New efficient integral algorithms for quantum chemistry

Jaime Axel Rosal Sandberg

Doctoral Thesis in Theoretical Chemistry and Biology
School of Biotechnology
Royal Institute of Technology
Stockholm, Sweden 2014

ISBN 978-91-7595-237-6
ISSN 1654-2312
TRITA-BIO-Report TRITA-BIO Report 2014:13
Printed by Printed by Universitetservice US AB,
Stockholm, Sweden, 2014

To my family

Abstract

The contents of this thesis are centered in the developement of new efficient algorithms for molecular integral evaluation in quantum chemistry, as well as new design and implementation strategies for such algorithms aimed at maximizing their performance and the utilization of modern hardware.

This thesis introduces the K4+MIRROR algorithm for 2-electron repulsion integrals, a new ERI integral scheme effective for both segmented and general contraction, which surpasses the performance of all previous ERI analytic algorithms published in the literature. The performance of the K4 kernel contraction scheme is further boosted by the use of some new recurrence relations, CDR/AERR family of recurrences, and the algorithms is further refined for spherical GTOs with the also new SKS method.

New prescreening methods for two-electron integrals are also derived, allowing a more consistent methodology for discarding negligible ERI batches. This thesis introduces new techniques useful to pack integrals efficiently and better exploit the underlying modern SIMD or stream processing hardware. These algorithms and methods are implemented in a new library, the Echidna Fock Solver, a hybrid parallelized module for computing Coulomb and Exchange matrices which has been interfaced to the Dalton suite of quantum chemistry programs. Self-Consistent Field and Response Theory calculations in Dalton using the new EFS library are substantially accelerated, also enabling for the first time the use of general contraction basis sets as default basis for extended calculations.

The thesis further describes the derivation and implementation of an integral algorithm for evaluating the matrix elements needed for the recently introduced QM/CMM method, for which many of the techniques previously derived are also used, along with a suitable prescreening method for the matrix elements. The implementation is also interfaced to the Dalton quantum chemistry program, and used in production calculations.

The last chapter of the thesis is devoted to the derivation of a general analytic solution for type-II Effective Core Potential integrals, arguably one of the most troublesome molecular integrals in quantum chemistry. A new recurrence is introduced for the integrals, and a screening method is presented. Based on these results, a new efficient algorithm for computing type-II ECPs is also described.

Preface

The work presented in this thesis was carried out at the Department of Theoretical Chemistry, School of Biotechnology of the Royal Institute of Technology, Stockholm, Sweden.

List of papers included in the thesis

Paper I: An algorithm for the efficient evaluation of two-electron repulsion integrals over contracted Gaussian-type basis functions
J.A. Rosal Sandberg and Z. Rinkevicius
J. Chem. Phys., **137**, 234105 (2012)
DOI: 10.1063/1.4769730

Paper II: Density functional theory/molecular mechanics approach for linear response properties in heterogeneous environments
Z. Rinkevicius, X. Li, **J.A. Rosal Sandberg**, K. Mikkelsen and H. Ågren
J. Chem. Theor. Comp., **10**, 989-1003 (2014)
DOI: 10.1021/ct400897s

Paper III: Non-linear optical properties of molecules in heterogeneous environments: A quadratic density functional/molecular mechanics response theory
Z. Rinkevicius, X. Li, **J.A. Rosal Sandberg** and H. Ågren
Phys. Chem. Chem. Phys., **16**, 8981-8989 (2014)
DOI: 10.1039/C4CP00992D

Paper IV: Efficient time dependent density functional theory calculations using general contraction basis sets
J.A. Rosal Sandberg, Z. Rinkevicius and H. Ågren
Submitted

Paper V: New recurrence relations for analytic evaluation of two-electron repulsion integrals over highly contracted gaussian-type orbitals
J.A. Rosal Sandberg and Z. Rinkevicius
In manuscript

Comments on my contribution to the included papers

My contributions to papers I and V include the analysis, design and implementation of the K4+MIRROR, CDR/AERR and SKS algorithms, as well as their implementation, testing and optimization. I am responsible for a majority of the text in both manuscripts, as well as the benchmark calculations in the supplementary material of paper I.

My contributions to papers II and III are the analysis, design and implementation of the QM/CMM integrals as well as the subsequent hybrid paralellization and optimization of the integral module.

My contributions to paper IV are in the form of the already mentioned K4+MIRROR and SKS algorithms, and in the design, implementation and hybrid parallelization of the EFS library which is used to evaluate the Coulomb and Exchange matrices, as well as the interfacing to the Dalton program for SCF and linear response.

Acknowledgments

I would like first and foremost to express my most sincere gratitude to my supervisor, professor Zilvinas Rinkevicius. His support, advice and guidance and the many discussions held have been a rich source of ideas and have deeply widened my perspectives about science.

I would also like to specially thank my co-supervisor and head of the Division of Theoretical Chemistry and Biology, professor Hans Ågren, for giving me the invaluable oportunity to pursue my PhD at his department, as well as for his support and patience.

I am very grateful to professor Yi Luo and professor Cao for their invitation to Xiamen and their excellent hosting in China.

Special thanks to professors Olav Vahtras, Faris Gel'mukhanov, Mårten Ahlquist, Boris Minaev, Yaoquan Tu, Michael Schliephache, Berk Hess, Erwin Laure and Bo Kågström for many enriching discussions; to Radovan Bast for his help and to Arul Murugan for his advice.

Thanks to my colleagues past and present: Bogdan Frecus, Cui Li, Guanying Chem, Fu Qiang, Rocio Marcos, Johannes Niskanen, Rui Zhang, Sai Duan, Shijing Tan, Robert Zalesny, Guangjun Tin, Xue Liqin, Guangping Zhang, Zhuxiz Zhang, Rocío Sánchez de Armas, Ce Song, Chunze Yuan, Lu Sun, Ignat Harczuk, Li Gao, Irina Osadchunk, Asghar Jamshidi Zavaraki, Junfeng Li, Jing Huang, Yongfei Ji, Kayathri Rajarathinam, Guanglin Kuang, Hongbao Li, Lijun Liang, Xin Li, Lu Sun, Matti Vapa, Yong Ma, Rafael Carvalho Couto, Tuomas Loytynoja, Xu Wang, Yan Wang, Wei Hu, Xiao Cheng, Xianqiang Sun, Xinrui Cao, Ke-Yan Lian, Wing Wang, Zhengzhong Kang, Zuyong Gong and Vinicius Vaz da Cruz for their contribution to a friendly and relaxed working environment.

An extra round of thanks to Bogdan, Murugan, Radovan, Robert, Zilvinas and the rest of the bowling team for the many priceless moments, and Robert in particular for not allowing me in many occasions to lose the match, my money and my honor.

And finally, no acknowledgements section would be complete if I didn't specially recognize the positive attitude and good mood of all the coleagues with whom we've shared the office: Lu, Li, Xin, Johannes, Ignat and Vinicius.

