyter Notebooks that other alternatives do not have. Plotly has more of the same, the only difference is that Plotly has a bigger feaute set for customizing interactive plots, bqplot instead is more focused on interaction of other features like tab

For this thesis Bqplot is used for all the interface with the user, the widget for the user input, the widget for defining the hamiltonian, have function terms and variables and the widget for running both the simulated annealing and time-depentent simulations are made with Bqplot. Plotly is used on the interactive plots used to observe values while the simulations are running.

Numba

Numba is an open-source JIT compiler for python that translates a subset of python code into high performance code using llvm. Numba has many features to make code faster like multithreading, vectorization, making ufuncs, GPU-accelerated code or C callbacks.

Performance is key for this thesis, therefore, we need a way to reduce our bottleneck of running everything on the python interpreter, Numba gives us the possibility of still be running python while boosting the performance of the simulation we want to run, to the point it is nearly as fast or faster than a specific Fortran code to be run for that purpose.

Sympy

Sympy is a computer algebra tool that has a huge feature set of expression manipulation, it has everything a normal computer algebra tool would have and more, but one of the main features we are interested in is the set of tools to manipulate expressions and convert the expressions into code from different languages like C, Fortran, Matlab or Julia. This makes Sympy even more appealing as a tool to generate optimal runnable functions independent of the platform where we could run whatever simplification and common sub-expression algorithm we wanted to find the runnable function going our of the standard optimizations a compiler would make to a mathematical expression.

Chapter 5

Project Planning

5.1 Introduction

Given the university premises, this project has a management time of 136h and an estimate development and testing time of 330h. This project starts at 27/01/2020 and its planned to end at 23/05/2020 given that the lecture turns are on 02/06/2020. There is no requirement made by the Barcelona Quantum Simulation group given that we have no client and this project is a proof of concept.

5.2 Task division

The tasks are divided in three groups, management tasks, development tasks and testing tasks.

5.2.1 Project Management

Management tasks are a must to deliver great quality results by means of a good organization. The following task are the tasks we plan to follow for the development of this thesis.

- **MT - weekly or biweekly meetings:** Given the agile methodology we have chosen for this project, weekly or bi-weekly meetings are done to summarize previous work, analyze pitfalls, check correctness of the work and establish future tasks for the next meeting. We also have to take into account the monitoring of the development of the memory for the project. These meetings

take around one or two hours per week, therefore this takes 24 hours given the 12 weeks of this thesis.

- **SD - scope definition:** The scope of the project was defined within the first 3 weeks, during these weeks we defined the domain of these problems, primary goals, possible paths to take once the primary goals are functional, the feasibility of this project for an undergrad without physics background. This has taken 30h.
- **PD - Plan definition:** Once the scope was given, we spent around 10h to organize the resources needed for each of the development tasks, prerequisites of the tasks and the approximate sprint where this task will be developed.
- **DOC - Documentation:** Documentations is intended to be done concurrently throughout the development of the project, based on the new requirements, problems we have to overcome and the findings we do for the proof of concept. There are no requirements other than having a computer and the estimated time, given previous experience on similar documentation, is around 50h.
- **SR - Sustainability Report:** This document determines the consequences our project has for the environment and ,if possible, what is possible four us to reduce the footprint it gives. Our project is mainly based on simulations therefore other than running our program in a cluster, our footprint is near zero. Therefore, the estimate time is around 5h for the Sustainability Report.
- **BG - Budget:** Given this is a research project for a non benefit purpose, without grants and its purpose is a proof of concept of a simulation that doesn't require any non ordinary resources other than a computer and workforce, the estimated time for this task is 5h.

5.2.2 Development phase

The development has been estimated to be done in three sprints with Design and Evaluation phase. From the first sprint we can estimate that the time from both Design and Evaluation phases will be 10 hours. The overall tasks needed to be completed for the development phase are the following:

- **OBOE - One-Body Oscillator Experiment:** design the simulation of the one-body oscillator for the computation of the variational energy for a certain parametrization of the one-body wave function by means of variational quantum montecarlo. Resources needed are scientific research articles and a computer for the simulations. This has taken 20h.
- **MBOE - Many-Body Oscillator Experiment:** extend the previous **OBOE** point to the many-body case.Resources needed are scientific research articles and a computer for the simulations. Given the time spent on **OBOE** this should take 30h.

- **GMBE - General Many-Body Experiments:** extend the **MBOE** to the general case of different interesting many-body interactions. The resources needed are scientific research articles and a computer for the simulations. Given the complexity of this problem we estimate 60h to finish this task.
- **GUIP - Interface for User Input:** design a user interface that lets the user specify the parameters of the simulation like the Hamiltonian of the system or the functional space to which the wave function belongs. The resources needed are a computer. This takes around 10h given the low complexity of this task.
- **VIZ - Visualization search space Tool:** this tool provides real time feedback to the user about the simulation. This helps the user decide whether the simulation is doing well or not, so that the user can stop the execution if needed to start all over again from a different set of parameters. This is intended to run while the simulation is executing so it is not intended to be resource intensive. The resources needed are a computer. This takes around 20h the different models this could be tested on.
- **HYPO - Hyper Parameter Optimization:** Analyse the performance and accuracy of several space search methods for finding the ground-state. Requirements are a computer and scientific research articles about the topic. The expected time 50h given the complexity and different methods we can follow to do this task.

5.2.3 Testing phase

The testing phase will be critical for the analysis of the performance and the simulation to evaluate the end result of the simulation tool. The tasks to be done are the following:

- **DAT - Define Analytical Tests:** To prove the correctness of our simulation tool we have to build tests based on simulations with known analytical solutions. The resources used are scientific books and articles where known experiments are described, and a computer to run the simulations. This should take around 20h.
- **PACS - Performance Analysis and Cluster Support:** We must check whether our simulation tool is using efficiently the available CPU power given the possible complexity of the simulations. The resources used are the Physics department's cluster and GPU, and a laptop. The expected time is around 30h given the multiple methods we will have to analyze and optimize.
- **TM - Test Meetings:** some of the meetings will be devoted to check the correctness of the simulation. For this meetings the presence of both supervisors will be required, as well as a computer for running the simulations. Taking into account we spend weekly 2.5h on meetings for test, the estimated total for all these meetings is 30h.

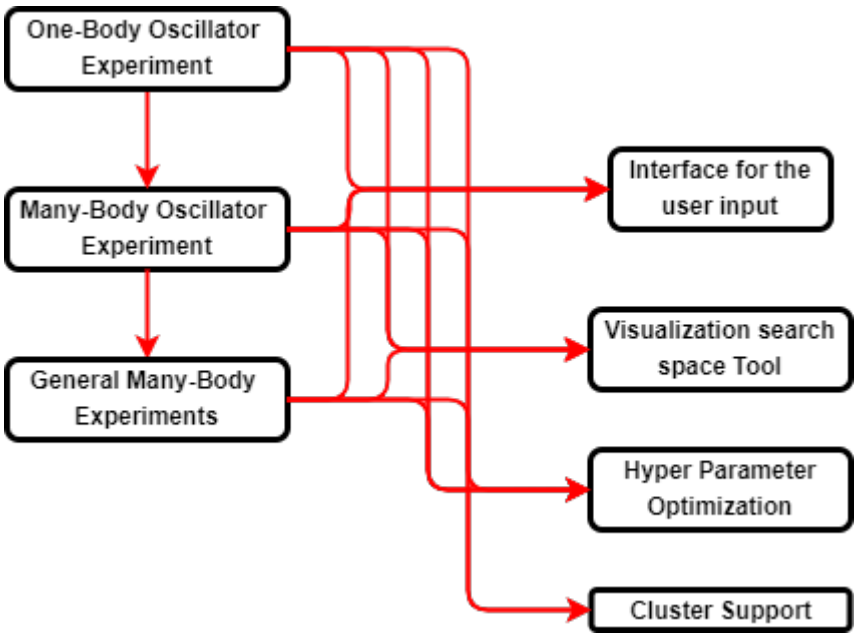


Figure 5.1: Workflow dependencies

Acronym	Task Name	Hours	Prerequisites	Requirements
	Project Management	136		
MT	weekly or biweekly meetings	24		PC, Overleaf
SD	Scope definition	30		PC, Overleaf
PD	Plan definition	7		PC, Overleaf, GanttProject
DOC	Documentation	65		PC, Overleaf, draw.io
SR	Sustainability Report	5		PC, Overleaf
BG	Budget	5		PC, Overleaf
	Project Development	250		
	Sprint 1	70		
DS	Design	10		PC, Overleaf
OBOE	One-Body Oscillator Experiment	20		PC, Visual Studio Code, Git, articles
MBOE	Many-Body Oscillator Experiment	30	OBOE	PC, Visual Studio Code, Git, articles
EV	Evaluation	10		PC, Overleaf
	Sprint 2	90		
DS	Design	10		PC, Overleaf
GMBE	General Many-Body Experiments	60	MBOE	PC, Visual Studio Code, Git, articles
GUIP	Interface for User Input	10	OBOE, MBOE, GMBE	PC, Visual Studio Code, Git
EV	Evaluation	10		PC, Overleaf
	Sprint 3	90		
DS	Design	10		PC, Overleaf
VIZ	Visualization search space Tool	20	OBOE, MBOE, GMBE	PC, Visual Studio Code, Git
HYPO	Hyper Parameter Optimization	50	OBOE, MBOE, GMBE	PC, Visual Studio Code, Git, articles
EV	Evaluation	10		PC, Overleaf
	Testing phase	80		
DAT	Define Analytical Tests	20		PC, Overleaf
PACS	Performance Analysis and Cluster Support	30	OBOE, MBOE, GMBE, HYPO	PC, laptop, Visual Studio Code, Git, Cluster
TM	Test Meetings	30		PC, Overleaf

Table 5.1: Summary of the tasks

5.3 Risk management

Given the purpose of this program, a tool used for simulations inside the physics department that is not intended to put any lives at stake and has no affiliates whose investments could be at risk. There are no social, economic or environmental risks, the only risk possible is the failure of this proof of concept. There are different parts where this experiment could lead to failure, either:

- Numerical methods used do not give enough precision to be able to find the ground truth of the system
- We can't find a method of search space for the parametrization of the wave function that is reliable at giving the ground-state and is feasible to run in nowadays hardware in a short span of time.
- The method of searchspace is realiable but does not generalize for general Many-Body experiments.
- The tool is not expressive enough for the purposes of the user.
- The tool cannot reach the generalized model given the short life span of this project and the complexity of finding the ground-state in the generalized Many-Body experiment has.

In case of failure by some of the previously mentioned pitfalls, the contingency plan is investing more time of research in scientific articles to find feasible alternatives that can adapt to the purpose of this thesis, since most of the pitfalls are related to applying some knowledge from a scientific article to our thesis. The only exception is if we have have a tool that is not expressive enough, this would lead to major changes of the tool structure and design that would take longer than just doing more research.

In case of failure we can always document the current state of the thesis given the importance of such research in this field has, for future lines of research know where the pitfalls are and where should they look at given the results of this thesis.

5.4 Deviations of the Project development schedule

The following are the resulting shedule we have followed compared to the one estimated.

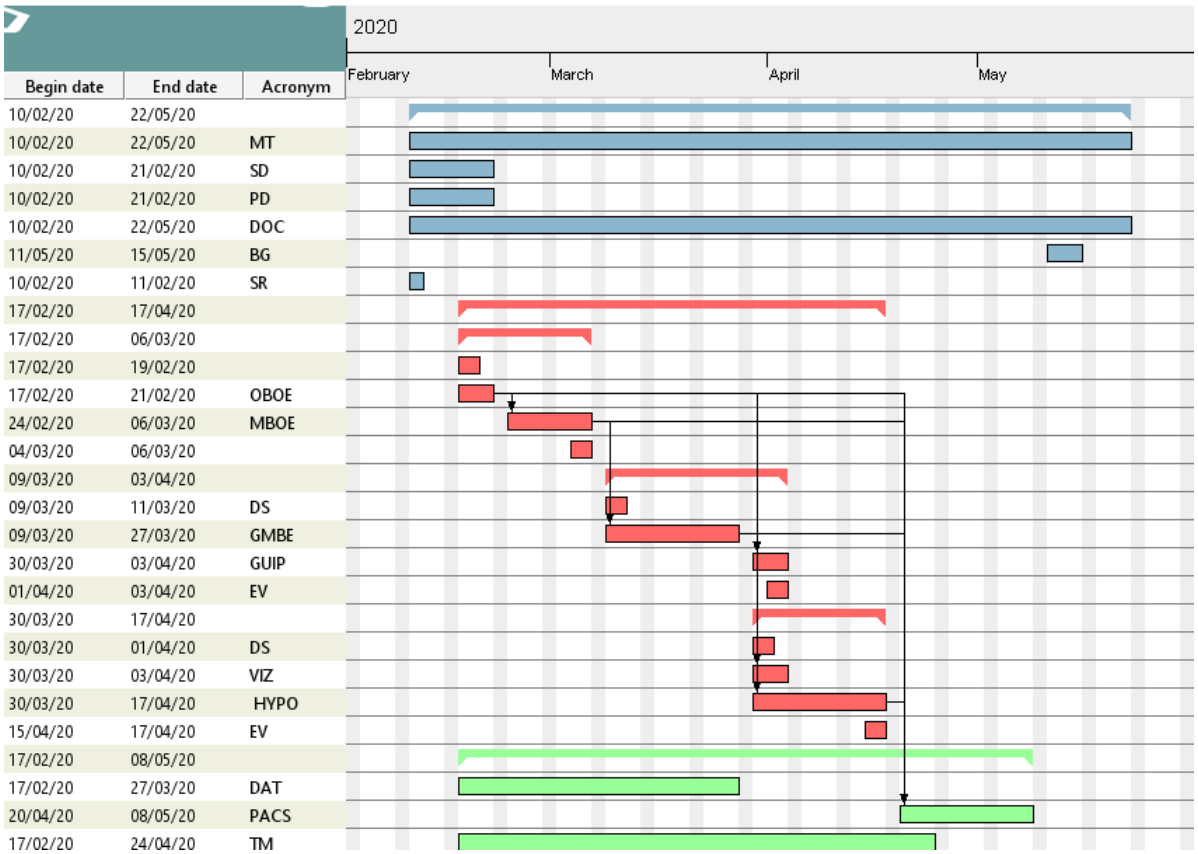


Figure 5.2: Gantt Diagram

Begin date	End date	Name	Acronym	Dur...	Risk
10/02/20	22/05/20	Project Management		136	
10/02/20	22/05/20	Weekly or Biweekly Meetings	MT	24	
10/02/20	21/02/20	Scope Definition	SD	30	
10/02/20	21/02/20	Plan Definition	PD	10	
10/02/20	22/05/20	Documentation	DOC	65	
11/05/20	15/05/20	Budget	BG	5	
10/02/20	11/02/20	Sustainability Report	SR	5	
17/02/20	17/04/20	Project Development		250	
17/02/20	06/03/20	Sprint 1		70	
17/02/20	19/02/20	Design		10	
17/02/20	21/02/20	One-Body Oscillator Experime...	OBOE	20	Low
24/02/20	06/03/20	Many-Body Oscillator Experi...	MBOE	30	Low
04/03/20	06/03/20	Evalutation		10	
09/03/20	03/04/20	Sprint 2		90	
09/03/20	11/03/20	Design	DS	10	
09/03/20	27/03/20	General Many-Body Oscillator...	GMBE	60	High
30/03/20	03/04/20	Interface for User Input	GUIP	10	Low
01/04/20	03/04/20	Evaluation	EV	10	
30/03/20	17/04/20	Sprint 3		90	
30/03/20	01/04/20	Design	DS	10	
30/03/20	03/04/20	Visualization Search Space Tool	VIZ	20	Low
30/03/20	17/04/20	Hyper Parameter Optimization	HYPO	50	Medium
15/04/20	17/04/20	Evaluation	EV	10	
17/02/20	08/05/20	Testing Phase		80	
17/02/20	27/03/20	Define Analytical Tests	DAT	20	
20/04/20	08/05/20	Performance Analysis and Cluster...	PACS	30	Low
17/02/20	24/04/20	Test Meetings	TM	20	

Figure 5.3: Gantt Diagram Text

Phase	Expected starting date	Current starting date	Expected finishing date	Current finishing date
Sprint 1	17/02/20	17/02/20	6/03/20	6/03/20
Sprint 2	9/03/20	9/03/20	3/04/20	17/04/20
Sprint 3	3/04/20	17/04/20	17/04/20	8/05/20

Table 5.2: Summary of the tasks

From a thorough analysis of the resulting schedule we can make the following conclusions:

- Spints 2 and 3 have been delayed given the current context of covid-19, where we had a lack of meetings for almost 3 weeks, making it harder to follow the tasks given that we had to validate the current state of the general many-body simulation.
- Sprint 1 was taken without delays and given the fast progress we had, we added a symbolic expression preprocessing to be able to specify the exact equations used on the simulation.