CONTENTS

CHAPTER 1

Introduction

"Science is what we understand well enough to explain to a computer.
Art is everything else we do."
Donald Knuth

Computational chemistry constitutes one of the most profound and spectacular successes in the modern scientific endeavour. Thanks to the advent of modern computers, computational chemistry methods like quantum chemistry and classical molecular dynamics can now be routinely applied to a vast array of problems arising in multiple scientific disciplines, from engineering to biophysics. The theoretical grounds of some of its more important methodology has been recognized as worthy of the Nobel prize in chemistry on two occasions; first in 1998 to Walter Kohn for *his development of Density Functional Theory* and John A. Pople for *his development of computational methods in quantum chemistry*[1], and more recently in 2013 to Martin Karplus, Michael Levitt and Arieh Warshel for *the development of multiscale models for complex chemical systems*[2]. Many new interdisciplinary fields which were born out of the convergence of chemistry, physics and biology at the nanoscale use in-silico simulations as a primary tool for analysis and prediction.

Quantum chemistry is the application of quantum physics to the study of chemical systems. The underlying theoretical framework -quantum theory- tends to be very counterintuitive to many scientists, yet it is the single most successful theory in the history of science. Its success lies in great measure in the generality of its principles, but also in the astonishing precision of its predictions, some of which[3] match the experiment with a relative error of less than 1 in 10^9. In practice, researchers investigating physics and chemistry at the nanoscale are not interested in such numerical accuracy, but focus on modelling the properties and behaviour of their systems of study first qualitatively, and then perhaps to a few digits of precision. Quantum chemistry provides a unifying description of all chemical and electronic structure of atoms and molecules - which explains their chemical reactivity- and their physical and chemical properties, including spectroscopy. However, the full quantum mechanical description is, except for the smallest cases, too impractical and often out of the reach of any present supercomputer, so nearly all methodology of quantum chemistry has been developed around the use of reasonable physical or chemical approximations which tradeoff some accuracy for speed. Besides very simplified models constructed to qualitatively predict the outcome of some reactions or explain some experimental data from some orbitals or bonds, the computation of the full wavefunction of a molecule was a practical impossibility until transistor-based computers started to spread in research labs in the mid 1950s, and soon started to attract the attention of many chemists[4].

The first successful methods used in computational chemistry were developed within the following decade, extending to the entire molecule the orbital description, and using empirical data to estimate the value of the matrix elements. As computers became faster and memory increased, the number of atoms that could be treated with semiempirical methods also increased, and more expensive and accurate models based on fewer simplifications from first principles (ab-initio) became also more popular. Nowadays there are dozens of methods available, capable of simulating systems up to several thousands of atoms to various degrees of approximation and computational cost. The value of such simulations goes well beyond their use as simple tools of prediction, as the joint use of fast algorithms and 3D computer visualization has transformed the value of the output, from what years ago was just a stream of numbers, into today's window to the behaviour of matter and energy at the nanoscale, observable otherwise only indirectly and with highly sofisticated laboratory equipment. This graphic-oriented human-machine communication is invaluable for developing better intuition of the physics governing a regime where the macroscopic intuition of everyday experience is of little use.

Computational chemistry continues to be a field in constant expansion due to the demand of new theoretical methods to study a growing list of problems in chemistry, biology and physics, and the need to extract good performance off hardware architectures constantly changing and improving in computational power. Like many other scientific disciplines, computational chemistry has greatly benefited from the exponential growth in semiconductor integration of the last decades, to a point where SCF calculations of thousands of atoms and molecular mechanics simulations with millions of atoms are not unheard of.

Scientific software has been traditionally written in the FORTRAN programming language[5]. FORTRAN was originally designed by IBM during the 50s as an alternative to the assembly languages of the time. The objective was to abstract any low-level details of the hardware and OS and provide just the functionality required to write numerical software that could be easily understood (by humans). Many programmers of the time were reluctant to use a high-level language due to concerns about performance loss, which prompted IBM to develop the first optimizing compilers for FORTRAN. The generally good performance of the compiled programs popularized FORTRAN amongst scientists, until the language was finally standardized in 1966. During decades, supercomputer manufacturers provided a FORTRAN compiler for every new machine, which would aggressively optimize the code for the specifics of the architecture. Since architectures were in general incompatible amongst themselves there were no good reasons to attempt any low-level optimization of the code by hand. This model began to change in the 90s, when the relative low cost and high volume of production of PC hardware gradually changed supercomputer architectures. The standarization of software libraries, programming languages and other software tools, induced a transition in the HPC community from the expensive monolithic supercomputers of the 80s based on custom hardware, to massively parallel systems composed of networked high-end PC hardware in the following decade. FORTRAN compilers for different OSs allowed porting old scientific codes to the x86. However C/C++ were already the languages of choice to develop PC applications and offered a greater flexibility than FORTRAN, so many developers switched to C/C++ for their new projects or mixed C/C++ with Old Fortran codes. After a few years of competition with the PVM model[6], the MPI standard[7] became the de-facto programming model for multi-processor and cluster scientific software. During the decade, x86 processors doubled in frequency and halved in size every 18 months or so, increasing the performance of any program written for x86 without changing a single line. Any increases in hardware complexity, like the increasingly more complex instruction pipeline were handled transparently to the programmer.

The turn of the millenium brought two important additions: first with the addition of vector instructions and later with the jump to 64 bits. The original MMX vector instructions were a simple way to operate on two integers with a simple instruction, but many more instructions would be added later on which converted the floating point registers into vectore registers not unlike those in supercomputers two decades earlier. This change would eventually mean that either compilers needed to become smarter at finding potential vectorization spots in non-vector code or numerical codes would need to be partially rewritten to take advantage of a two-, four- or eight-fold speedup in floating point operations. The jump to 64 bits doubled the reggister widths of the processor, added new registers and enabled addressing more than 4Gb of memory.

The ”race of the megahertz” finished abruptly around the year 2004, when it became obvious that processor frequencies were about to reach power constraints[8]. The power consumed by a processor increases linearly with the frequency at which it operates. This energy is converted to heat, which must be dissipated from the die quickly enough before it damages its components. The thermal conductivity of the materials and the temperature gradients in a computer impose practical limits to the magnitude of heat flux that can be extracted. The cost of the energy itself was another factor that required attention, as processors had become the most energy-hungry component of the hardware, and laptops and clusters demanded higher efficiencies. The move from single core to multicore processors kept increasing the total theoretical performance after frequencies peaked at about 4Ghz, but the actual increase in performance for many applications was dubious. Many programs, even today, do not support parallelism, so no amount of extra cores will accelerate their execution. Parallel programming also requires a much more careful analysis of algorithms and data structures, because Amdhal’s law[9] predicts that any sequential or poorly parallelized section of the software can become a major bottleneck of the program. Fortunately, the scientific software built using MPI was ready to use any extra cores without modification, at the cost of sacrificing the advantages of shared memory.

The increasing gap between CPU performance and memory bandwidth, which had been previously addressed by increasing the cache memory and eventually adding a second cache (L2), became again a problem considering all cores had to share the same bus. There are different cache hierarchy designs implemented in different processors, but most use a shared L3 for all cores and per-core L1 and L2.

With the end of *Dennard scaling*[10], by which processors had been exponentially improving in performance per watt without design modifications, finding new approaches to increase efficiency soon became one of the main concerns of microprocessor design. This eventually forced redesigns in CPU microarchitectures, adding considerable amounts of extra logic in the form of register renaming, out-of-order execution, more sofisticated jump and branch predictors, multiple execution ports, etc.

All these technologies have been steadily improving, and are part of virtally every computer processor being sold as of 2014. While still improving in performance, the situation is far from when it was possible to get a ”free” performance improvement from a code by buying a new computer. Processors sold as much a seven years ago still offer nearly the same single-core scalar performance as a similarly priced processor today. How to make the newest hardware run anywhere near peak performance has to be necessarily integrated in the design of the program.