We can clearly see that our work has been delayed starting from the second sprint. This has happened given current context of covid-19, there has been a gap of almost three weeks were we had no meetings

Chapter 6

Project Budget

6.1 Considerations

The expenses of this thesis is based on the current hardware's powerhouse, the cost of projects of this caliber may change given the development of computer architectures that outperforms current ones. It may even be possible for future hardware's capabilities to run these simulations without the need of a cluster, making the costs significantly drop. Doing the thesis inside the a university department and without a budget given, meant for us that the project had the be the most affordable. Therefore, we have searched for resources that were free of use meaning that, if possible, the software used would be free of use and the facilities should be provided by the university.

6.2 Identification of costs

For the development of this thesis there is a need for both human and material resources. The material resources are hardware (laptop and computers), facilities (University facilities and Cluster) and the software used for the development.

The human resources are those that carry each of the defined tasks of the thesis. Each role has a specific responsibility and tasks to do. The following are the specific roles that are needed for this thesis.

- **Test Engineer:** person responsible of designing and checking the tests so that the platform has the required robustness and precision wanted. Specifically, the robustness and precision of the simulation and the results taken from it.

Role	Annual Salary	Total including SS	Price hour	Total hours project	Role total cost
Back-end Engineer [6]	36.790,00 €	49.666,5	27,9 €	50	1.395,00€
Front-end Engineer [8]	30.000,00 €	40.500 €	22,75 €	30	682,50€
System Engineer [11]	32.493,00 €	43.865,55 €	22,75 €	30	682,50€
Physicist [7]	33.000,00€	44.550,00 €	25,00 €	134	3.350,00€
Programmer [10]	26.198,00 €	35.367,30 €	19,87 €	274	5.444,38€
Tests Engineer [13]	29.498,00 €	39.822,30 €	22,37 €	74	1.655,38€
Machine Learning Engineer [9]	27.306,00 €	36.864,10 €	20.70 €	50	1.035,00€
Technical Writer[12]	26.263,00 €	35.355,05 €	19,92 €	112	2.231,04€

Table 6.1: Annual Salary roles thesis

Product	Price	Units	Useful life	Total estimated amortization
XPS 13 7390 2-in-1	1.299,00 €	1	5 years	86,60 €
OnePlus 7	559,00 €	1	3 years	62,11 €
	1.858,00 €			148,71€

Table 6.2: Hardware budget

- **Physicist:** person responsible of defining the basis of the simulation, the domain and the particle interaction.
- **Programmer:** person responsible of implementing the overall architecture of software. This includes the development of the simulations of the Single-body and Many-body systems.
- **System Engineer:** person responsible of orchestrating the software on a remote system. In this case, the system to be orchestrated is the applied physics department's cluster.
- **Machine Learning Engineer:** person responsible of models based on artificial intelligence techniques. In our case, the person will be responsible of the development of the parameter space search using artificial intelligence techniques like hill climbing.
- **Back-End Engineer:** person responsible of the development of the communication of the values given by the simulation with the front-end by means of basic databases and communication protocols.
- **Front-End Engineer:** person responsible of the design and development of the graphical user interface that will be used for the selection of domain of the simulation, selection of the particles interaction and selection of the experiments simulation.
- **Technical Writer:** person responsible of writing the overall documentation of this project. Mostly documentation that comes form the project management.

Facility	**Time**	**Total price**
Applied physics cluster	300h	free
FIB B5 meetings room	28h	free
30 m^2 flat in barcelona (les corts) [5]	4 months	1.780,80€
		1.780,8€

Table 6.3: Facilities budget

Name	**Price**
Overleaf	free
VS studio code	free
Github	free
GantProject	free
Draw.io	free
LibreOffice	free
Anaconda Distribution	free
	0,00€

Table 6.4: Software budget

Activity	Import	Roles	Hours
MT - Weekly or Biweekly Meetings	1.614,47€	Programmer, Physicist, Test Engineer	24
SD - Scope Definition	597,60€	Technical Writter	30
PD - Plan Definition	139,44€	Technical Writter	7
DOC - Documentation	1.294,80€	Technical Writter	65
SR - Sustainability Report	99,60€	Technical Writter	5
BG - Budget	99,60€	Technical Writter	5
DS - Design	1.346,10€	Programmer, Physicist	30
EV - Evaluation	1.346,10€	Programmer, Physicist	30
OBOE - One-Body Oscillator Experiment	955,40€	Programmer, Back-End Engineer	20
MBOE - Many-Body Oscillator Experiment	1.433,10€	Programmer, Back-End Engineer	30
GMBE - General Many-Body Experiments	1.192,20€	Programmer	60
GUIP - Interface for User Input	227,5€	Front-End Engineer	10
VIZ - Visualization Search Space Tool	455,00€	Front-End Engineer	20
HYPO - Hyper Parameter Optimization	2.028,8€	Programmer, Machine Learning Engineer	50
DAT - Define Analytical Tests	947,4€	Physicist, Tests Engineer	20
PACS - Performance Analysis and Cluster Support	1.278,60€	Programmer, System Engineer	30
TM - Test Meetings	1.421,10€	Tests Engineer, Physicist	30
	16.476,81€		367

Table 6.5: Human resources budget relative to Gantt tasks

6.3 Cost Estimates

The estimated costs for each of the roles of the tasks are taken from the website glassdoor [4] that has real world data taken daily in the fields of each of the roles used for this project. The annual intake for each job role taken from glassdoor has no consideration for social security therefore to calculate the cost per hour we take into account the social security and a 1780h of estimated work per year. From here we can calculate the labor agreement costs each role would have.

The estimated costs of hardware are low given that the only need for hardware is a laptop to develop the software of the simulation and the phone to test the graphical user interface in different size of displays.

The estimated cost of software, given the open-source approach that we have taken, will end up being zero.

Given that this thesis is made in the applied physics department most of the facilities are amortized by the university. Therefore, the costs of accessing to a cluster and university buildings is amortized, so the only cost accounted is a 30 m^2 flat rent in les corts.

Taking into account possible unexpected events, the resulting budget has a contingency of 15% making the resulting budget $(SB + FB + HB + HRB) \cdot 1.15 = (0,00 + 1.780,80 + 148,71 + 16.476,81) \cdot 1.15 = 21.167,27$€.

A contingency plan of 15% has been taken because most of the resourses of this project are human resources and most probably in case of failure in some of the tasks and average of 50h extra has been estimated to be added in human resources

so budged would round up to a 15% increase in case of failure.

Chapter 7

Sustainability Report

When talking about the sustainability of a thesis there are many ways we can tackle this problem. For example, in computer science one of the main aspects is the efficiency of the computation, a less efficient computation will lead to more energy consumption of the system that holds that program, and a more efficient computation will consume less and have more free resources for other programs.

Another aspect we can take into account is the social aspect. Many software nowadays is used to enhance and improve our social lives by easing daily activities like communication, transport and information gain.

The economic aspect is also important, programs are know for replacing, making easier and faster tasks we do. This can lead to major economic benefits that we all know and thrive from them.

7.1 Economic Dimension

Regarding PPP: Reflection on the cost you have estimated for the completion of the project:

Budget reflected in this thesis is precise and transparent, there has been no intention on overloading the budget with meaningless resources or underloading the budget to make it harder to follow.

Regarding Useful Life: How are currently solved economic issues (costs...)

related to the problem that you want to address (state of the art)?

This project has no cost other than human resources, also this was intended just for internal purposes, meaning it has no external costs.

How will your solution improve economic issues (costs ...) with respect other existing solutions?

There are no other existing solutions that do give accurate static and dynamic descriptions of the many-body simulation, this is a proof of concept and it is not comparable to other solutions because none exceed in both describing the static and dynamic properties and dynamic properties accurately. Therefore, there is no economic improvement, also this tool is an internal tool with a use meant for research, no economic purposes are related to this thesis.

7.2 Environmental Dimension

Regarding PPP: Have you estimated the environmental impact of the project?

The only negative impact this project may have is the energy consumption on the hardware where the simulation is run. This can be reduced by doing more efficient and performant computations of the simulation, nevertheless the results outweigh the electrical consequences.

If we suppose that this tool will also be used outside the applied physics department then, if the simulations results are not correct and are used in an industry with risks like a pharma or food industry. Then this tool could lead to major threats.

Regarding PPP: Did you plan to minimise its impact, for example, by reusing resources?

The only resources that can be reused are physical resources like computers, laptops and the cluster. And for sure, these will be reused because of the much longer lifespan these have compared to this thesis.

Regarding Useful Life: How is currently solved the problem that you want to address (state of the art)?, and how will your solution improve the environment with respect other existing solutions?

The simulation contains both the precision of the static and dynamic simulations of the many-body simulation. This is a ground no other paper has tried and done correctly, if done correctly this could lead to more precise and performant solutions, related to the environment, this could lead to less costs in energy consumption.

7.3 Social Dimension

Regarding PPP: What do you think you will achieve -in terms of personal growth- from doing this project?

This is a major challenge for me. I wanted to learn from other fields of study that could be applied to my degree and one was physics. This project contains quantum physics, calculus and artificial intelligence and all of these will be achieved proficiently by the end of this thesis.

Regarding Useful Life: How is currently solved the problem that you want to address (state of the art)?, and how will your solution improve the quality of life (social dimension) with respect other existing solutions?

As mentioned above, the simulation contains both the precision of the static and dynamic simulations of the many-body simulation. This is a ground no other paper has tried and done correctly, if done correctly this could lead to more precise and performant solutions.

This tool will ease the research process in studying the interaction of bosonic particles, by giving faster and equally or more precise simulation and an easier interface for physicists, that will let them fully express the physical constraints they want on the domain and particles interaction, without the burden of developing a tool for that specific experiment case.

Regarding Useful Life: Is there a real need for the project?

The Barcelona Quantum Simulation group needs a proof of concept of a the theoretical proposal they had on how to solve both the dynamic and static properties accurately of the particle simulation. If this thesis gives feasible results, then the simulations they already had could be improved and/or get faster than previous ones. And there is always a need for more precise and faster simulations given the time these simulations can take.

	PPP	Exploitation	Risks
Environmental	Reduce impact by making more efficient and cost effective algorithms that produce same results with less consumption of resources.	No existing state of the art is known and analytical solutions are too expensive, our solution would be way more efficient compared to analytical solutions.	If the solution is found no environmental risks can be made, just environmental benefits.
Economic	Given the research purpose of this project reducing expenses in human resources would lead to poor results and material resources are amortized by the university, therefore, we find ourselves with an already optimal solution.	Our approach could lead to more precise and performant solutions therefore less expenses in compute would be taken.	Only if money is at steak and simulation results do not give the expected results.
Economic	Given my non existing background in physics this project is a major challenge for improving calculus and physics knowledge.	By giving faster and more precise simulation and an easier interface for physicists, this tool would ease the work for physicists	The tool may not adapt to the physicists purposes making the main purpose of this tool meaningless.

Table 7.1: Sustainability Matrix Summary

Chapter 8

Prior Knowledge

In this chapter we will make an insight in on the foundations needed to understand how our tool works, this thesis has no intention on going in depth on the physical properties of the systems we want to study, but still, we must understand the foundations of them.

8.1 Hamiltonian

In quantum mechanics the Hamiltonian is an operator that defines the possible measurable total energies of a system. Similar to classical mechanics, the expression corresponds to the sum of the kinetic $\hat{T}$ and potential $\hat{V}$ energies in the system. In the case of one particle we have:

$$\hat{H} = \hat{T} + \hat{V} \tag{8.1}$$

or more explicitely

$$\hat{H} = -\frac{h^2}{2m}\nabla^2 + \hat{V} \tag{8.2}$$

Where $\hat{T} = -\frac{h^2}{2m}\nabla^2$, when extended to many particles we have

$$\hat{H} = -\frac{h^2}{2m}\sum_{i}^{n}\nabla_i^2 + \hat{V} \tag{8.3}$$

For our intended purposes we want a highly flexible formulation for the Hamiltonian so the expression we want to manipulate is

$$\hat{H} = -\frac{h^2}{2m}\sum_{i}^{n}\nabla_i^2 + \sum_{i}^{n}V(r_i) + \sum_{i<j}^{n}V(r_{ij}) \tag{8.4}$$

having one body kinetic energy terms, one and two body potential energy interactions, where $\nabla_i^2 = \left(\frac{\partial^2}{\partial x_i} + \frac{\partial^2}{y_i} + \frac{\partial^2}{\partial z_i}\right)$ is the Laplace operator for the i body.

The Hamiltonian will be important to compute $E_v(a) = \frac{\langle \Psi(a)|\hat{H}|\Psi(a)\rangle}{\langle \Psi(a)|\Psi(a)\rangle}$, i.e, the expected value of the sum of the kinetic and potential energy. This is important for finding the ground-state of a trial wave function.

8.2 Wave Function

The wave function is an expression that represents the whole state of a quantum system. The wave function looks like the following

$$\Psi = \Psi_0 \Psi_1 \Psi_2 \Psi_3 ... \tag{8.5}$$

where

$$\begin{aligned} \Psi_0 &= f_0 \\ \Psi_1 &= \prod_{i}^{N_p} f_1(r_i) \\ \Psi_2 &= \prod_{i}^{N_p} \prod_{j=i+1}^{N_p} f_2(r_{ij}) \\ &\dots \end{aligned} \tag{8.6}$$

An example of wave function would be trial wave function of the helium atom. The potential energy of the helium atom is defined by the attraction from the nucleus of both electrons r_1, r_2 given by $V(r_1, r_2) = -\frac{2ke^2}{r_1} - \frac{2ke^2}{r_2} + \frac{ke^2}{r_{12}}$ where $r_{12} = |r_1 - r_2|$ and the repulsion of both electrons $\frac{ke^2}{r_{12}}$, from here we can extract a trial wave function composed by $f_1(r_i) = e^{-\alpha(r_i)}$ and $f_2(r_i j) = e^{\beta(r_{ij})}$.

8.2.1 Laplacian of the wave function

For the stochastic optimization methods we need a clear understanding on how to apply the Laplacian to the wave function and how the resulting function looks as a composite of derivatives of functions f_1, f_2. Let $\Psi = \Psi_1 \Psi_2$ where $\Psi(\alpha, i)$ is

$$\alpha \in 0, 1, ..., N_p : \quad \Psi(\alpha, i) \equiv \begin{cases} f_1(r_i) & i = j \\ f_2(r_{ij}) & i \neq j \end{cases} \tag{8.7}$$

Let $\nabla_i^0 = \frac{\partial}{\partial x_i}$, $\nabla_i^1 = \frac{\partial}{\partial y_i}$ and $\nabla_i^2 = \frac{\partial}{\partial z_i}$. We have to find a reduced form of $\frac{\Delta\Psi}{\Psi}$ for the compute of $\hat{T}\Psi$. Let $\frac{\nabla_i^a \nabla_i^a \Psi}{\Psi} = \nabla_i^a \left(\frac{\nabla_i^a \Psi}{\Psi}\right) + \left(\frac{\nabla_i^a \Psi}{\Psi}\right)^2$, if we work with the right expression we can work with $\left(\frac{\nabla_i^a \Psi}{\Psi}\right)$ and then $\nabla_i^a \left(\frac{\nabla_i^a \Psi}{\Psi}\right)$. To reduce the size of the expression, given by

$$\frac{\nabla_i^a \Psi}{\Psi} = \frac{\nabla_i^a \Psi(i)}{\Psi(i)} = \sum_{\alpha}^{N_\alpha} \left(\frac{\nabla_i^a \Psi(\alpha, i)}{\Psi(\alpha, i)} \right) = \sum_{\alpha}^{N_\alpha} \begin{cases} \dfrac{\nabla_i^a f_1(i)}{f_1(i)} & i = \alpha \\ \dfrac{\nabla_i^a f_2(\alpha, i)}{f_2(\alpha, i)} & i \neq \alpha \end{cases} \tag{8.8}$$

$$\nabla_i^a \left(\frac{\nabla_i^a \Psi}{\Psi} \right) = \nabla_i^a \sum_{\alpha}^{N_\alpha} \left(\frac{\nabla_i^a \Psi(\alpha, i)}{\Psi(\alpha, i)} \right) = \sum_{\alpha}^{N_\alpha} \begin{cases} \nabla_i^a \left[\dfrac{\nabla_i^a f_1(i)}{f_1(i)} \right] & i = \alpha \\ \nabla_i^a \left[\dfrac{\nabla_i^a f_2(\alpha, i)}{f_2(\alpha, i)} \right] & i \neq \alpha \end{cases} \tag{8.9}$$

from wich we can compute $T_{\text{OneBody}}(i, \alpha, a, 1) \equiv \dfrac{\nabla_i^a f_1(i)}{f_1(i)}$, $T_{\text{OneBody}}(i, \alpha, a, 2) \equiv \nabla_i^a \left[\dfrac{\nabla_i^a f_1(i)}{f_1(i)} \right]$, $T_{\text{TwoBodies}}(i, \alpha, a, 1) \equiv \dfrac{\nabla_i^a f_2(\alpha, i)}{f_2(\alpha, i)}$ and $T_{\text{TwoBodies}}(i, \alpha, a, 2) \equiv \nabla_i^a \left[\dfrac{\nabla_i^a f_2(\alpha, i)}{f_2(\alpha, i)} \right]$ from which we have that

$$T_{\text{Der}}(i, \alpha, a, k) \equiv \begin{cases} T_{\text{OneBody}}(i, \alpha, a, k) & i = j \\ T_{\text{TwoBodies}}(i, \alpha, a, k) & i \neq j \end{cases} \tag{8.10}$$

where we have that $T_{\text{Der}}(i, \alpha, a, 2) = \nabla_i^a T_{\text{Der}}(i, \alpha, a, 1)$, from here we can clearly see that

$$\frac{\nabla_i^a \nabla_i^a \Psi}{\Psi} = \sum_{\alpha}^{N_\alpha} T_{\text{Der}}(i, \alpha, a, 2) + \left(\sum_{\alpha}^{N_\alpha} T_{\text{Der}}(i, \alpha, a, 1) \right)^2 \tag{8.11}$$

In other words

$$\hat{T}\Psi = -\frac{h^2}{2m} \sum_{a=1}^{3} \sum_{i=1}^{N_p} \left[\left(\sum_{\alpha}^{N_\alpha} T_{\text{Der}}(i, \alpha, a, 2) \right) + \left(\sum_{\alpha}^{N_\alpha} T_{\text{Der}}(i, \alpha, a, 1) \right)^2 \right] \tag{8.12}$$

8.3 Monte Carlo Integration

The idea behind Monte Carlo Integration is to compute the definite integral $I = \int_a^b h(x)dx$ by randomly sampling from the function $h(x)$. Monte Carlo integration is well known as a consistent and unbiased method. Given its simplicity it adapts well to multidimensional integration, with a signal-to-noise ration converging as $\sqrt{N}$, that does not depend on the number of dimensions, makes it a clear competitor to other approximation methods like Riemann sum or Simpson's rule that fall into the curse of dimensionality where they require N^d samples, making a much slower convergence as dimensions grow.

8.3.1 Naive Monte Carlo Quadrature

This is the simplest possible way of computing the definite integral:

$$I = \int_a^b h(x)dx \tag{8.13}$$

where $h(x)$ is an arbitrary function, and provided the integral is defined. Let $(x_1, x_2, \ldots, x_n)$ be a set of random numbers uniformly distributed over the interval (a, b). Because of the central limit theorem we know that.

$$I = \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} h(x_i) \tag{8.14}$$

The variance

$$\sigma^2 = \int_a^b (h(x) - I)^2 dx \tag{8.15}$$

can also be determined with the following

$$\sigma^2 = \lim_{n\to\infty} \frac{1}{n} \sum_{i} = 1^n h^2(x_i) - I^2 \tag{8.16}$$

For a limited sampling n we can write the result of the integral as the following:

$$I \pm (\sigma^2/N)^{1/2} \tag{8.17}$$

with a confidence interval of 68%
The problem with the crude technique is lack of efficiency to get a good estimate. Several techniques have been developed to mitigate the problems of the crude Monte Carlo but for the many body problem we will be using importance sampling.
Importance sampling Importance sampling handles both the sampled function h and the sampling algorithm, in order to is to significantly decrease the variance of the resulting function. Given a probability distribution function $f(x)$ within the interval (a, b), equation 8.13 can be rewritten as

$$I = \int_a^b (h(x)/f(x))f(x)dx \tag{8.18}$$

Let $(x_1, x_2, \ldots, x_n)$ be a set of random numbers drawn from the probability distribution function $f(x)$ the estimate integral is:

$$I = \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} h(x_i)/f(x_i) \tag{8.19}$$

And the variance is:

$$I = \int_a^b (h(x)/f(x))^2 f(x)dx - I^2 \tag{8.20}$$

A good choice of $f(x)$ can lead to a smaller variance while the value of the integral remains the same.

8.3.2 Markov Chains

The problem with importance sampling is the determination of the importance sampling function such that it minimizes the variance of I. Best cases would be functions that mimic the function $h(x)$ that we want to integrate. Either way, there is no aid in choosing the wanted distribution function. Also, we have the problem of carrying the sampling with this function. Both of these problems have intimidating multivariate integrals. In our case the importance sampling function is already defined in the problem we want to solve, the importance sampling is the trial wave function squared.However, it still remains the problem of sampling from a very complicated distribution function. Metropolis–Hastings is a stochastic algorithm that produces a random walk that follows this complicated distribution function.

This random walk is of the type known as Markov Chain or Markov Process. Given a system with states $S_1, S_2, ..., S_N$.At each step of the evolution of the system, each state can jump to any other state of the set, itself included. This jump is characterized by a transition probability matrix p_{ij}, which represents the probability of jumping from the state i to any other state j. Note that p_{ii} is also included, and that these matrix elements must fulfill the following conditions:

$$0 \leq p_{ij} \leq 1 \tag{8.21}$$

$$\sum_j p_{ij} = 1 \tag{8.22}$$

In addition, it is required that the transition from i to j is independent on the previous occupied states during the evolution.

A matrix that satisfies both (8.21) and (8.22 is called a stochastic matrix, and the process related to it is called Markov Chain.

The generalization to the case of the continuum is called Markov process. We should use now x to label the states, and define a transition density $p(x, x')$ with the properties

$$p(x, x') \geq 0 \tag{8.23}$$

$$\int p(x, x')dx' = 1 \tag{8.24}$$

We want to know Which is the probability P_k of having passed on some state S_k. More precisely. Assume we carry on N jumps, and let N_k bet the number of stops at S_k. The searched probability is $P_k = lim_{N\to\infty}\frac{N_k}{N}$

The stochastic matrix has to fulfill the fillowing conditions to be able to know the probability P_k.

1. The random walk must be endless, i.e., there is none state with $p_{ii} = 1$. If that state would exist, then our walker would be trapped at it. This kind of

state is called an absorbing wall.

2. The chain is irreducible. Probably the opposite concept is easier to understand: if we can classify the states in two subsets so that there cannot be a transition from one subset to the other, then the chain is termed reducible.

3. The chain must be aperiodic, let p_{ii}^n be the probability of returning to the state i at an n step and let $t \in 2, 3, ...$, chain is periodic if $p_{ii}^n = 0$ for $n \neq t, 2t, ...$ and $p_{ii}^n \neq 0$ for $n = t, 2t, ...$, otherwhise it is aperiodic.

8.3.3 Solution of the direct problem

The algebraic determination of the probabilities P_k is quite simple. The probability of arriving to a state S_i is the product of the probability of being previously in another state S_k times the transition probability, i.e.

$$P_i = \sum_k P_k P_{ki} \tag{8.25}$$

This is a set of homogeneous linear equations, which as expected are not independent. as it can be checked by summing up over the free index i. It should be supplemented by the normalization condition

$$\sum_i P_i = 1 \tag{8.26}$$

and the system can now be solved.

Algorithm 1 Compute Probability Distribution Function

input : A transition matrix T_{ij}, a set of states S and an integer n

output: A probability distribution function Pr_i from n samples

$\forall i$: $Pr_i = 0 \quad s_i = s_0 \in S$ **for** $step \in 1, 2, ..., n$ **do**

  $z = U(0,1)$ **for** $s_j \in S \setminus \{s_i\}$ **do**

    if $z < T_{ij}$ **then**

      break;

    else

      $z = z - T_{ij}$

  end

  $Pr_j = Pr_j + 1 \; s_i = s_j$

end

for $s_i \in S$ **do**

  $Pr_i \leftarrow Pr_i/n$

end

return Pr

8.3.4 Metropolis-Hastings

The Markov Chains algorithm let us determine the probability distribution given a random walk. The analog of this problem is given a known probability distribution, find a random walk that follows the probability distribution. In other words, given P_i we want to find the stochastic matrix P_{ij}. As we may know, this problem is not unique.

Given a symmetric and stochastic matrix q_{ij} the random walk is given by the following.

$$p_{ij} = \begin{cases} q_{ij} & i \neq j \wedge P_i < P_j \\ q_{ij}\dfrac{P_j}{P_i} & i \neq j \wedge P_i \geq P_j \\ q_{ii} + \displaystyle\sum_k q_{ik}(1 - P_k/P_i) & 100 \leq x \end{cases} \tag{8.27}$$

So given a state S_i, we pick a trial state S_j with probability q_{ij}. If P_i ¡ P_j then the move is accepted. Otherwise, the move is accepted with probability $\dfrac{P_j}{P_i} < 1$ if the new state is not accepted we go back to the old state S_i.

The algorithm holds for continuous probability functions

Algorithm 2 Metropolis Hastings

input : A probability distribution p an integer n

output: A random walk RandWalk of length n

RandWalk ← EmptyQueue RandWalk = `Queue` () x = pick a random starting state

while $n > 0$ **do**

  $x' = x + U(-D, D)$ **if** $p(x') < p(x)$ **then**

    $z = U(0, 1)$ **if** $p(x')/p(x) < z$ **then**

      `/* Move is not accepted */`

      $x' = x$

    end

  end

  RandWalk.push(x')

end

return RandWalk

Which can be simplified to the following

Algorithm 3 Metropolis Hastings Shorter

input : A probability distribution p an integer n

output: A random walk RandWalk of length n

RandWalk ← EmptyQueue RandWalk = `Queue` () x = pick a random starting state

while $n > 0$ **do**

  $x' = x + U(-D, D)$ $z = U(0, 1)$

  if $p(x')/p(x) < z$ **then**

    `/* Move is not accepted */`

    $x' = x$

  end

  RandWalk.push(x')

end

return RandWalk

The value of D has a direct effect on the ratio of acceptance and rejection, an appropiate value would be one that has an acceptance rate around 50% and 70%.

8.4 Variational Montecarlo

Given a many body hamiltonian

$$\hat{H} = -\frac{h^2}{2m}\sum_i \nabla_i^2 + \sum_i V(i) + \sum_{i<j} V(i,j) \tag{8.28}$$

We want to compute the exact solution of the ground-state energy. One of the features which makes the many body problem so complex is the fact that not all stochastic methods can be applied the two body potential has a strong repulsion when the particles are near. Perturbative methods could be a way to solve this problem but these are hard given the strong repulsion at short distances given by the potential.