One of the obvious inconveniences of CPU designs when it comes to numerical performance is the small area reserved to FLOP instructions compared to the area for control. GPUs, on the other hand, have these ratios almost reversed; they employ most of the surface in floating point ALUs, with some small control units alternating. Although this use is not new[11], GPUs have become increasingly popular in HPC thanks to the commercialization of products with full double precision support, registered memory, and peak FLOP performances up to 5 Tflop/s. GPU-accelerated numerical libraries are free to download and use[12]. On the downside, programming numerical algorithms for GPUs can prove to be quite challenging, although it is becoming easier as users demand missing hardware and software functionalities and manufacturers successively implement them in every next iteration.

Accelerators such as Intel Xeon Phi[13] are a response to the rise of GPUs in HPC. Instead of introducing a new programming model that new users might be very unfamiliar with, Intel has integrated a manycore card with very minimalistic in-order x86 cores, each containing a vector ALU wider than the SIMD of regular x86 cores, and a control unit that can manage the concurrent execution of up to 4 threads. Its main advantage over GPUs is that code written for CPU can be recompiled and run on the Xeon Phi without modification, although obtaining good performance requires tuning.

The near future of HPC seems to be tightly constrained by the laws of physics. Despite some improvement, the problems of energy efficiency, heat dissipation and memory bandwidth are sooner or later unavoidable bottlenecks to current semiconductor technology. The trend aims clearly towards increasing multilevel parallelism and distributed computing, which benefit inherently scalable and parallel problems, but fail to improve their sequential bottlenecks. The imminent introduction of 3D stacked memory will surely give some fresh air and boost many memory-intensive applications, but this is a one-time trick.

Despite the active interest in finding more efficient technologies, there is still no alternative material or process even close in its maturity to be a substitute candidate to CMOS in the next decade. The current trend in miniaturization is expected to reach the 5nm node as soon as 2020[14], coinciding with the expected arrival of the first exascale supercomputers[15]. 5nm is about the width of ten silicon cells, possibly the absolute smallest functional transistor that can be made with silicon without deviating too much from its ideal behaviour. 3nm and smaller may still be possible to print, but current leakage at these scales consumes nearly all power and degrades the energy efficiency sharply. It is also not clear how reliable would these circuits be, and how much error correction would be necessary to make them viable. Overall, CPUs and accelerators of the time could well peak at 10 to 30 times the performance and density of current hardware based on the 22 nm node, and perhaps 10 times the current memory bandwidth, mostly due to the migration from the traditional bus designs to high throughtput stacked memory. The improvement in performance per watt is also expected to rise by a similar factor due to the migration to Silicon-on-Insulator manufacturing and Near-Threshold Voltage computing.

It is of course expected that manufacturers will keep adding incremental updates to the microarchitecures afterwards, but this will only benefit control and logic intensive software. Numerical performance relies necessarily on the number of transistors and die surface dedicated to floating point multiplier/FMA units, which are already heavily optimized logic blocks. Any substantial increase in performance will have to come from radically new microprocessor designs, materials or technology.

If something has been proven constant during the 70 years of scientific computing is that no amount of computational resources is ever enough. Even with today's petaflop supercomputers there is a myriad of simulation problems of an even bigger magnitude waiting to be solved. However, if hardware performances stall, it is not likely that this demand will be met by multiplying the number of machines in the HPC clusters. Tianhe-2, the world's fastest supercomputer at the moment peaks at 33 Pflop/s and uses about 24 MW, with the refrigeration system alone using 7 MW[16]. Tianhe-2 delivers about 1.9 Gflop/J Other supercomputers are much less energy efficient, to the point of having an annual energy bill exceeding the price of the hardware. The road to exaflop according to DARPA's program PERFECT[17] requires an efficiency of at least 75 Gflop/J, about 40 times the efficiency of Tianhe-2 and 17 times that of the top efficient cluster in the Green 500 list as of August 2014[18]. Even if all manufacturing and architectural challenges necessary for exascale computers are met by 2020, computing beyond exascale will remain a logistic impossibility unless the root problems of energy efficiency and other limitations of the silicon technology are addressed in one way or another.

Since user applications running on current low-end hardware rarely experience bottlenecks anymore, computer sales have declined in the last few years notably, and portable consumer electronics are slowly taking their place. Without a large consumer base fueling a demand for new hardware, it looks like the smaller HPC and server markets need to look elsewhere to be able to supply an always increasing demand. This performance wall poses the biggest challenge to the future of HPC. Recent history has proven in many occasions that speculating about the viability and speed of diffusion of future technologies can be an excercise in futility. However, it seems reasonable that some technologies existing today such as ASICs (application-specific integrated circuits) and FPGAs (Field-Programmable Gate Arrays) will tenmporarily bridge the gap. Special-purpose hardware usually delivers much better performance per watt and speedups between one and three orders of magnitude over equivalent algorithms running on regular processors. The GRAPE (gravity pipe) supercomputer[19] uses heavily pipelined ASICs to solve N-body systems. The latest GRAPE-8 design powering the cluster is synthesized on a stripped-down FPGA board running at 250Mhz, and yet it provides 480 Gflop/s per chip at only 26 W consumption[20]. A similar design ported to ASIC could possibly achieve 5 Tflop/s. The Anton supercomputer[21] is a 512-node machine with special-purpose ASICs designed to solve MD simulations quickly, and at the time of its debut it beat the fastest MD software running on a 256-node opteron cluster by a factor of 300. The best FFT software libraries peak at about 15 Gflop/s in multicore processors[22] and 150 Gflop/s on GPUs[23], while FPGA implementations achieve up to 20 Tflop/s[24]. Even in the field of quantum chemistry, there has been some interest in developing hardware for accelerating the evaluation of electron repulsion integrals[25][26]. The obvious inconveniency to these approaches is the increased complexity of making an integral analysis of the application, study different algorithmic solutions and jointly engineer and program the hardware and software. ASICs in particular also have a high barrier to entry largely due to the cost of development and testing of the photolitographic masks, which can only be amortized by the producing large enough quantities of the hardware.

Regardless of the upcoming hardware solutions that will appear in the next years, the central paradigm of HPC will remain to make the best use of available resources: computational, energetic and economic. The key to this is to use the best algorithms to solve the problem, optimized for the hardware at hand. This idea is always stressed in every introductory text to HPC, but is very rarely followed by actual examples, so here is one: during the golden age of Moore's Law, from 1988 to 2003, the performance of microprocessors multiplied by about three orders of magnitude. However, during the same period the algorithms for solving linear programming problems improved by 4 to 5 orders of magnitude[27]. Similarly, when SCF calculations became more widespread, the asymptotic complexity of HF and DFT calculations was improved from the canonical $\mathcal{O}(N^4)$ to $\mathcal{O}(N^2)$[28] or even $\mathcal{O}(N)$ in some cases, with the focus being now shifted towards reducing the linear prefactor. Linear algebra problems, which seemed to be a completely solved field, are now some of the most studied problems in HPC and numerical analysis, with most of the focus of the last decade being put in finding parallel scalable algorithms[29] and developing efficient methods for sparse matrices. Examples like these are common, and illustrate how progress in numerical simulation methods comes from many sources, and how the evolution of algorithms and hardware can influence one another.