A simple alternative to the exact solution is to compute a variational upper bound to the ground-state energy using a trial function wich takes into account appropiately the repulsive core of the two body interaction. The simplest form is known with the name of Jastrow and is made up of a single particle orbitals and a correlation factor which is the product of all pairs of two body correlation.
A general form for the trial wave function to represent bosons is

$$\Psi_T = f_0 \prod_i f_1(r_i) \prod_{i<j} f_2(r_{ij}) \tag{8.29}$$

Other general forms for the trial wave function may include three body correlation, spin or isospin dependence, but this is not our case.
The upper bound to the ground-state energy for a parametrization a is

$$E_v(a) = \frac{\langle \Psi(a)|\hat{H}|\Psi(a)\rangle}{\langle \Psi(a)|\Psi_T(a)\rangle} = \frac{\int \Psi^\dagger(R,a)\mathcal{H}\Psi(R,a)dR}{\int \Psi^\dagger(R,a)\Psi(R,a)dR} \tag{8.30}$$

Which can be rewritten as the following

$$E_v(a) = \frac{\int |\Psi(X,a)|^2 \frac{\mathcal{H}\Psi(X,a)}{\Psi(X,a)}dX}{\int |\Psi(X,a)|^2 dX} \tag{8.31}$$

if we enterpret $\dfrac{|\Psi(X,a)|^2}{\int |\Psi(X,a)|^2 dX}$ as a probability distribution $p(X,a)$ and $E_L(X) = \dfrac{\mathcal{H}\Psi(X,a)}{\Psi(X,a)}$ then

$$E_v = \int p(X,a)E_L(X,a)dX \tag{8.32}$$

This integral is written as the Monte Carlo Quadrature with $p(X,a)$ as the importance sampling function. Givent a set of random variables $X_1, X_2, ..., X_N$ we can estimate E_v using metropolis hasting as

$$E_V = \frac{1}{N}\sum_i E_L(X_i,a) \tag{8.33}$$

And given the local energy $E_L(R)$

$$E_L(R) = \frac{1}{\Psi_T(R)}\hat{H}\Psi_T(R) \tag{8.34}$$

let $p(R)$ be a pdf

$$p(R) = \frac{|\Psi_T(R)|^2}{\int |\Psi_T(R)|^2 dR} \tag{8.35}$$

where $\int |\Psi_T(R)|^2 dR$ is the normalization constant. So given a positive, normalized probability distribution function $p(R)$, the variational energy is

$$E_V = \int p(R) E_L(R) dR \tag{8.36}$$

As we can see in 8.36, we have the Monte Carlo quadrature having $p(R)$ as the importance sampling function and $E_L(R)$ as the function to be sampled.
So given a random walk $R_1, R_2, ..., R_N$ that follows the probability function $p(R)$ we can obtain the estimate

$$E_V = \frac{1}{N} \sum_i E_L(R_i) \tag{8.37}$$

with a statistical error

$$\sigma = \frac{\sqrt{\frac{1}{N} \sum_i E_L^2(R_i) - E_v^2}}{\sqrt{N}} \tag{8.38}$$

8.5 Ground-state search

The main purpose of this thesis is that given a trial Wave Function $\Psi_T(a)$ parametrized by a and some defined potential of the hamiltonian, develop a tool capable of finding the groundstate $E_0 = min\{Ev(a) : a \in A\}$ where A is the set of all possible parametrizations of $\Psi_T(a)$. The finding of the ground-state is important for topics that may go out of this project like the study of superfluids.

8.6 Simulated Annealing

Simulated Annealing is one of the most known techniques of stochastic optimization, the main idea comes from, given a starting state s, a possible new s' and a temperature T, whether or not accept a move from s to s', based on the transition function $P(e, e', T)$. The transition function $P(e, e', T)$ that depends on the energies e and e' of the states s and s' respectively and a temperature T. States with smaller energy are better than those with grater energy. This means that going downhill is promoted but small steps going uphill may also bee accepted. The condition of accepting uphill moves gets harsher as T decreases.

Our objective in the ground-state search is finding the configuration a that is minimal on $E_v(a)$. We can model this objective function as the energy $E(s)$ of state s where each state will represent a configuration of the trial wave function.

The following is a basic algorithm of the simulated Annealing:

Algorithm 4 Basic simulated annealing

input : A number of maximum steps $max(k)$

output: A state s that is an approximate to the optimal solution

$s =$ pick a starting state s_0

for $k \in 1, 2, ..., max(k)$ **do**

  $t =$ temperature(k+1 / max(k))

  $s' =$ neighbor(s)

  $z = U(0, 1)$

  if $P(E(s), E(s') \geq z$ **then**

    /* accept move */

    $s = s'$

  end

end

return s

8.7 Time-dependent Variational Monte Carlo

Based on the fundamental priciple of quantum phisics, we know that the evolition of a phisical state along time is the following

$$\frac{d\Psi}{dt} = -i\mathcal{H}\Psi \tag{8.39}$$

and we are particularly interested in both the eigenstates and eigenvalues that define the energy, particularly the one that is the lowest, i.e., the ground-state. The eigen states of the energy are the foundation of the mathematical space that belong to the physical states, known as Hilbert space. This means the wave functions we use, can formally be written as a linear combination with some weights λ_k of the eigen states of the energy.

$$\Psi(c) =_{k=0}^{\infty} \lambda_k(c)\Psi(E_k) \tag{8.40}$$

For each set of paràmeters c exists a linear combination of eigen states that reconstructs the wave function. Our objective is finding the parametrization c such that it produces a combination of weights λ_k where the ground-state is the most important, ideally $\lambda_0 = 1, \lambda_{k>0} = 0$. The method we want to evaluate has the parameters evolving over time $c(t + \epsilon) = c_\alpha(t) + \dot{c}_\alpha(t)\epsilon t$.
Following a mathematical principle that states the evolution over time can be made using real time, as stated in 8.39 that would have the evolution $\Psi(t) = \Psi(c_\alpha(t))$, and complex time $\Psi(t = -\operatorname{Im}\tau)$. Defining $\Psi(t = 0) \equiv \Psi(c_\alpha)$, 8.39 can compute $\Psi(t)$ where $t \in C$, the way we compute $\Psi(t = -\operatorname{Im}\tau)$ is the following

1. when τ is real, computing $\Psi(t = -\operatorname{Im}\tau)$ is easier that $\Psi(t = \tau)$.

2. when τ is complex,

   2.1. $\Psi(\tau)$ satisfies

$$\frac{d\Psi(\tau)}{dt} = -\mathcal{H}\Psi(\tau) \text{ that we will rewrite as } \frac{d\Psi(\tau)}{dt} + \mathcal{H}\Psi(\tau) = 0 \quad (8.41)$$

   2.2. when τ increases, $\Psi(\tau)$ gets closer to the ground-state, i.e, we have

$$lim_{\tau\to\infty}\Psi(\tau) = \Psi_0 \text{ (where } \Psi_0 \text{ is the Wave Function of the ground-state)} \quad (8.42)$$

Following the scheme of evolving the wave function $\Psi(c)$ using parameters $c \to c(t)$ that change over time, this wave function will be called $\Psi(\tau)$, so we have

$$\frac{d\Psi(\tau)}{d\tau} + \mathcal{H}\Psi(\tau) = 0 \quad (8.43)$$

From now on we will use t as τ. Therefore, we have

$$\frac{\partial\Psi}{\partial t} + \mathcal{H}\Psi = (\dot{\Psi} + \mathcal{H}\Psi) = 0 \quad (8.44)$$

where the time dependence is given by $c \to c(t)$:

$$\dot{\Psi} = \sum_{\alpha=1}^{N} \dot{c}_\alpha(t)\frac{\partial\Psi}{\partial c_\alpha} \quad (8.45)$$

Now given a finite set of operators $O_k(R) \quad k \in 0, 1, ...$ for exmaple:

$$\boxed{\begin{aligned} &\mathcal{O}_0(R) = 1 \\ &\mathcal{O}_1(R) = \sum_{i=1}^{n} x_i + y_i + z_i \\ &\mathcal{O}_2(R) = (\sum_{i=1}^{n} x_i + y_i + z_i)^2 \\ &\quad ... \end{aligned}} \quad (8.46)$$

let $\Psi = \Psi(c(t))$ and 8.41 we want to satisfy

$$\boxed{\int \Psi^\dagger \mathcal{O}_k(R)(\dot{\Psi} + \mathcal{H}\Psi)dR = 0} \quad (8.47)$$

knowing 8.45 we have that

$$\sum_{i=1}^{n} c(t) \int \Psi^\dagger \mathcal{O}_k(R)\frac{\partial\Psi}{\partial c}dR = -\int \Psi^\dagger O_k(R)\mathcal{H}\Psi dR \quad (8.48)$$

In the stationary state where $\dot{c}_{\alpha>0} = 0$ we have

$$\Psi(t) = e^{c_0(t)}\Psi_0 = e^{-E_0 t}\Psi_0 \text{ ,i.e. , } c_0(t) = -E_0 t \quad (8.49)$$

$$\dot{c}_0(t)\int \Psi^\dagger \mathcal{O}_k(R)\frac{\partial \Psi}{\partial c_0}dR = -E_0\int \Psi^\dagger \mathcal{O}_k(R)\Psi dR = -\int \Psi^\dagger \mathcal{O}_k(R)\mathcal{H}\Psi dR \quad (8.50)$$

Given a montecarlo run we can acumulate the estimates of the following equations

$$V_k = -\int \Psi^\dagger \mathcal{O}_k(R)\mathcal{H}\Psi dR \quad (8.51)$$

$$M_{k\alpha} = \int \Psi^\dagger \mathcal{O}_k(R)\frac{\partial \Psi}{\partial c_\alpha}dR \quad (8.52)$$

From (8.48) we see that

$$\sum_{\alpha=0}^{N_\alpha} M_{k\alpha}\dot{c}_\alpha(t) = V_\alpha \quad (8.53)$$

if we think of $V_k \quad k \in 0, 1, ..., N_K$ as the matricial form V, $M_{k\alpha} \quad \alpha \in 0, 1, ..., N_\alpha \; k \in 0, 1, ..., N_k$ as the matricial form M and $\dot{c}_\alpha(t) \quad \alpha \in 0, 1, ..., N_\alpha$ as the matricial form $c(t)$, we have that

$$\boxed{M \cdot \dot{c}(t) = V} \quad (8.54)$$

knowing that $\langle\Psi|f(R)|\Psi\rangle = \int \Psi^\dagger f(R)\Psi dR$ solving $\int \Psi^\dagger O_k \mathcal{H}\Psi dR$ or $\int \Psi^\dagger O_k(R)\frac{\partial \Psi}{\partial c_\alpha}dR$ involves sampling the following:

$$\boxed{\mathcal{O}_k(R)E_L(R) \quad or \quad \mathcal{O}_k(R)\frac{\partial \Psi(R)}{\partial c_\alpha(t)}(\Psi(R))^{-1}} \quad (8.55)$$

So given $\dot{c}_\alpha(t) = (M)^T BV$ where B is a square matrix of N_k rows and columns given 8.54 we need that

$$M \cdot (M)^T \cdot B \cdot V = V \quad (8.56)$$

and it hapens only if

$$B = (M \cdot (M)^T)^{-1} \quad (8.57)$$

so given a Monte Carlo run, with the estimates from 8.55 M and V we know the following:

$$\boxed{\dot{c}_\alpha(t) = (M)^T \cdot (M \cdot (M)^T)^{-1} \cdot V} \quad (8.58)$$

this means at each step Δt we know that

$$\boxed{c_\alpha(t + \Delta t) = c_\alpha(t) + \dot{c}_\alpha(t)\Delta t} \quad (8.59)$$

Therefore, we have a method that always reaches ground-state $\Psi_0 = \Psi(\lim_{t\to\infty} c_\alpha(t))$, this method converges exponentially as t approaches infinity.

8.8 Computer Algebra System

Computer algebra systems are tools that let to the manipulation of mathematical expressions. These tools have a large set of methods to perform operations like, Taylor series, derivation, indefinite integrals, or simplification rules like factoring, canceling terms, collecting terms or even term rewriting like expanding terms or finding common sub-expressions. These tools are used when perfect analytical solutions are needed, another benefit is having the change to get mathematical expressions that are less computationally expensive once they are simplified. But there are some trade-offs when using these tools, the main concern is related to the size of the expression you're manipulating, common known term rewriting algorithms **??** are know to be of polynomial complexity and normally rule-sets size although being a constant factor to the cost, tend to have a high influence given the size of these.

The need for mathematical manipulation tools comes from the need to solve expression involved in the simulation like the following: $\dot{c}_\alpha(t)$, $\frac{\nabla_i^2 \Psi}{\Psi}$

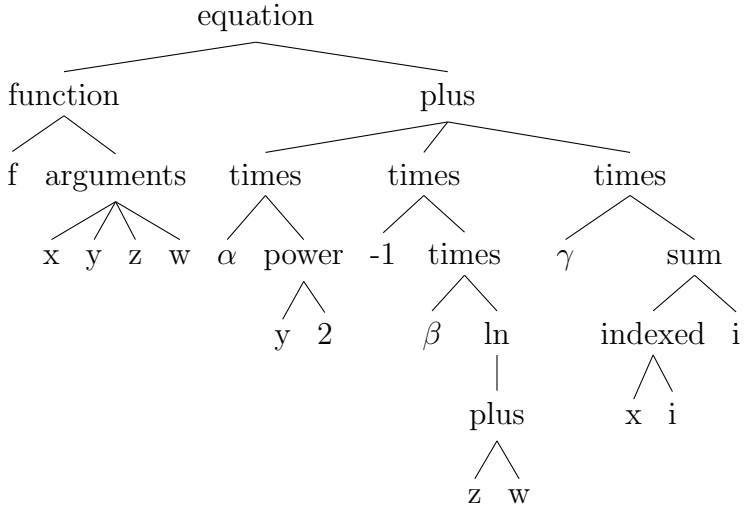
8.9 Parser

A parser is a tool that checks that the following input follows the syntactic and semantic rules defined. From here it builds an abstract syntax tree (AST), a data-structure that stores all the structural details and does not represent any unnecessary information.

Given a trial expression

$$f(x, y, z, w) = \alpha y^2 - \beta \ln(z + w) + \gamma \sum_i x_i \tag{8.60}$$

The AST is



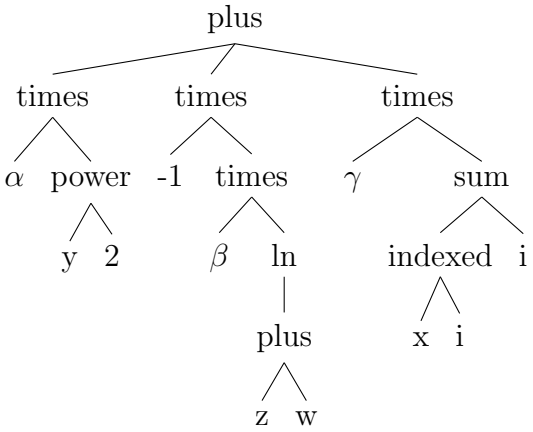
(8.61)

```
\[Alpha] * y^2 + \[Beta]*log[E, z + w] + \[Gamma] * Sum[x[i],
    i]
```

Listing 8.1: Sample Mathematica expression

In our case the parser is necessary to check that the structure of the equations is written properly and follows all the syntactic and semantic rules of the formal systems in mathematics, our users are highly experienced in the use of Mathematica so in our case we will have to parse Mathematica expression like the following.

That would produce the following AST



(8.62)

From the ast given by the parser we can easily construct any symbolic expression we want in our desired tooling for manipulating mathematical expressions.

Parsers are also capable of pointing out where possible mistakes could be made, giving easy and trackable response to the user.

Chapter 9

Development

9.1 Frontend-Backend

Our project has two main structures, the frontend where all the user input will be caught and used for the specific needs the user has defined, like defining the wave function or hamiltonian to be used in the simulation, searching for the grount state using the time-dependent variational montecarlo or using simulated annealing. The backend will be responsible of running both simulation methods, storing all necessary information of the simulation, converting the defined equations into simplified and high performant code, checking that the user input is correct and that the expressions pass the defined integrity tests.