Algorithms, in general, have hoever fallen short in the parallelism facet. Despite the existence of clusters with thousand to millions of cores, few (if any) codes are able to utilize the totality of the hardware of a modern supercomputer without incurring in any major bottleneck in its design that degrades its performance well below what is reasonable. This scale of computation is of course neither necessary nor recommended for most simulations, but will become an increasingly noticeable issue as supercomputers become even more massively parallel. The scalability problems of many parallel numerical codes have pushed already the design of new algorithms. These designs and their implementation can be particularly difficult considering multicore, manycore and massively parallel computing are relatively new and many programmers have not yet developed the skills to think concurrently. Beside the lack of standard hardware and software mechanisms to support concurrency, the issue of resiliance (the probability of any one hardware component failing during the computation is dramatically increased in massive parallel systems) needs to be eventually addressed in the design.

Chapter 2

Quantum Chemistry

2.1 Basic concepts

The main goal of quantum chemistry is the determination of the properties and reactivity of molecules, for which the wavefunction of the chemical system is required. The time-independent Schrödinger equation[30] provides the starting point for the mathematical formulation of non-relativistic quantum chemistry:

$$H|\Psi_i\rangle = E_i|\Psi_i\rangle \tag{2.1}$$

Where H is the quantum Hamiltonian operator of the system, E_i is the energy of the i-th state and ψ is its wavefunction, which is a function of the positions (or equivalently, the momentums) and spins of all the particles in the system, and the time.

$$|\Psi_i\rangle = \Psi_i(\mathbf{r}_1, s_1, \ldots, \mathbf{r}_N, s_N; t) \tag{2.2}$$

The full solution of the time-independent Schrödinger equation is the energy spectrum of the system, and all the corresponding states. In the time-independent formulation, the time only appears to shift the phase of the function $\Psi(\mathbf{r}; t) = \Psi(\mathbf{r}; 0)e^{-2\pi iEt/h}$ and is irrelevant for the discussion. The states form an orthogonal basis and are assumed normalized, which in Dirac's bracket[31] notation is expressed as:

$$\langle\Psi_i|\Psi_j\rangle = \sum_{s_1,\ldots,s_N}\int_{V_1} d\mathbf{r}_1 \ldots \int_{V_N} d\mathbf{r}_N \Psi_i^*(\mathbf{r}_1, s_1, \ldots, \mathbf{r}_N, s_N)\Psi_j(\mathbf{r}_1, s_1, \ldots, \mathbf{r}_N, s_N) \quad (2.3)$$

$$\langle\Psi_i|\Psi_j\rangle = \langle\Psi_j|\Psi_i\rangle = \delta_{ij} \quad (2.4)$$

The non-relativistic evolution of a the system is determined by the time-dependent Schrödinger equation:

$$i\hbar\frac{\partial}{\partial t}|\Psi\rangle = H|\Psi\rangle \quad (2.5)$$

The Hamiltonian is defined in correspondence to the classical Hamiltonian by applying canonical quantization and substituting the classical momentum with the quantum momentum operator.

$$\begin{aligned}
H &= T_N + V_{NN} + T_e + V_{eN} + V_{ee} && (2.6)\\
T_N &= \sum_n -\frac{1}{2M_n}\nabla^2_{R_n} && (2.7)\\
V_{NN} &= \sum_{n<m}\frac{Z_n Z_m e^2}{4\pi\epsilon_0|R_n - R_m|} && (2.8)\\
T_e &= \sum_i -\frac{1}{2m_e}\nabla^2_{r_i} && (2.9)\\
V_{eN} &= \sum_{i,n}\frac{-Z_n e^2}{4\pi\epsilon_0|R_n - r_i|} && (2.10)\\
V_{ee} &= \sum_{i<j}\frac{e^2}{4\pi\epsilon_0|r_i - r_j|} && (2.11)
\end{aligned}$$

The full Schrödinger equation finds actually little use in quantum chemistry. The molecular structure of the system is often known in advance from experimental methods or lower levels of theory, and the sought information is mostly encoded in the electronic degrees of freedom. The Born-Oppenheimer approximation[32] splits the total system wavefunction as a product of nuclear and electronic wavefunctions or, equivalently, splits the Hamiltonian into the direct sum of nuclear and electronic Hamiltonians.

$$\begin{aligned}
H_N &= T_N + V_{NN} + E_e(\vec{r}_n) && (2.12)\\
H_e &= T_e + V_{ee} + V_{eN}(\vec{r}_n) && (2.13)
\end{aligned}$$

This approximation is justified whenever the time scale of the nuclear motion is much larger than the time scale of the electron motion, which is given to first order by their mass ratio. Because nuclei are thousand of times heavier than electrons, the system can be thought of in the nuclear scale as a cloud of electrons responding instantly to any nuclear movement, or in the electronic time scale as an electronic wavefunction determined by the electrostatic potential generated by a set of nuclei fixed in space. There are situations in which this approximation breaks down, most notably around conical intersections[33] in excited states. The treatment of such cases requires the explicit coupling of the relevant nuclear degrees of freedom to the electronic wavefunction, an approach known as *vibronic coupling*. Otherwise, the nuclear degrees of motion are usually treated on their own through a variety of methods and approximations.

The convention for the electronic Hamiltonian is the use of atomic units, with $e = m_e = \hbar = 1/(4\pi\epsilon_0) = 1$. It is often expressed as the sum of 1- and 2-particle components:

$$H = \sum_i h_i + \sum_{i>j} g_{ij} \tag{2.14}$$

$$h_i = -\frac{1}{2}\nabla^2_{r_i} - \sum_n \frac{Z_n}{|r_i - R_n|} \tag{2.15}$$

$$g_{ij} = \frac{1}{r_{ij}} = \frac{1}{|r_i - r_j|} \tag{2.16}$$

With indices i and j representing the coordinates of each electron. The 1-particle components of the Schrödinger equation have their relativistic counterpart in the Dirac 1-particle Hamiltonian[34]

$$h_i^D = -ic\vec{\alpha}_i\vec{\nabla}_{r_i} + c^2(\beta_i - \mathbf{I}_4) - \sum_n \frac{Z_n}{|r_i - R_n|} \tag{2.17}$$

correct to all orders of $\alpha = 1/c$. The relativistic Hamiltonian is no longer a scalar operator; $\vec{\alpha}_i$ and β_i are 4×4 Dirac matrices, and its solutions are 4-vector wavefunctions. The special importance of this lies in the physical meaning of its four solutions, corresponding to the four combinations of the two possible spin states and its two possible energy states, one for the electron and the other for the positron. The most accurate many-body relativistic equations are derived directly within the QED (quantum electrodynammics) framework, but given the scale of the energies under consideration in chemistry, the most significant relativistic effects for all intents and purposes can be incorporated with the few first terms of the expansion in α, and treated usually by perturbation theory. The most used relativistic corrections are the Gaunt and the Breit interactions

$$g_{ij}^{CG} = \frac{1}{r_{ij}} - \frac{\vec{\alpha}_i \vec{\alpha}_j}{r_{ij}} \tag{2.18}$$

$$g_{ij}^{CB} = \frac{1}{r_{ij}} - \frac{\vec{\alpha}_i \vec{\alpha}_j}{2r_{ij}} + \frac{(\vec{\alpha}_i \vec{r}_{ij})(\vec{\alpha}_j \vec{r}_{ij})}{2r_{ij}^3} \tag{2.19}$$

which are correct to $\mathcal{O}(\alpha^0)$ and $\mathcal{O}(\alpha^2)$, respectively.

The time-independent Hamiltonian is a differential equation which admits very few known analytic solutions. In practice, however, approximate solutions can be obtained by selecting a basis on which the operators are projected, effectively transforming the differential equation formulation into an algebra problem. Matrices were, in fact used in the original formulation of quantum theory by Heisenberg, later shown to be equivalent to Schrödinger's ondulatory formulation. This approach can be extended into what is known as *second-quantization* formulation, where the Hamiltonian and other operators are represented as linear combinations of the elements of the *CCR* and *CAR* algebras, also known as creation and annihilation operators. These algebras correspond to the behaviour of fermions and bosons.

2.2 Electronic structure methods

The electronic wavefunction of a system of N electrons is a function of the $3N$ coordinates of space and its N spin coordinates. Additionally, it must obey Pauli's exclusion principle, which can be expressed as the function's antisymmetry with respect to the exchange of the coordinates of any two electrons:

$$\Psi(\ldots, \mathbf{r}_i, s_i, \ldots, \mathbf{r}_j, s_j, \ldots) = -\Psi(\ldots, \mathbf{r}_j, s_j, \ldots, \mathbf{r}_i, s_i, \ldots) \tag{2.20}$$

and must be an eigenstate of the total spin operator $\hat{S}^2$. In practice, one builds an N-particle basis satisfying the antisymmetry constraint, guaranteeing that any linear combination of the basis will also trivially satisfy the condition. This translates the problem into the linear algebra domain, where it can be solved efficiently. The Hamiltonian is an hermitian operator, which implies that the energy of the ground state (its extremal eigenvalue) is bound by the variational principle:

$$\forall\Psi : E_0 \leq E[\Psi] = \frac{\langle\Psi|\hat{H}|\Psi\rangle}{\langle\Psi|\Psi\rangle} \tag{2.21}$$

coinciding for the exact ground-state wavefunction $E_0 = E[\Psi_0]$. Attempting the direct minimization of the energy functional over general parametrized antisymmetric functions is not the preferred solution in routine calculations. The overwhelmingly preferred approach is to construct solutions as linear combinations of a particular class of antisymmetric functions named Slater determinants. Slater determinants are N-particle functions constructed by antisymmetrizing a product of 1-particle functions:

$$\Psi(\mathbf{r}_1, \mathbf{r}_2 \ldots \mathbf{r}_N) = (N!)^{-1/2} \begin{vmatrix} \phi_1(\mathbf{r}_1) & \phi_2(\mathbf{r}_1) & \ldots & \phi_N(\mathbf{r}_1) \\ \phi_1(\mathbf{r}_2) & \phi_2(\mathbf{r}_2) & \ldots & \phi_N(\mathbf{r}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(\mathbf{r}_N) & \phi_2(\mathbf{r}_N) & \ldots & \phi_N(\mathbf{r}_N) \end{vmatrix} = |\phi_1\phi_2 \ldots \phi_N|$$

where the 1-particle functions are assumed orthonormal. This *ansatz* simplifies considerably the manipulation, because any pair of Slater determinants constructed from a common orthonormal set of functions are orthogonal if and only if it contains at least one different function.

One way to resolve the Hamiltonian spectrum is to project it on a complete basis of $\binom{B}{N}$ Slater determinants, constructed from B orthonormal single-particle functions. This approach is also known as full-CI, and is performed with an iterative eigenvalue algorithm (a variant of the power method), which gives a few selected lowest pairs eigenvalue/eigenvector. This approach is however rarely used in practice given the prohibitive computational cost, which is roughly exponential with the number of electrons in the system, having a scope that is limited to very small systems.

For practical calculations, there are many methods based on different levels of approximation to the wavefunction or Hamiltonian, with more reasonable computational costs than full-CI. The most common electronic structure methods used nowadays are HF/DFT ($\mathcal{O}(N^4)$), Möller-Plesset pertubation theory ($\mathcal{O}(N^5)$ for MP2) and Coupled Cluster ($\mathcal{O}(N^7)$ for CCSD(T)), all exhibiting polynomial asymptotic behaviour which depending on the implementation can be made reduced to more reasonable powers and made even linear-scaling. The HF/DFT approaches are outlined next.

2.2.1 SCF methods

The Self-Consistent Field (SCF) methods of quantum chemistry are the quantum equivalent of Mean-Field Theory in statistical mechanics[35], which is sometimes also refered to as ”self-consistent field theory”. SCF methods are the very heart of quantum chemistry, and have been extensively studied in the literature from every angle: range of applicability, convergence of the procedures, functionals (for DFT), methods for accelerating its convergence, low asymptotic scaling algorithms for large molecules, etc.

Hartree-Fock

The Hartree-Fock method[36] is perhaps the simplest ab-initio approximation to the full electronic Hamiltonian, but provides meaningful results to a variety of chemical problems, and its solution is often a good reference from which higher-order solutions are build. It starts with the assumption that the electronic degrees of freedom are uncorrelated, and that every electron ”feels” the potential of the averaged distribution of the rest of the electrons. This is the equivalent of substituting the electron-electron interaction potential of the electronic Hamiltonian with a one-particle potential, which makes the N-electron Hamiltonian fully separable into N x 1-particle Hamiltonians, which can be solved simultaneously by diagonalizing the Fock operator. The solution is used to contruct a new averaged potential from which a new Fock operator is built and solved, and the procedure is carried on iteratively until convergence. For a closed-shell system, the Fock operator is:

$$F = h + \sum_{j}^{N/2} (2J_j - K_j) \tag{2.22}$$

The Coulomb operator J_j is the contribution to the repulsion generated by the orbital j of the previous iteration. The exchange operator K_j has no classical counterpart, and arises from the antisymmetry of the total wavefunction.

The common HF procedure consists in choosing a basis set for the system and evaluating $\langle \phi_n | F | \phi_m \rangle$ by computing the necessary 1-electron integrals of the core Hamiltonian h and the Coulomb and Exchange operators. The projection of the operators J and K in the basis requires two-electron integrals, which are numerous and time consuming. In the early days they were computed at the beginning of the calculation, stored in disk and retrieved as necessary, but after decades of increasing speed gap between the electronics and the electromechanical components of computers, it has become more efficient to compute them when necessary, and use more thoughtful methods in general. Today there are many different approaches aimed at substantially speeding up, and even in some cases at even circumventing the whole procedure entirely. Given that the algorithm is iterative, the first steps can use much faster and cruder approximations, increasing the numerical precision as the procedure converges.

For the most part, Hartree-Fock calculations involve non-orthogonal atom-centered bases. The molecular orbitals can be solved with the Roothaan-Hall equation[37], which is a generalized eigenvalue problem:

$$F^{(n)} C_i^{(n)} = S C_i^{(n)} E_i \tag{2.23}$$

with F the Fock matrix, S the overlap matrix, C_i the orbital solutions and E_i their respective energies. The N orbitals of lowest energy are used to construct the 1-electron density matrix:

$$\rho_{ij}^{(n)} = \sum_k^{N/2} C_{ik}^{(n)} C_{kj}^{(n)} \tag{2.24}$$

which in turn is used to build the Coulomb and Exchange contributions to the Fock matrix of the next cycle:

$$F_{ij}^{(n+1)} = h_{ij} + 2J_{ij}(\rho^{(n)}) - K_{ij}(\rho^{(n)}) \tag{2.25}$$

The one-particle reduced density matrix (or some guess) is used to compute the next Fock matrix. The Fock matrix is diagonalized in the system basis and a new density matrix is constructed from the orbitals corresponding to the lowest energy eigenvalues. The density matrix can also be obtained directly without diagonalization through a number of methods based on density matrix purification[38]. The density matrix obtained is normally not fed back in the loop right away. Convergence accelerators such as DIIS, ODA or EIIS keep track of the results of previous steps and can extrapolate a better guess for the next DM, which reduces the number of SCF cycles. They can also fail to converge in some cases.