9.1.1 Frontend

At first, when we thought of having the frontend and the backend separated we wanted to have something easy to modify, flexible and with libraries for visualization that could adapt to our needs without much effort. We thought that electron was a good choice because we could make a shippable application that could run in any device. It was a good idea but the problem we had was the communication between frontend and backend, we wanted to see the data of the simulation at realtime and doing that wasn't as trivial as we expected. Our work was always sketched in a jupyter notebook and once we found everything correct the process would be moving the visualizations from the jupyter notebook to the electron frontend and write the comunication with the backend of the parameters wanted from the simulations. Then we found Voila, it was an alternative that merged both frontend and backend and let us preserve all of our previous work, another benefit we had was that we already knew how it looked in the notebook so the effort of trying to make it to resemble to what it looked like in the notebook wasn't neccessary. So we ended up choosing Voila which led us to not even making the communication process and rewriting the visualization because voila makes this job for us. So we ended up with an even more simple cycle that lend us more time for the more important parts of the project.

Simulation Data

While we were making the first experiments with the One-body Oscillator and the Many-body oscillator we wanted to check how the progression of the local energy sampled by the variational montecarlo and also see the standard error, given that our random walk started at a random point our values could be biased at the start of the simulation we wanted to have the possibility to discard blocks of sampled values, so we did the following tool.

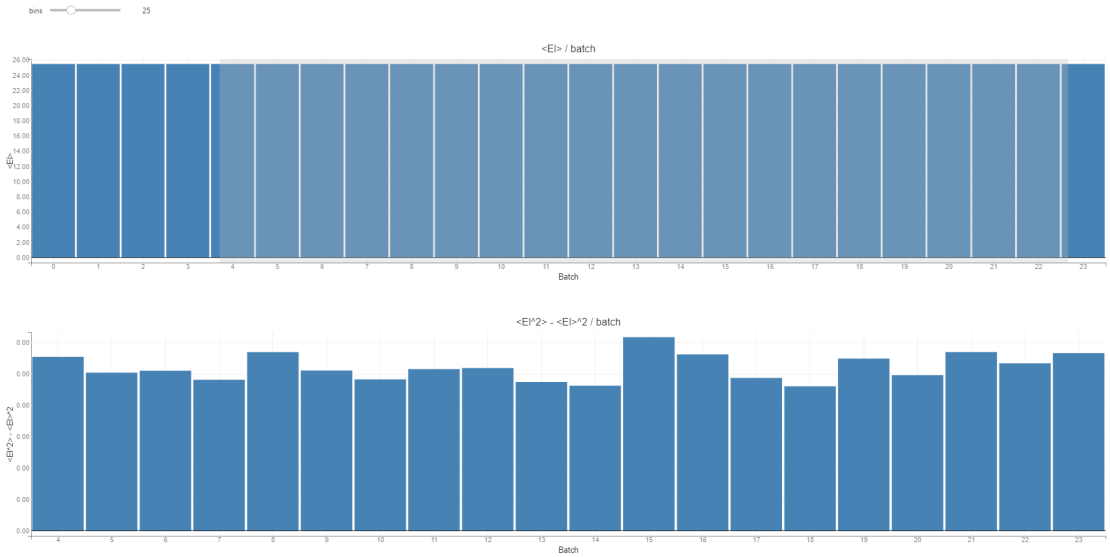


Figure 9.1: Metrics simulation many-body random walk with groundstate parametrization

In 9.1 we have a run of a many-body harmonic oscillator of 5 bodies with the parametrization of the ground state, the selected region is the region from wich we compute de E_v and also we can see the standard error of the selected region above.

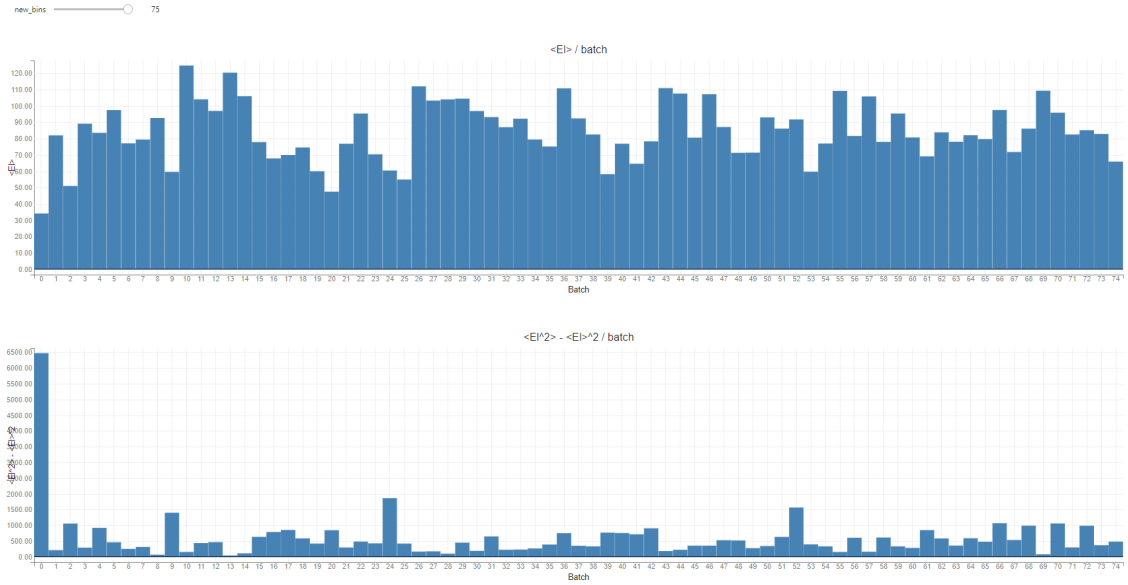


Figure 9.2: Metrics simulation many-body random walk with parametrization far from ground state

In (9.2) we have a run of a many-body harmonic oscillator of 5 bodies with a parametrization far from the groundstate, here we can clearly see a high standard error at the first batch that shoud be discarted.

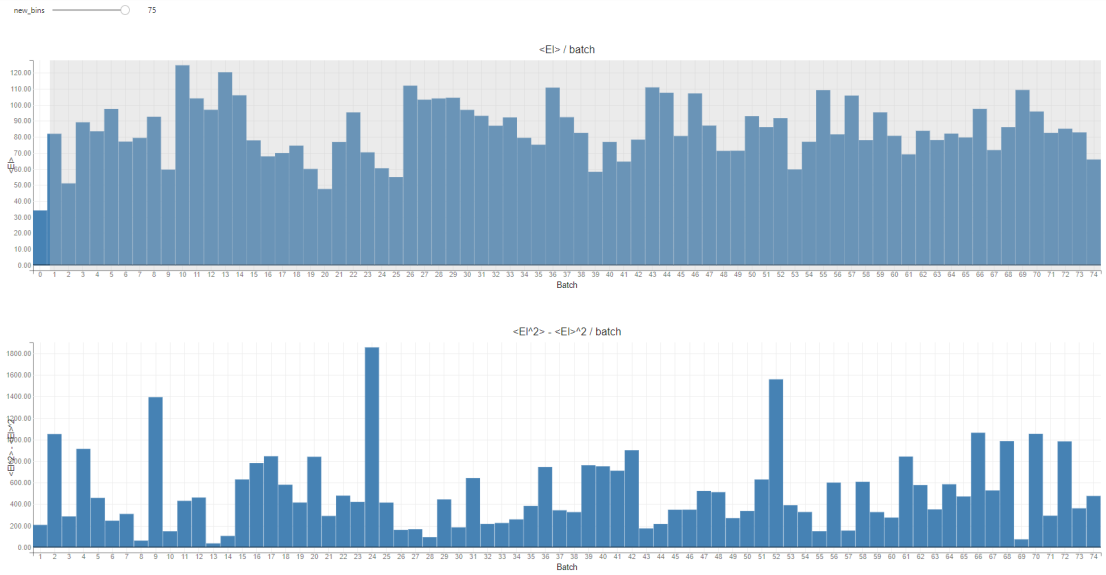


Figure 9.3: Metrics simulation many-body random walk with parametrization far from ground state discarting first batch

Equations input

To make it possible for the user to run the wanted simulation, the user has to define the Jastrow functions, the operators in case of wanting a Time-Dependent Variational Monte Carlo run and the potential terms of the hamiltonian.

Functions	Example Expression
Jastrow terms	
$f_0(R)$	e_0^c
$f_i(R,i)$	$e^{-\gamma(x_i^2+y_i^2+z_i^2)}$
$f_{ij}(R,i,j)$	$e^{-\alpha((x_i-x_j)^2+(y_i-y_j)^2+(z_i-z_j)^2))}$
Operators terms	
$\mathcal{O}_0(R)$	1
$\mathcal{O}_1(R)$	$\sum_{i=1}^{n} x_i^2 + y_i^2 + z_i^2$
$\mathcal{O}_2(R)$	$\sum_{i=1}^{n}\sum_{j=i+1}^{n} (x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2$
Hamiltonian potential terms	
$V_i(R,i)$	$k(x_i^2 + y_i^2 + z_i^2)$
$V_{ij}(R,i,j)$	$w((x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2)$

Table 9.1: Terms to be defined by the user in the UI

In 9.1 we have all the possible terms of that can be defined by de user that will be used by the resulting sumulation.
The following are some examples of how the UI looks once the equations in mathematica are written.

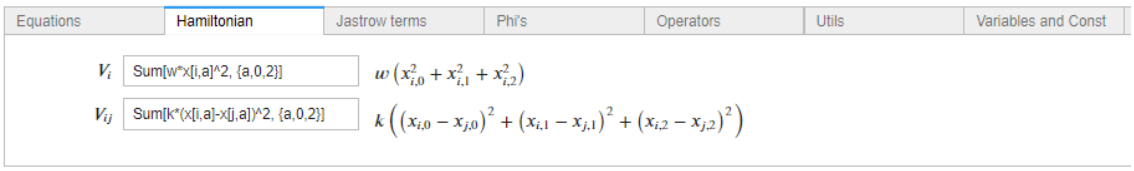


Figure 9.4: UI defining potential of the hamiltonian

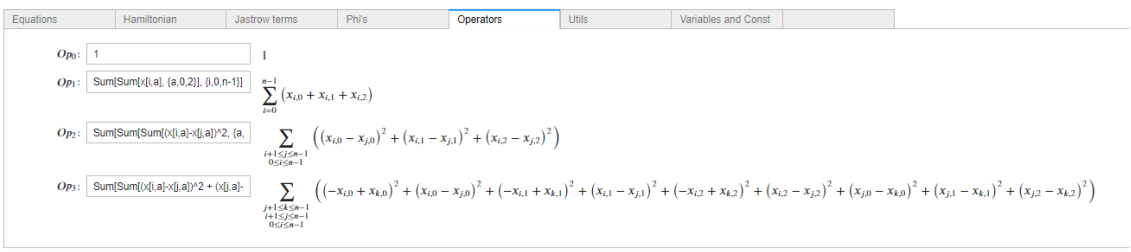


Figure 9.5: UI defining Operators

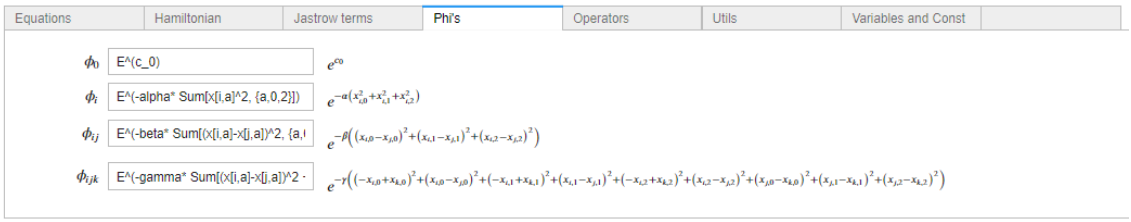


Figure 9.6: UI Wavefunction ϕ terms

9.2 Parser

The user has to be able to write wathever expresion that defines the jastrow of the wave function and the expressions that define the interactions from the potential of the hamiltonian.
We first thought of using plain latex notation to write the mathematical equations but as we developed more this first approach we found that many operations could have an ambiguous meaning. Like the following

LaTeX	Possible meanings
`A \times B`	Cross Product of A and B Product of A and B
`X^T`	Transpose X X to the power of T
`xyz`	Product of x, y, z Variable called xyz
`\sum _i x_i \sum _j y_j`	$\sum_i (x_i \sum_j y_j)$ $(\sum_i x_i)(\sum_j y_j)$

So removing the ambiguity meant doing a subset of the latex notation that could clearly differentiate between these cases. We didn't like this because it meant that the user had to study a specific LaTeX notation different than the original which could lead to missleading equations if the user didn't know about all of them and also that we had to write a specification that didn't have much meaning for the purposes of this project.

In the end we had to scrape out LaTeX and take a notation that could be easily readable and already tested and know to not be ambiguous. Our team arelady knew lots of Mathematica and the syntax was thought to be used for a computer algebra system so we didn't have better options than following Mathematica.

Our grammar accepts any Mathematica expression that is either an asignment to a variable or a mathematical expression.

The context free grammar ends up being the following:

```
?start:  expr
    | NAME "=" expr -> assig

?expr: plus_substract

?plus_substract: times_divide
    | plus_substract "+" times_divide -> plus
    | plus_substract "-" times_divide -> substract

?times_divide: power
    | times_divide "*" power -> times
    | times_divide "/" power  -> divide

?power: atom
    | power "^" atom -> power

?atom: NUMBER -> number
     | call
     | list
     | indexed_symbol
     | symbol
     | "+" atom
     | "-" atom -> neg
     | atom "//" NAME -> simplification
     | "(" expr ")"



?call: "Sum" "[" expr ("," expr )*  "]" -> sum
     | "Product" "[" expr ("," expr )*  "]" -> product
     | "Power" "[" expr "," expr "]" -> power
     | "Times" "[" expr ("," expr )*  "]" -> times
     | "Plus" "[" expr ("," expr )* "]" -> plus
     | "Substract" "[" expr "," expr "]" -> substract
```

```
      | "Divide" "[" expr "," expr "]" -> divide
      | "Function" "[" expr "," expr "]" -> function

?list: "{" ((expr ",")* expr)? "}" -> tuple

?symbol: "E" -> e
       | "\\[" NAME "]" -> symbol
       | NAME       -> symbol

?indexed_symbol: NAME "[" (expr ",")* expr "]" ->
   indexed_symbol



%import common.CNAME -> NAME
%import common.NUMBER
%import common.WS_INLINE

%ignore WS_INLINE
```

Listing 9.1: CFG mathematica parser

The parsing tool used to parse the mathematica expressions is lark, this choice was made because its fast, easy to debug if the grammar has errors, and was completely made in python.