The final molecular orbitals can be used to construct a Slater determinant that can be proven to minimize the expected value of the Hamiltonian within the subspace of single determinants generated from the given basis set. In other words, the HF method provides the best approximation to the true wavefunction expressable with only one Slater determinant. This is the reason why the Hartree-Fock method is used as a reference ground state on top of which better approximations are built. The HF solution is however limited in its scope. Systems with only dynamic correlation, like the ground state of most organic and many inorganic materials, can be reasonably approximated by a Hartree-Fock ansatz. The solution can be further refined with perturbational and/or exdended variational approaches such as Möller-Plesset[39], Configuration Interaction[36] and Coupled-Cluster[40]. But as expected, the method predicts solutions very departed from the variational minimum in any system containing highly correlated electrons, also known as static correlation. Breaking chemical bonds, transition metal oxides or metals with incompletely filled $d-$ and $f-$ electron shells are typical examples of static correlation where the HF method doesn't provide a good approximation. Even the usual correlated approaches for these systems tend to fail, due to their limited ability to treat correlation other than local (dynamic). The proper description of these systems requires a reference ansatz which explicitely includes some level of static correlation, like valence-bond theory, multiconfiguration SCF[36] or DMRG[41].

Some variants of HF can treat systems with open shells. Unrestricted Hartree-Fock (UHF)[42] simply decouples alpha and beta electrons and generates different spinorbitals and occupation numbers for the alpha and beta electrons. The problem with this approach is that its solutions are not in general proper solutions of the total spin operator S^2. The Restricted Open-shell HF method (ROHF)[43], is slightly more complicated in its formulation, but solves the spin contamination problem,

Density Functional Theory

Density Functional Theory is a method based on an entirely different approach than Hartree-Fock. The Hohenberg–Kohn theorems[44] state that the wavefunction of a non-degenerate system is uniquely determined from its electronic density $n(\vec{r})$, and that the energy of the system as a functional of the density is minimized only by the correct ground state density. This parallels the variational principle for wavefunctions, but on a significantly reduced number of degrees of freedom, making it a very attractive computational approach. The energy functional can be split as a sum of kinetic functional, external potential (arising from the nuclei and any other contribution), Coulomb repulsion, and exchange-correlation functional. The functional contributions from the kinetic $T[n]$ and 2-electron terms $U[n]$ should always have the same form, since they depend only on the electron density, and are thus called *universal*. The electron density -and therefore all properties of the ground state- is determined only by the external potential $V(\vec{r})$.

$$E[n] = T[n] + U[n] + \int V(\vec{r})n(\vec{r})d^3r \tag{2.26}$$

The exact functional form of the kinetic energy and exchange-correlation terms is however not known, and there are compelling mathematical reasons to think that it might not be possible to express them in a useful way. The exact functional has to be able in principle to describe correctly not only loosely correlated electrons, but also states with high amounts of static correlation, which amounts to solving the problem of N-representability[45]. In other words, the cost of evaluating the exact functional would be equivalent to solving the system by traditional means. There are however many approximations and parametrizations for the functionals that can work good enough in a range of chemical systems, ranging from the very simple -like the Thomas-Fermi model[46]- to the very complex, each of them with its strengths and weaknesses.

The Kohn-Sham formulation of DFT[47] results in a SCF procedure very similar to HF. The fundamental difference is the introduction of an exchange-correlation functional, which should correct the deficiencies in the kinetic functional used, the lack of proper exchange for same-spin electrons and the rest of the correlation. The Kohn-Sham SCF equation is analogous to the Roothaan Hall equation:

$$K^{(m)}C_i^{(m)} = SC_i^{(m)}E_i \tag{2.27}$$

projected in some basis $\chi_k(\vec{r})$, on which the resulting orbitals are expressed as:

$$\phi_i^{(m)}(\vec{r}) = \sum_k C_{ik}^{(m)}\chi_k(\vec{r}) \tag{2.28}$$

from which the density is computed:

$$n^{(m)}(\vec{r}) = \sum_i^N |\phi_i^{(m)}(\vec{r})|^2 \tag{2.29}$$

which is used to generate the next Kohn-Sham operator:

$$\hat{K}^{(m+1)} = \hat{T} + \hat{V}_{ex}(\vec{r}) + \hat{J}[n^{(m)}] + \hat{V}_{XC}[n^{(m)}] \qquad (2.30)$$

Hybrid functionals use the HF exchange matrix as part of their formulation. The rest of the terms of the exchange-correlation functionals are local functionals of the density and its derivatives (an extra assumption not inferred from the method), and are not that computationally intensive in comparison. Regarless of its limitations, DFT is one of the preferred methods in quantum chemistry because it is able to introduce the effects of electron correlation at a negligible cost over HF.

2.2.2 QM/MM

The use of ab-initio QM to calculate the properties of systems of the scale of biomolecules has been impossible until recently, and continues to be too impractical even with linear scaling state-of-the-art QM algorithms and large computer clusters. Fortunately, in most cases the interesting chemistry and phyisics of the system is localized to relatively few atoms. For instance, in the study of enzymatic catalysis, the region of interest includes the atoms of the active site and the substrate, where the chemical reaction occurs. In reactions in solution, the accurate description of the solvent can be considered a second-oder effect of smaller impact than the correct description of the reactants. QM/MM is a very useful approach in such cases[48], combining the strengths of QM methods and MM methods. QM/MM methods partition the system into a chemically-active subsystem, treated with the QM method of choice, and the rest of the system, treated with MM methods.

QM/MM allow the study not only of chemical reactions within the much larger system, but also photochemistry, spetroscopy, and other properties depending on the electronic wavefunction. The main problem of QM/MM is the treatment of the boundary and the interaction between the subsystems, which is not straightforward considering that the transition from a region described by classical phenomenological potentials due to the combined effects of nuclei and electrons to a region where electrons are described in detail is not smooth. Different embeddings can be used to more or less accurately enforce the consistency between systems. To solve the issue, it is also necessary to calibrate the QM/MM interaction potentials to reproduce the interaction between the subsystems. A more difficult problem appears when the QM region is covalently bonded to the MM system, because chemical bonds appear naturally in the QM solution, but in MM they need to be explicitly defined. One usual solution in these cases is to cut the systems at some sigma bond and treat the boundary of the QM region by either adding frozen orbitals at the bound MM atoms or terminating the QM system with a virtual ”dummy” (link) atom or group that doesn’t significantly change the shape of the wavefunction in the QM region.