9.3 Transforming AST to computational algebraic expressions

Given the AST of each expression that defines the system wanted for the simulation, we have to convert them into symbolic expression from which we can extract every term that defines the E_L. The conversion is quite straight forward given that we already have the AST, an easy to traverse datastructure. The only caveats are that we have to convert mathematica expressions to expressions of our symbolic engine of desire, in our case sympy, and it happens that not all expressions are evaluated the same.
One of the main differences between mathematica and sympy is when they evaluate the expressions. For example:

```
>>> D[Sum[(x[i] - x[j])^2, {j, i + 1, n}], x[i]]
Sum[2*x[i] - 2*KroneckerDelta[i, j]*x[i] - 2*x[j] + 2*
```

```
   KroneckerDelta[i, j]*x[j], {j, 1 + i, n}]
```

Listing 9.2: Mathematica expression without constants

```
>>> Sum((x[i]-x[j])**2, (j,i+1,n))
Sum((x[i] - x[j])**2, (j, i + 1, n))
```

Listing 9.3: Sympy expression without constants

Both of them are quite similar but when we replace the variables by constants they don't operate in the same way

```
>>> D[Sum[(x[i] - x[j])^2, {j, i + 1, n}], x[i]] /. {i -> 1,
   n -> 3}
4*x[1] - 2*x[2] - 2*x[3]
```

Listing 9.4: Mathematica expression with constants

```
>>> Sum((x[i]-x[j])**2, (j,i+1,n)).diff(x[i]).subs(((i,1),(n
   ,3)))
Sum((2 - 2*KroneckerDelta(1, j))*(x[1] - x[j]), (j, 2, 3))
```

Listing 9.5: Sympy expression with constants

Mainly, Mathematica tries to evaluate expressions that have constant values when possible, sympy does not. A way to solve this is doing:

```
>>> Sum((x[i]-x[j])**2, (j,i+1,n)).diff(x[i]).subs(((i,1),(n
   ,3))).doit()
4*x[1] - 2*x[2] - 2*x[3]
```

Listing 9.6: Sympy expression with constants and evaluated

What *doit()* does is to evaluate the expression recursively by applying all the context from the parent expression, this leads to a higher cost to solve an expression because Mathematica does it preemptively. A way to solve this is applying *doit()* to any concrete expression like *Sum, Product* not always this operation is necessary and applying *doit()* to a long expression or with long ranges of summation or product will lead to unnecessary and expensive costs.
In the end we decided to apply *doit()* to each expression with *Sum, Product* because we couldn't find any better way.
Apart from this, sympy expressions were as expressive as Mathematica ones.

9.4 Computing Shortest terms of the expression

We have got to a spot where we have the symbolic expressions that compose the wave function, the potential and the operators, but we have not solved any of the

equations neccessary to be able to compute E_V knor $\dot{\alpha}(t)$. In these section we will be talking about how to solve these and convert the simplified minimal expressions that are needed to compute those into python code.

9.4.1 Variational Monte Carlo

When solving the Variational Monte Carlo we want to solve the following expression

$$E_V = \int p(R,a)E_L(R,a)dR \tag{9.1}$$

By constructing a random walk $R_1, R_2, ..., R_N$ form p, sampling E_L given the random walk and estimate E_V with the following

$$E_V = \frac{1}{N}\sum_{i=1}^{N} E_L(R_i,a) \tag{9.2}$$

As defined in (9.39) $E_L(R) = \dfrac{1}{\Psi_T(R,a)}\hat{H}\Psi_T(R,a)$, from (9.15) we know that $\hat{T}\Psi_T = -\dfrac{h^2}{2m}\displaystyle\sum_{a=1}^{3}\sum_{i=1}^{N_p}\left[\left(\sum_{\alpha}^{N_\alpha} T_{\text{Der}}(i,\alpha,a,2)\right) + \left(T_{\text{Der}}(i,\alpha,a,1)\right)^2\right]$. As seen previously, to compute the kinetical energy $\hat{T}\Psi_T$ we need to compute the expression $T_{\text{Der}}(i,\alpha,a,k)$, by definition (9.13) we know that

$$T_{\text{Der}}(i,\alpha,a,k) \equiv \begin{cases} T_{\text{OneBody}}(i,\alpha,a,k) & i = j \\ T_{\text{TwoBodies}}(i,\alpha,a,k) & i \neq j \end{cases} \tag{9.3}$$

where $T_{\text{OneBody}}(i,\alpha,a,1) \equiv \dfrac{\nabla_i^a f_1(i)}{f_1(i)}$, $T_{\text{OneBody}}(i,\alpha,a,2) \equiv \nabla_i^a\left[\dfrac{\nabla_i^a f_1(i)}{f_1(i)}\right]$, $T_{\text{TwoBodies}}(i,\alpha,a,1) \equiv \dfrac{\nabla_i^a f_2(\alpha,i)}{f_2(\alpha,i)}$ and $T_{\text{TwoBodies}}(i,\alpha,a,2) \equiv \nabla_i^a\left[\dfrac{\nabla_i^a f_2(\alpha,i)}{f_2(\alpha,i)}\right]$, so as we can see to compute the expression $T_{\text{OneBody}}(i,\alpha,a,k)$ and $T_{\text{TwoBodies}}(i,\alpha,a,k)$ we need to compute the expressions $\dfrac{\nabla_i^a f_1(i)}{f_1(i)}$, $\dfrac{\nabla_i^a f_2(i,j)}{f_2(i,j)}$,$\nabla_i^a\left[\dfrac{\nabla_i^a f_1(i)}{f_1(i)}\right]$ and $\nabla_i^a\left[\dfrac{\nabla_i^a f_2(\alpha,i)}{f_2(\alpha,i)}\right]$, as we can see these depend directly from the user input and are the shortest possible terms to be solved that compose the Variational Monte Carlo. Here is where the computer algebra system kicks in. We have to solve the ∇'s to be able to compute the kinetic energy of i or the kinetic energy of the interaction between i and other α's, a possible way to compute all these terms of the kinetic energy would be to solve $\dfrac{\nabla_i^a f_1(i)}{f_1(i)}$, $\dfrac{\nabla_i^a f_2(i,j)}{f_2(i,j)}$,$\nabla_i^a\left[\dfrac{\nabla_i^a f_1(i)}{f_1(i)}\right]$ and $\nabla_i^a\left[\dfrac{\nabla_i^a f_2(\alpha,i)}{f_2(\alpha,i)}\right]$ for all possible values of α and i, then whenever we want to compute one of those simply acces to the right row and colum of the matrix that stores all these computed terms, the problem with this is that the cost of computing the shortest terms would be $\mathcal{O}(N_p^2)$ but we can do better without additional costs, we can do the following

- compute $\dfrac{\nabla_i^a f_2(i,\alpha)}{f_2(i,\alpha)}$ assuming $i \neq \alpha$

- compute $\nabla_i^a \left[\frac{\nabla_i^a f_2(i,\alpha)}{f_2(i,\alpha)} \right]$ assuming $i \neq \alpha$
- compute $\frac{\nabla_i^a f_1(i)}{f_1(i)}$ assuming $i = \alpha$
- compute $\nabla_i^a \left[\frac{\nabla_i^a f_1(i)}{f_1(i)} \right]$ assuming $i = \alpha$

These assumptions must be specified to the algebra system not to get any Kronecker Delta $\delta_{i\alpha}$, without the assumptions the terms would still be correct but the expressions generated would be sub-optimal, once the expressions are generated, we simplify the expressions and compute the common sub-expression from which we will produce the runnable optimal python code. Once the python code from the lower expressions is made, converting the remaining expressions into python code until reaching E_L is quite trivial.Once E_L is computed, to get E_V we would need to sample E_L with a random path made from p.
We have the code to compute E_L but we are missing out on how to produce the random walk that follows the distribution p, to compute it we have to solve $p(R,a) = \frac{|\Psi(R,a)|^2}{\int |\Psi(R,a)|^2 dR}$ which means we should solve the integral, but if we look more in depth in the metropolis algorithm **??** we can see that in reality what we need to compute is $\frac{p(X',a)}{p(X,a)} = \frac{|\Psi(R',a)|^2 / \int |\Psi(R,a)|^2 dR}{|\Psi(R,a)|^2 / \int |\Psi(R,a)|^2 dR} = \frac{|\Psi(R',a)|^2}{|\Psi(R,a)|^2}$, so there is no need to compute the integral, now the other problem is the order we solve the products and divisions of exponentials, if we did all the products and then divided both terms this would lead to possible numerical errors given the nature of this problem, away to mitigate the numerical error would be expanding it as following $\frac{p(X',a)}{p(X,a)} = \prod_{i=1}^{N_p} \frac{f_1(X,a,i)}{f_1(X,a,i)} \prod_{i=1}^{N_p} \prod_{\alpha=i+1}^{N_p} \frac{f_2(X,a,i,\alpha)}{f_2(X,a,i,\alpha)}$ but knowing Jastrow terms are made of exponentials an even better solution would be using the natural logarithm so $\ln\left(\frac{p(X',a)}{p(X,a)}\right) = \sum_{i=1}^{N_p} \ln(f_1(X,a,i)) - ln(f_1(X',a,i)) \sum_{i=1}^{N_p} N_p \sum_{\alpha=i+1}^{N_p} \ln(f_2(X,a,i,\alpha)) - \ln(f_2(X',a,i,\alpha))$, so what we need is to compute the natural logarithms from the Jastrow functions, simplify the resulting expressions, compute the common sub expression and generate the python code from it. Once we have these functions, generating code until reaching $\frac{p(X',a)}{p(X,a)}$ is trivial.
Now we have the tools to make it possible to compute E_V by generating a random walk from p and sampling E_L.

9.4.2 Time-Dependent Variational Monte Carlo

When searching for the ground state using Time-Dependent Variational Monte Carlo we have to solve $\dot{c(t)}$ and E_V, we have already solved E_v, so the reamining part to solve comes from (8.58) where $\dot{c}_\alpha(t) = (M)^T \cdot (M \cdot (M)^T)^{-1} \cdot V$, i.e., computing the pseudo inverse of M and V. Let M and V be the matrices

$$M = \begin{pmatrix} M_{11} & M_{12} & \cdots & M_{1N} \\ M_{21} & M_{22} & \cdots & M_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ M_{N_k 1} & a_{N_2} & \cdots & M_{N_k N_\alpha} \end{pmatrix} V = \begin{pmatrix} V_1 \\ V_2 \\ \vdots \\ V_{N_\alpha} \end{pmatrix}$$

and given (8.52) and (8.51) we have that

$$V_k = -\int \Psi^\dagger \mathcal{O}_k(R) \mathcal{H} \Psi dR = -\int \mathcal{O}_k(R) E_L(R) dR \tag{9.4}$$

$$M_{k\alpha} = \int \Psi^\dagger \mathcal{O}_k(R) \frac{\partial \Psi}{\partial c_\alpha} dR \tag{9.5}$$

From here we can see that we have to follow the same procedure as before for handling these integrals, we have to compute the expressions of $\mathcal{O}_k$ and $\frac{\partial \Psi(R)}{\partial c_\alpha(t)}(\Psi(R))^{-1}$.

Getting the expressions from $\mathcal{O}_k$ is trivial because we've already have them from the user input so the only thing needed is to simplify the expressions and compute the common sub-expression to generate the python code.

For $\frac{\partial \Psi(R)}{\partial c_\alpha(t)}(\Psi(R))^{-1}$ it gets a little harder, its simmilar to what we did with the Variational Monte Carlo, given

$$\frac{\partial \Psi(R)}{\partial c_\alpha(t)}(\Psi(R))^{-1} = \sum_{i=1}^{N_p} \sum_{\alpha=1}^{N_\alpha} \frac{\partial \Psi(\alpha, i)}{\partial c_\alpha(t)}(\Psi(\alpha, i))^{-1} \tag{9.6}$$

When reaching the expression $\frac{\partial \Psi(\alpha, i)}{\partial c_\alpha(t)}(\Psi(\alpha, i))^{-1}$ given the nature of this exponential function the fraction lets us cancel terms giving a more robust expression. So what we really need to compute is simply $\frac{\partial \Psi(\alpha, i)}{\partial c_\alpha(t)}(\Psi(\alpha, i))^{-1}$ from the expression with all the canceled terms, simplify it and compute the common sub expression, from here generate the python function that will handle this compute and then $\frac{\partial \Psi(R)}{\partial c_\alpha(t)}(\Psi(R))^{-1}$.
From here we have the tools to compute both M and V , i.e. , we have a way to solve $\dot{c}(t)$.

9.5 Generate higher level code

From the previous section we have seen a way to compute all the necessary functions that compose the expressions E_v and $\dot{\alpha}(t)$ that are needed to solve the problems from both the Variational Monte Carlo and the time-dependent Variational Montecarlo. Yet, we haven't talked about how to get E_v and $\dot{\alpha}(t)$ from the previous steps. In this section we will be going more in depth on how to solve both problems given what we have already computed.

9.5.1 Variational Quantum Montecarlo

As previously mentioned in **??** Variational Quantum Montecarlo works as following, compute a random walk from $p(R, c_\alpha) = \frac{\Psi(R, c_\alpha)}{\int |\Psi(R, c_\alpha)|^2 dR}$, from this random walk $R_{\text{walk}} = R_1, R_2, ..., R_N$ we will be using it to solve the following quadrature $E_v = \int p(R, c_\alpha) E_l(R, c_\alpha)$ and by the principle of importance sampling (**??**) we know that $E_v = \frac{1}{N} \sum_{R_i \in R_{\text{walk}}} E_L(R_i, c_\alpha)$. So in the end we would have an algorithm that would look like the following, given (**??**)

Algorithm 5 Computing variational energy

input : The number of particles N_p and number of iterations N_{it}

output: returns E_v with a standard error σ

s2 = 0 s = 0 R = pick a random stating state

while $N_{it} > 0$ **do**

  for $i \in 1, 2, ..., N_p$ **do**

    $z = U(0, 1)$

    $R'_i = R_i + N(O_3, \Delta I_3)$

    if $|\Psi(R', c)|^2 / |\Psi(R, c)|^2 < z$ **then**

      $R'_i = R_i$

    $s2 = s2 + E_l(R', c)^2$

    $s = s + E_l(R', c)$

  end

  $N_{it} = N_{it} - 1$

end

$E_v = \frac{s}{3N_p}$

$\sigma = \sqrt{\frac{s2 - s^2}{N_p N_{it}}}$

return E_v

This is the simplest and fastest implementation of all, but the main problem it has is that the order we the state is always fixed, an alternative that does not have any bias would be the following

Algorithm 6 Computing variational energy with no particle order bias

input : The number of particles N_p and number of iterations N_{it}

output: returns E_v with a standard error σ

s2 = 0 s = 0 R = pick a random stating state

while $N_{it} > 0$ **do**

  for $i \in$ *Permutation*$(1, 2, ..., N_p)$ **do**

    $z = U(0, 1)$

    $R'_i = R_i + N(O_3, \Delta I_3)$

    if $|\Psi(R', c)|^2 / |\Psi(R, c)|^2 < z$ **then**

      $R'_i = R_i$

  end

  $s2 = s2 + E_l(R', c)^2$

  $s = s + E_l(R', c)$

  $N_{it} = N_{it} - 1$

end

$$E_v = \frac{s}{3N_p}$$

$$\sigma = \sqrt{\frac{s2 - s^2}{N_p N_{it}}}$$

return E_v, σ_α

By not having a strict order on the choice of the next state the chances of having a biased result are much lower but we are doing it at the expense of accessing to the particles randomly which will lead to worse performance once we have a high number of particles.

For the Time-Dependent Variational Monte Carlo we need to also compute $\dot{c_\alpha}(t)$, a way to do it would be to sample the previously mentioned matrices M and V using the same algorithm as for the Variational Monte Carlo. A possible algorithm would look like the following.