2.2.3 QM/CMM

The QM/CMM method[49] is a new multiscale method developed in our laboratory, that expands the reach of the QM/MM methodology to include the interaction with metallic nanoparticles and surfaces where plasmonic effects are not dominant. The metallic part is modelled with the capacitance-polarizability method of Jensen et al.[50], in which the individual atoms are defined as having capacitance and polarizability, and the interacting system is solved by computing the partial charges and dipoles of every atom self-consistently. This level of description of the metallic system lies somewhere between the QM models and the solution of the macroscopic Maxwell equations, and is capable of accurately reproducing the experimental polarizability of noble metal nanostructures. The rest of the system is modelled using the traditional QM/MM approach, using DFT for the QM region, and a polarizable force field for the MM part. One obvious restriction of this method is that chemical bonding between the metallic part and the other subsystems is not allowed. The method is currently being tested and developed, but initial application to the prediction of the optical properties of molecules physisorbed on noble metal nanoparticles and surfaces are promising.

2.2.4 Effective core potentials

One of the fundamental observations in chemistry are the similar physical and chemical properties of elements of the same group, which is the direct consequence of the partial shielding of the nuclear charge from the filled inner electron shells leaving somewhat similar effective potentials for the electrons in the outermost valence shells, which are responsible of nearly all of the chemistry of an element. Inner shells tend also to remain largely unaltered by the chemical environment of the atom, and in heavier atoms contribute with many electrons and basis functions to the system, significantly increasing the dimension of the problem and the required computational cost to solve it. Pseudopotentials combine the nuclear charge and the shielding effects of the core electrons into a single potential, eliminating the need of an explicit all-particle approach. Moreover, pseudopotentials can include the relativistic effects of the innermost electrons, and can considerably simplify the relativistic treatment of heavy elements or avoid it entirely. Pseudopotentials are modified from the true potential in such a way that they produce atomic (pseudo)orbitals smooth and nodeless close to the nucleus, and coinciding with the "true" orbitals obtained from the full system when the radius is larger than some distance.

2.2.5 Response theory

Many observable physical properties of a chemical system are due to its interaction with the environment, and can therefore not be computed directly from the ground state wavefunction or density. Some of these properties can be calculated from the time-dependent response functions of the system. In particular the time-dependent response functions due to an oscillating electric field are useful to predict the frequency-dependent polarizabilities and hyperpolarizabilities, as well as the energy of excited states, their transition moments and many of their properties. Response theory is even more relevant for DFT, since it cannot, by design, compute the density of states other than the ground state[51].

The response theory formalism adds a time-dependent pertubation to the system's Hamiltonian, coupled adiabatically. The calculation of properties to different levels is due to the expansion of the perturbation as linear superposition of frequencies plus terms quadratic, cubic, etc. to generate the linear, quadratic, cubic, etc. response functions.

2.3 Basis sets

The choice of an adequate basis set plays a key role in obtaining meaningful results from a QC calculation. In principle, any complete basis set could be used to compute a QC problem, but in practice only a few general types are commonly used. The choice of some basis set is usually due to its flexibility for the problem of interest, the cost of evaluating the necessary integrals, and other computational limiting factors such as available memory, performance of the necessary algebra subsystems, etc. The chosen basis set has to be able to describe the features of the system that are important for the properties or behaviour we are interested in studying. A poor choice of basis will result in meaningless data regardless of the level of theory or quality of the method, since no amount of correlation can fix the shape of the 1-particle orbitals.

Common basis sets in QC can be largely divided in what we will refer to as "systematic" and atom-centered basis sets. The first kind are usually insensitive to the chemical system of study and are uniquely defined by some resolution or cutoff parameters, and the shape of the box containing the system. Improving the resolution provides a systematic way of converging towards the complete basis set (hence the name). The second type is arguably the most popular in quantum chemistry, and is characterized by the extensive parametrization required to have a working basis set. Note however that this classification can be ambiguous in some cases, and that the word ”systematic” used here is not to be taken literally.

Many different types of basis sets have been used throughtout the years in quantum chemistry. Following is a summary of the most relevant.

2.3.1 Systematic basis sets

The main advantages of systematic basis sets are that molecular integrals can often be solved analytically, and can be computed in advance before even knowing any details about the system. Their main drawback is the high number of functions needed to represent the system with good accuracy. This is in part due to the difficulty of capturing the characteristic discontinuous shape of the orbitals at the nuclei positions, and the overall behaviour after that. The core orbitals also become more localized as the atomic charge increases, while other features such as long-range decay of valence orbitals are mostly unaffected due to screening, so more functions are needed to resolve all features. Increasing the number of functions rapidly increases the memory and the computational cost of the algebra. A usual approach is to eliminate the core shell electrons from the description and the $1/r$ Coulomb potential of the nuclei and substituting them for some type of pseudopotential. Pseudopotentials incorporate the effects of both the nucleus and core electrons of the atom, and their shape is modified close to the atom center so that the valence orbitals obtained from each pseudopotential are close to the true orbitals away from the center, but remain smooth and nodeless close to the atom core. This approach can reduce drastically the resolution of the basis set needed to describe the chemistry of the system.

Plane waves

Plane waves are popular basis sets due to their unique properties. Plane waves are the solution to the Schrödinger equation for the free-moving particle, and a set is uniquely defined by some boundary box and an energy cutoff,

$$\chi_{\mathbf{k}}(\mathbf{r}) = e^{-2\pi i \mathbf{k}\cdot\mathbf{r}} \tag{2.31}$$

As advantages, plane waves offer one of the most systematic ways to extend the accuracy of calculations, by just increasing the energy cutoff. They are orthonormal, which simplifies QC methods by removing orbital dependency of the overlap matrix. Since the basis is always the same for a given (set) region, forces can be computed by the direct application of the Hellman-Feynmann theorem[52]. On the negative side, they can be one of the most "wasteful" sets for many systems. This is because of the irregular spacial distribution of the atoms of many chemical systems, which leave regions of the bounding box practically empty of electron density. Another problem with plane waves is that they invariably lead to dense linear algebra, which despite the efficiency of modern algorithms, eventually does not scale well with the dimension of the problem.

Spacial grids

A spacial grid basis set is composed of a set of points in space, which may or may not be arranged in a periodic grid. A periodic grid basis is related to a plane wave basis by a Fourier transform,

$$\chi_{\mathbf{A}}(\mathbf{r}) = \delta(\mathbf{r} - \mathbf{A}) \tag{2.32}$$

The orbitals or densities are described by their numerical values at some given points in space. Integration and derivation are usually carried out by implicitly assuming an interpolating function spanning the point and its nearest neighbours or is done analytically in Fourier space. The resolution considerations mentioned above also apply to this case, so the grid should be dense enough to represent the most contracted orbitals/region with largest density fluctuation. This makes spacial grids an acceptable basis for representing the density in DFT, but a very wasteful representation of orbitals, unless orbital localization of some kind is enforced. Some spacial grids incorporate some dependence with the shape of the external potential, making them somewhat less "systematic".