Algorithm 7 Computing $\dot{c}$

input : The number of particles N_p and number of iterations N_{it}

output: returns $\dot{c}$

$S(E_l) = 0$

$S(E_l^2) = 0$

$\forall k : S(\mathcal{O}_k) = 0$

$\forall \alpha : S(\frac{\partial \Psi}{\partial c_\alpha}(\Psi)^-1) = 0$

R = pick a random stating state

while $N_{it} > 0$ **do**

  for $i \in 1, 2, ..., N_p$ **do**

    $z = U(0, 1)$

    $R_i' = R_i + N(0_3, \Delta I_3)$

    if $|\Psi(R', c)|^2 / |\Psi(R, c)|^2 < z$ **then**

      $R_i' = R_i$

    $\forall \alpha : S(\frac{\partial \Psi}{\partial c_\alpha}(\Psi)^{-1}) = S(\frac{\partial \Psi}{\partial c_\alpha}(\Psi)^{-1}) + \frac{\partial \Psi(R', c)}{\partial c_\alpha}(\Psi(R', c))^{-1}$

    $\forall k : S(\mathcal{O}_k) = S(\mathcal{O}_k) + \mathcal{O}_k(R', c)$

    $S(E_l^2) = S(E_l^2) + E_l(R', c)^2$

    $S(E_l) = S(E_l) + E_l(R', c)$

  end

  $N_{it} = N_{it} - 1$

end

$E_v = \frac{S(E_l)}{N_{it} N_p}$

$\forall k, \alpha : M_{k\alpha} = (S(\mathcal{O}_k))(\frac{\partial \Psi}{\partial c_\alpha}(\Psi)^{-1}) / (N_{it} N_p)$

$\forall k, \alpha : M_{k\alpha} = (S(\mathcal{O}_k S E_l)) / (N_{it} N_p)$

$\forall k : V_k = (S(\mathcal{O}_k) S E_l) / (N_{it} N_p)$

$\dot{c} = (M)^T \cdot (M \cdot (M)^T)^{-1} \cdot V$

return $\dot{c}$

9.6 Integration Tests

Once the previous ensemble of complex methods was made had no idea if the simulations we run were giving the expected values we wanted. A way to solve this would be by comparing our results with anallytically correct known ones, the main problem is that once we crank up the number of particles the problem gets so complex that no analytical solution is known.

9.6.1 Many-Body Harmonic Oscillator

We already knew the analytical solution of a well known and defined problem, the quantum harmonic oscillator. For an harmonic oscillator with the following structure:

$$V = w\sum_{i=1}^{N_p} x_i^2 + y_i^2 + z_i^2 + k\sum_{i=1}^{N_p}\sum_{j=i+1}^{N_p} (x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2 \tag{9.7}$$

$$\Psi = \exp^{c_0}\exp\left\{-\gamma\sum_{i=1}^{N_p} x_i^2 + y_i^2 + z_i^2 - \alpha\sum_{i=1}^{N_p}\sum_{j=i+1}^{N_p} (x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2\right\} \tag{9.8}$$

$$E_l = -Dh(\sum_{i=1}^{N_p}(\frac{\partial\Psi}{\partial x_i} + \frac{\partial\Psi}{\partial y_i} + \frac{\partial\Psi}{\partial z_i})(\Psi)^{-1} + V \tag{9.9}$$

When $N_p = 2$, $Dh = \frac{1}{2}, w = 2, k = 1$ we know the exact analytical solution of $\lim_{t\to\infty}\gamma(t) = 1$, $\lim_{t\to\infty}\alpha(t) = \frac{1}{\sqrt{2}} - \frac{1}{2}$ and $\lim_{t\to\infty} c_0(t) = -E_0 = 3(1+\sqrt{2})$ for wich we can check the correctness of our method.

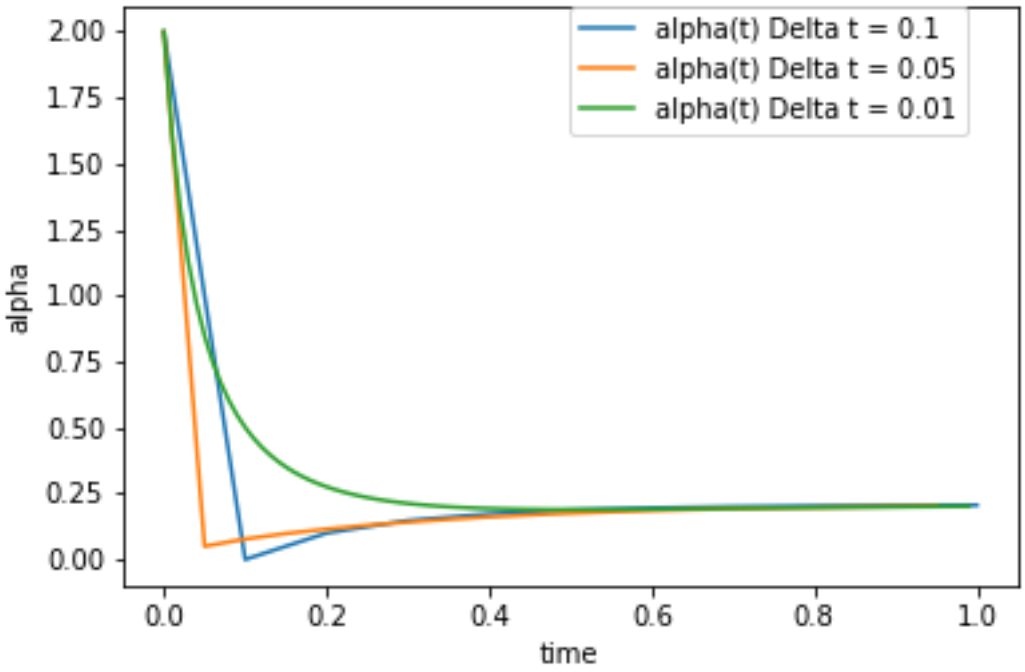


Figure 9.7: $\alpha(t)$ for different Δt values when $N_p = 2$

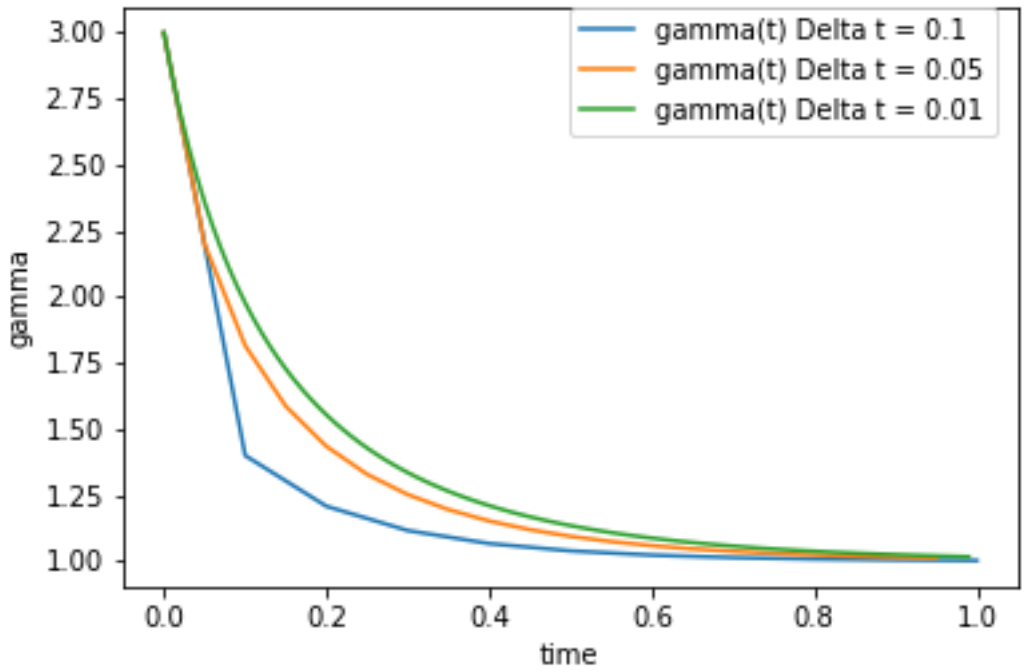


Figure 9.8: $\gamma(t)$ for different Δt values when $N_p = 2$

As it can be seen in (9.8) and (9.7) our method reaches the configuration of the ground state. We have also got average of $E_v = 7.24263$ with a standard error of

$3.1736e^{-16}$, therefore, our method follows an analytically correct solution.

When $N_p = 5$, $Dh = \frac{1}{2}, w = 2, k = 1$ we know of an analytical solution to $\langle E_l \rangle = \frac{3}{2}(20\alpha + \frac{1}{\gamma} + 5\gamma + \frac{14}{5\alpha + \gamma})$ and $\min\{\frac{3}{2}(20\alpha + \frac{1}{\gamma} + 5\gamma + \frac{14}{5\alpha + \gamma}) : \alpha \in \mathcal{R}, \gamma \in \mathcal{R}\}$ happens when $\gamma = 1$ and $\alpha = \frac{\sqrt{\frac{7}{2}}}{5} - \frac{1}{5}$. with the exact solution $E_0 = 3 + 6\sqrt{14}$.

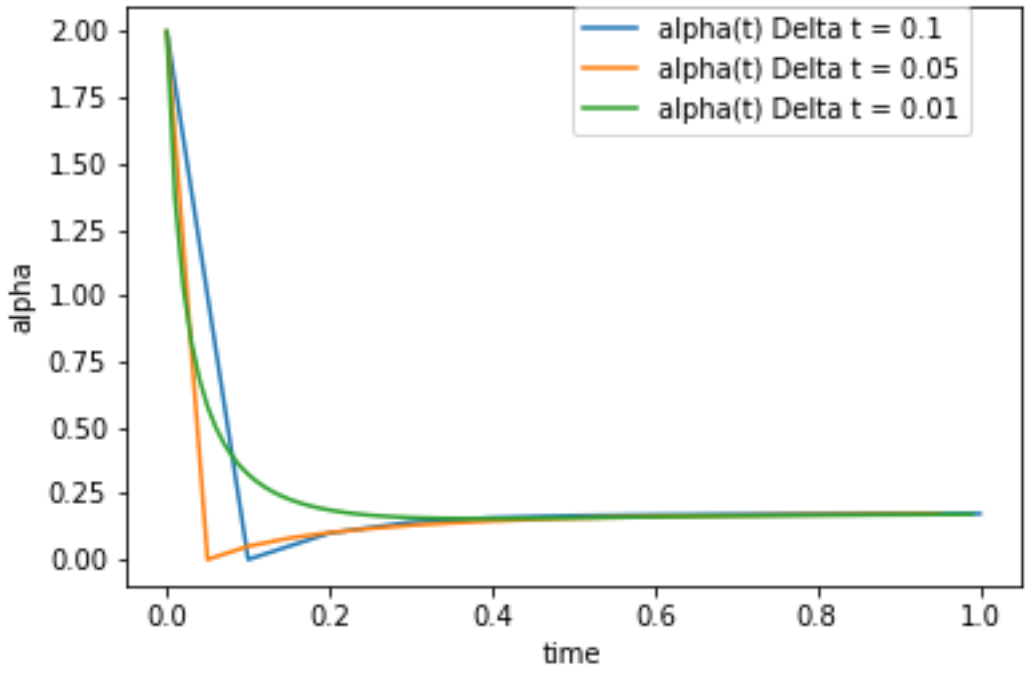


Figure 9.9: $\alpha(t)$ for different Δt values when $N_p = 5$

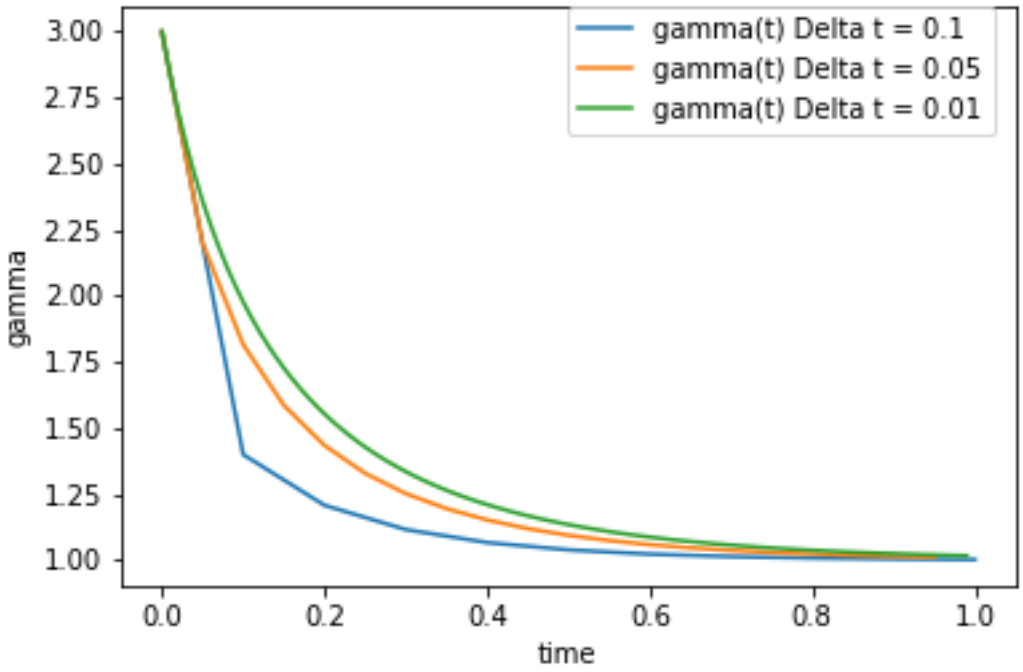


Figure 9.10: $\gamma(t)$ for different Δt values when $N_p = 5$

As it can be seen in (9.10) and (9.9) both results follow the analytical solution given. The results are pretty similar to when we just had 2 particles. Our average E_0 when doing random walks of 10000 steps is 25.6125 with a standard error of $3.57061e^{-16}$.

For the Simulated Annealing we reach a suboptimal solution where $E_0 = 30.26$ and the progression of E_v looks as following

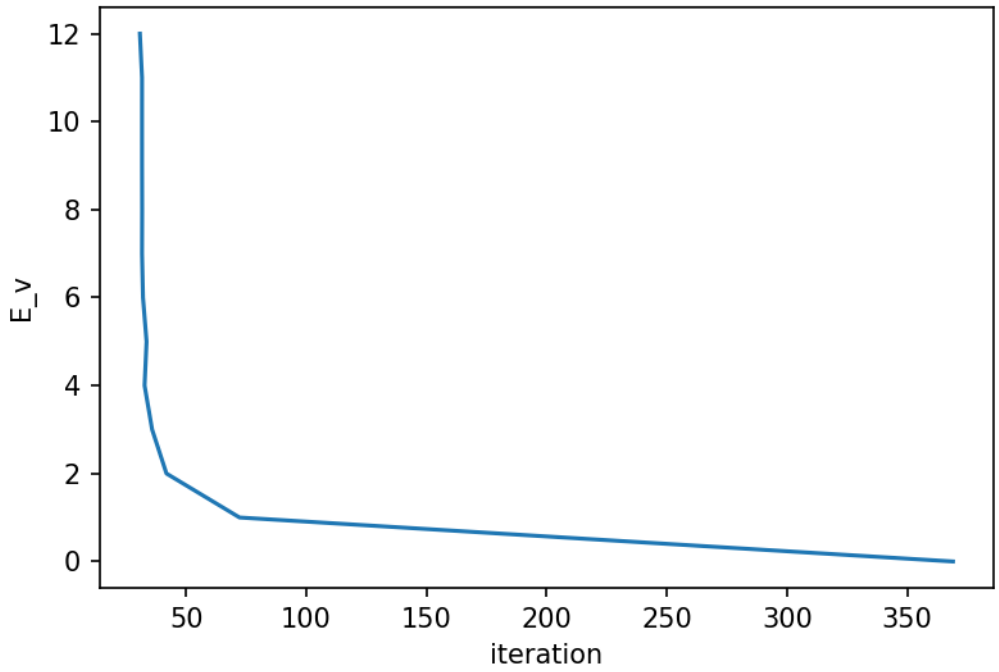


Figure 9.11: Progression of objective function E_v on Simulated Annealing when $N_p = 5$

It must be stated that we tried several runs trying to find the best configuration of neighbor function and temperature function and still we couldn't find anything that could get closed than $E_0 = 30.03$ and on average the runs lasted for 2 minutes. Compared to the previous run that only lasted 10.2381 seconds this is definitely an order of magnitude difference. All runs using the Variational Monte Carlo to sample E_v had the same configuration, therefore this speed improvement is just from the search of the parameters. We can state that simulated annealing might give nearby values but it is extremely complicated to give exact values.

9.7 Generating llvm code

The python code previously generated if run on the back-end would lead to poor performance given its all made in python, the python code is great for testing we are in need of fast code given that simulations may last for a long time so minor speedups can have a major influence on the experience of the user. The only reason our we are bounded to slow runs is that our efficiently structured code runs on python, an interpreted language that has lots of overhead for our intended purposes.

Nevertheless, python has a tool called numba that transaltes python code to llvm code, a lower level inter-machine code that is capable of performing high order of magnitude optimizations that will lead to faster runtimes. Numba is also capable of multithreading, vectorizing functions, and even converting code into runnable GPU

cuda code. Numba works on the base of jit optimizations so we are abled to inline the smaller expressions computed and also parametrize them at the same time which makes that our tool can produce an even faster code that non jit compilers could do. The only caveat jit has is that the first time you call a function will not be as fast as later runs but still given we will be calling the same functions of orders of $\mathcal{O}(n)$, $\mathcal{O}(n^2)$ or even $\mathcal{O}(n^3)$. The jit compilations and optimizations during run-time shouldn't be noticeable compared to the time spend on the simulation.

We wanted to proof if this non ordinary methodology would be up to the performance standards of the tools dictated by the research team, fortran is one of the main choices when making a tool for scientific of the research team so we decided to make a comparison of a simple Monte Carlo algorithm to compute π comparing numba, pure python and fortran. The code used is the following:

```
import random


def monte_carlo_pi(nsamples):
    acc = 0
    for i in range(nsamples):
        x = random.random()
        y = random.random()
        if (x ** 2 + y ** 2) < 1.0:
            acc += 1
    return 4.0 * acc / nsamples
```

Listing 9.7: Pure Python code of a Monte Carlo algorithm to compute π

```
from numba import jit
import random


@jit(nopython=True)
def monte_carlo_pi_numba(nsamples):
    acc = 0
    for i in range(nsamples):
        x = random.random()
        y = random.random()
        if (x ** 2 + y ** 2) < 1.0:
            acc += 1
    return 4.0 * acc / nsamples
```

Listing 9.8: Python code wrapped in numba of a Monte Carlo algorithm to compute π

```
subroutine monte_carlo_pi_fortran(n, pi)
    integer, intent(in) :: n
    real,    intent(out):: pi
    integer :: k, count
    real :: x, y
    count = 0
    do k = 1, n
        call random_number(x)
        call random_number(y)
        if (x*x + y*y < 1.0) count = count + 1
    end do
    pi = 4 * real(count) / n
end subroutine monte_carlo_pi_fortran
```

Listing 9.9: Fortran code of a Monte Carlo algorithm to compute π

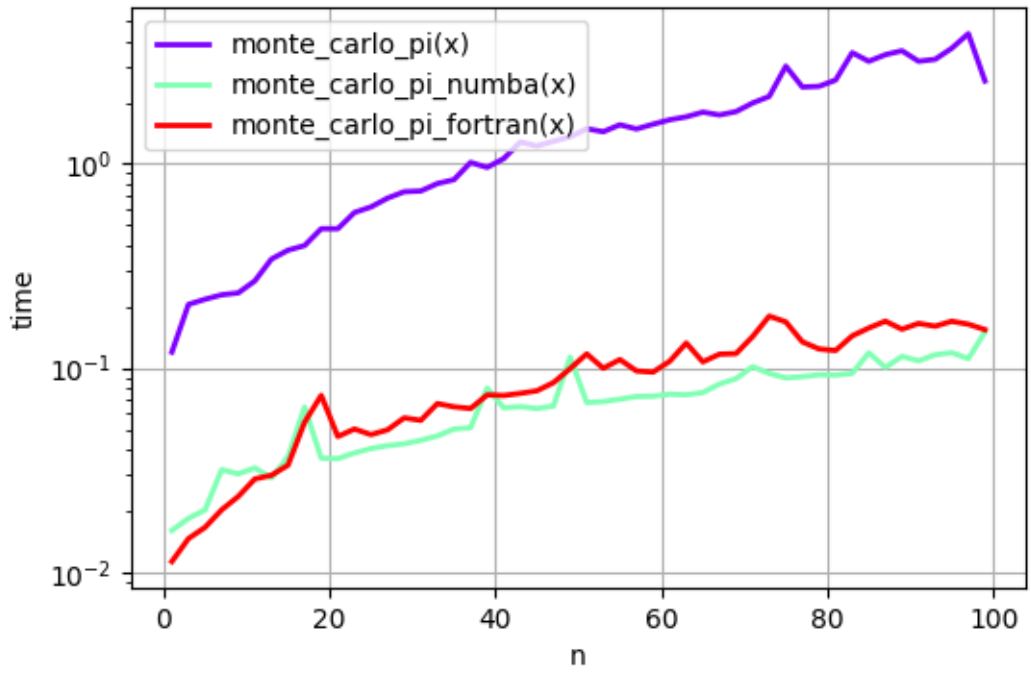


Figure 9.12: Comparison of a montecarlo run: pure python, numba, fortran

Fortran code has been compiled with *-O2*. As it can bee seen in (9.12) we have nearly a 30 fold speed up compared to pure python when using numba, we can also see that numba is as fast or slightly faster than fortran code. We have even found cases in our algorithms where the code made by numba was 100 times faster that pure python code.

Therefore, numba is a great option for stochastic algorithms compared to our standards of running the stochastic algorithms in fortran.

Chapter 10

Conclusions

We have completed the design, development and implementation of a tool that allows the Barcelona Quantum Monte Carlo group to execute Variational Monte Carlo simulations for quantum many body systems. We have implemented a method of accurately finding the ground state of a given system of bosonic particles by optimizing the parametrization of a given trial wave function having as objective function the energy of the system, thus obtaining information about the quality of the functional space specified by the user for the physical system. Our main objectives that we have accomplished are the following:

1. Allowing the users to fully specify the model of interaction between particles.
2. Being abled to handle large ammounts of particles with ease.
3. Being able to specify at user level the set of relevant operators (equivalent in quantum mechanics to the corresponding property in a classical system) for which we are interested in obtaining information.
4. Executing Monte Carlo Variational simulations, for any value of the parameters withing the defined functional space defined by the user.
5. Extracting statistical information obtained from the simulation.
6. Visualization of the statistical information obtained from the simulation.
7. Develop several methods of Stochastic optimization to optimize the parametrization of the wave function.
   (a) Optimizing the wave function by means of the Time Dependent Variational Monte Carlo.
   (b) Optimizing the wave function by means of Simulated Annealing.
8. Being able to compare the new method, the time-dependent VMC optimization, with other stochastic methods.
9. Developed high performance code for the compute of the VMC and the stochastic methods.

10. Develop an interface for the user to define all possible configurations needed for the previous objectives

11. Develop tools for the visualization and representation of the statistical information collected by the implemented methods, both in the optimization process and in the final ground state.

From here we have reached several conclusions, we have obtained evidence that the new method shows a high improvement in both performance and precision (for the problem considered) compared to Simulated Annealing.

We have produced a tool that provides high performance code generation from the given expressions by the used. We have applied non ordinary optimizations done at a symbolic level like canceling of terms, simplification, extracting common factors, applying assumptions to the engine and extracting common sub expressions to produce code that minimizes the numeric computational cost, as well as ordinary optimizations such as multi-threading and SIMD vectorization.

10.1 Technical competences

CCO1.1: Evaluate the computational complexity of a problem, know algorithmic strategies that can lead to its resolution, and recommend, develop and implement the one that guarantees the best performance in accordance with the established requirements. [In depth]

Although there has not been a thorough analysis on the computational complexity, it has been shown the possible alternatives to metropolis-hastings and why metropolis-hastings is the most efficient feasible solution for high dimensional data.

CCO2.1: Demonstrate knowledge of the fundamentals, paradigms and techniques of intelligent systems, and analyze, design and build computer systems, services and applications that use these techniques in any field of application. [Little bit]

We have developed stochastic optimization methods to optimize the wave function to find the ground-state of the system.

CCO2.3: Develop and evaluate interactive and presentation systems complex information, and its application to problem-solving computer-person interaction design. [In depth]

Our user interface is capable of handling complex mathematical equations, represent them, let the user select the parameters of the ground-state search for both methods and see in real time the complex information of the simulation.

CO2.4: Demonstrate knowledge and develop computer learning techniques; design and implement applications and systems that use them, including those dedicated to the automatic extraction of information

and knowledge from large volumes of data. [Quite]

Many body problems tend to work with high dimensional spaces which leads to interacting with a huge amount of data, it must be stated that this data is neither given know returned, the data is generated from the knowledge of the probability distribution of the wave-function and used for the stochastic optimization, i.e, from the knowledge of the wave function we can compute a random walk used for the stochastic methods.

CCO3.1: Implement critical code following execution time, efficiency and security criteria. [In depth]

This was one of our main objectives when designing the tool, it is a must to have code that is highly efficient given that the simulations may even run for spans of a month. Even minor speedups may mean a week less of compute. That is why part of the development was focused on the generation of optimal code from the given expressions by the user.

10.2 Future Work

Extend the current method to handle periodic boundary conditions
When dealing with an infinite number of particles, we have to use a different method that the actual, the way to approximate this case is to represent a finite set of particles in a cell and then have images of these in the nearby cells following a periodic condition. The interaction of two particles would be defined by the interaction of one particle with the nearest image of the other.

Extend the current method of Wave-Function optimization to spin models
Right now our equations follow the structure of a bosonic particles, to take into account fermion systems we need a way to represent the spin and isospin degrees of freedom, this would mean much more compute and having also a Slater determinant at each step of the metropolis algorithm.

Generate code different from llvm
We are using llvm code for the simulations but no clear comparison has been done on the performance we get from Numba generating llvm compared to the performance we could get it we generated Fortran code given the tools Sympy gives us.

Bibliography

[1] Giuseppe Carleo. "Spectral And Dynamical Properties Of Strongly Correlated Systems". In: (), p. 152.

[2] Giuseppe Carleo et al. "Light-cone effect and supersonic correlations in one- and two-dimensional bosonic superfluids". In: *Physical Review A* 89.3 (Mar. 27, 2014), p. 031602. ISSN: 1050-2947, 1094-1622. DOI: 10.1103/PhysRevA.89.031602. arXiv: 1310.2246. URL: http://arxiv.org/abs/1310.2246 (visited on 03/16/2020).

[3] Giuseppe Carleo et al. "Localization and Glassy Dynamics Of Many-Body Quantum Systems". In: *Scientific Reports* 2.1 (Dec. 2012), p. 243. ISSN: 2045-2322. DOI: 10.1038/srep00243. arXiv: 1109.2516. URL: http://arxiv.org/abs/1109.2516 (visited on 03/16/2020).

[4] *Company Salaries*. Glassdoor. Library Catalog: www.glassdoor.co.uk. URL: https://www.glassdoor.co.uk/Salaries/index.htm (visited on 03/16/2020).

[5] *Lloguers Barcelona per districtes i barris*. Habitatge. Library Catalog: habitatge.gencat.cat. URL: http://habitatge.gencat.cat/ca/dades/estadistiques_publicacions/indicadors_estadistiques/estadistiques_de_construccio_i_mercat_immobiliari/mercat_de_lloguer/lloguers-barcelona-per-districtes-i-barris/ (visited on 03/09/2020).

[6] *Salary: Backend Engineer in Spain*. Glassdoor. Library Catalog: www.glassdoor.co.uk. URL: https://www.glassdoor.co.uk/Salaries/spain-backend-engineer-salary-SRCH_IL.0,5_IN219_KO6,22.htm (visited on 03/09/2020).

[7] *Salary: Computational Physicist in Spain — Glassdoor.co.uk*. Library Catalog: www.glassdoor.co.uk. URL: https://www.glassdoor.co.uk/Salaries/spain-computational-physicist-salary-SRCH_IL.0,5_IN219_KO6,29.htm (visited on 03/09/2020).

[8] *Salary: Front End Engineer in Spain — Glassdoor.co.uk*. Library Catalog: www.glassdoor.co.uk. URL: https://www.glassdoor.co.uk/Salaries/spain-front-end-engineer-salary-SRCH_IL.0,5_IN219_KO6,24.htm (visited on 03/09/2020).

[9] *Salary: Machine Learning Engineer in Spain*. Glassdoor. Library Catalog: www.glassdoor.co.uk. URL: https://www.glassdoor.co.uk/Salaries/spain-machine-learning-engineer-salary-SRCH_IL.0,5_IN219_KO6,31.htm (visited on 03/09/2020).

[10] *Salary: Programmer in Spain — Glassdoor.co.uk*. Library Catalog: www.glassdoor.co.uk. URL: https://www.glassdoor.co.uk/Salaries/spain-programmer-salary-SRCH_IL.0,5_IN219_KO6,16.htm (visited on 03/09/2020).

[11] *Salary: System Engineer in Spain — Glassdoor.co.uk*. Library Catalog: www.glassdoor.co.uk. URL: https://www.glassdoor.co.uk/Salaries/spain-system-engineer-salary-SRCH_IL.0,5_IN219_KO6,21.htm (visited on 03/09/2020).

[12] *Salary: Technical Writer in Spain*. Glassdoor. Library Catalog: www.glassdoor.co.uk. URL: https://www.glassdoor.co.uk/Salaries/spain-technical-writer-salary-SRCH_IL.0,5_IN219_KO6,22.htm (visited on 03/09/2020).

[13] *Salary: Tests engineer in Spain — Glassdoor.co.uk*. Library Catalog: www.glassdoor.co.uk. URL: https://www.glassdoor.co.uk/Salaries/spain-tests-engineer-salary-SRCH_IL.0,5_IN219_KO6,20.htm (visited on 03/09/2020).