Wavelets

Wavelets are functions used originally in data processing and multiresolution analysis, but have in the last years received increasing interest in quantum chemistry. A wavelet can be any function with good localization both in space and momentum, and a wavelet basis is easily contructed from the original wavelet by applying translation and scaling transformations. There are countless wavelet basis sets, but Gaussian Plane Waves (GPW) also refered to as *Morlet* wavelets or

Gabor wavelets in signal processing, have attracted most attention for use in quantum chemistry. GPWs share most if not all the benefits of plane wave basis, but their locality translates into sparser representations of the system,

$$\chi_{\mathbf{k}}(\mathbf{r}) = e^{-2\pi i \mathbf{k}\cdot\mathbf{r}} e^{-\alpha|\mathbf{r}|^2} \tag{2.33}$$

2.3.2 Atom-centered basis sets

Atom-centered basis sets stem from the LCAO approach to bond description in chemistry, where bonds are approximated by a linear combination of the orbitals of the parent atoms[53]. The use of atom-centered basis sets complicates the computation of matrix elements in many ways compared to systematic basis sets, but they simultaneously lower significantly the dimension of the algebra problem, usually showing fast numerical convergence with the addition of extra functions to the ”exact” numerical solution obtained from a complete basis set. They are constructed by optimizing beforehand a set of parameters for every shell and every element; the total basis set used to compute a system is generated by a union of elemental sets centered in every atomic nucleus.

The explicit dependence of the basis with the external potential generated by the nuclei implies a-priori assumptions about the form of the solution and therefore also about the properties of the system, making the approach less ”systematic” than the previously discussed examples. In polyatomic systems, shells centered in neighbouring atoms are not orthogonal, and it is often preferrable to generate a molecular basis set by orthogonalizing the atom-centered functions. The incompleteness of the basis is in this case a problem that leads to a variety of unphysical results if not addressed. Computation of inter- and intra- molecular interactions are subject to Basis Set Superposition Error[54][55], where the orbitals of every fragment are artificially over-stabilized by the basis functions of the other in a distance-dependent way, producing unreliable results for dissociacion energy curves unless the basis is close to complete or some correction is applied[56][57]. Another case is the computation of forces for ab-initio molecular dynamics or geometry optimization, where the Hellman-Feynmann theorem[52] does not hold if the basis set is not kept constant in the process. A failure to compensate for this effect introduces ”phantom” forces in the system, also known as Pulay forces[58]. Regarless of the inconvenients, atom-centered basis sets are by far the most popular in quantum chemistry; due to early adoption, there is an extensive amount of literature dedicated to their optimization and their limitations. Atom-centered basis sets also allow more compact representations of the molecular orbitals, and preserve the LCAO picture in the chemist's mind, which becomes useful in many analysis. Many efficient algorithms have been developed over the years, and their performace has also been subject of extensive analysis and optimization. The majority of basis sets employed in calculations of non-periodic systems are nowadays of the GTO type.

STOs

Slater-type Orbitals were the first kind of basis functions used in quantum chemistry. The analytic solution of the nonrelativistic hydrogenoid atom factors into a radial and an angular component,

$$\Psi_{nml}(\mathbf{r}) = R_{nl}(\mathbf{r}) Y_l^m(\theta, \phi) \tag{2.34}$$

The radial part $R_{nl}(r) = L_{nl}(r) e^{-\alpha |r|/n}$ is the product of a Laguerre polynomial responsible for the nodal behaviour and an exponential rapidly decaying with the separation to the center.

STOs can be used as a basis to construct N-particle wavefunctions of polyelectronic atoms, giving reasonable results for optimized parameters. In STOs the Laguerre polynomials are dropped in favour of a simpler r^n factor, and the exponents of each function are optimized to give the variational minimum. The rationale behind changing the exponents comes from the idea of separating the total wavefunction as a direct product of hydrogenoid-like wavefuctions, where the densities of the other electrons partially "shield" every orbital from the nuclear charge by a constant factor everywhere,

$$\chi_{nml}(\mathbf{r}) = |\mathbf{r}|^n Y_l^m(\theta, \phi) e^{-\alpha |\mathbf{r}|} \tag{2.35}$$

STOs can also be used for calculations in molecules. This approach has been traditionally limited to very few atoms, due to the difficulty of calculating multicenter integrals over STOs. These integrals do not have in general closed analytic solutions and require numerical quadrature methods for their evaluation, so even the best available integral algorithms for STOs can be orders of magnitude less efficient in polyatomic computations compared to other basis sets. The advantages of STOs, are the proper behaviour at the cusp $r = 0$, which guarantees that the divergence in the kinetic energy is exactly cancelled by the divergence of the potential. They also show the expected exponential decay for long distances, which is desirable to compute intermolecular interactions like van-der-Waals.

GTOs

Boys pointed out that the difficulties of evaluating integrals over STOs in molecules could be bypassed with the use of Gaussians. Gaussian functions have many remarkable properties which allow much faster algorithms for integral evaluation, which will be discussed in more detail in the next chapter. Gaussians are localized both in real and Fourier space, they are infinitely smooth and many integrals admit an analytic solution. However, they differ significantly from STOs and do not satisfy the limits at $r \to 0$ and $r \to \infty$. These defects are solved in part by using *contracted* Gaussian-Type Orbitals or GTOs. Contracted GTOs are defined as a linear combination of Gaussians with different exponents - referred to as ”primitives” -, multiplied by a common polynomial part,

$$\chi_{\mathbf{n}}(\mathbf{r}) = P_{\mathbf{n}}(\mathbf{r}) \sum_{k}^{K} D_k e^{-\alpha_k |\mathbf{r}|^2} \tag{2.36}$$

In GTOs, the contraction coefficients and the exponents of all Gaussians are usually optimized simultaneously. Gaussians can fit any decaying radial function with arbitrary precision as more primitives are used, and in the limit, STOs can be exactly represented through the following integral transform:

$$e^{-\alpha|r|} = \frac{\alpha}{2\sqrt{\pi}} \int_0^\infty s^{-3/2} e^{-(a^2/4)s^{-1}} e^{-s|r|^2} ds \tag{2.37}$$

The earliest and simplest type of GTO basis is the STO-nG series[59], where the STOs optimized for single atoms are fitted (by minimal residues) by an n primitive expansion. More recently developed GTO basis obtain the parameters directly from the optimizing energies with repect to the parameters.

Polynomial part

The polynomial part in GTOs use either simple powers of the three spatial coordinates (Cartesian GTOs or CGTOs) or solid spherical harmonics (Spherical GTOs or SGTOs). Both approaches are equivalent for angular momentum less than 2 (for s- and p- shells), but start diverging in number thereafter. Cartesian GTOs come in sets of T_{l+1} functions, with l the angular momentum and $T_n = \frac{1}{2}n(n+1)$ the n-th triangular number,

$$P^C_{\mathbf{n}}(\mathbf{r}) = N_{\mathbf{n}} x^{n_x} y^{n_y} z^{n_z} \tag{2.38}$$

$$n_x + n_y + n_z = l \tag{2.39}$$

whereas Spherical GTOs come in sets of $2l + 1$ functions, corresponding to the typical quamtum magnetic number of the hydrogenoid atom:

$$P^S_{lm}(\mathbf{r}) = N_{lm} |\mathbf{r}|^l Y^m_l(\theta, \phi) = N_{lm} R^m_l(\mathbf{r}) \tag{2.40}$$

$$-l \leq m \leq l \tag{2.41}$$

The extra functions in CGTOs can be linearly combined to form additional sets of spherical harmonics of lower order. These extra functions have no correspondence to the original STO basis, usually add little extra flexibility to the basis (since other functions are used to treat the smaller angular momentums), and increase the dimension of the linear algebra problems. Because of this, SGTOs are usually preferred, and can be obtained by linear combination of CGTOs.

Minimal vs. split zeta basis

The basis constructed from the juxtaposition of the atomic orbitals of the individual atoms is referred to as ”minimal” basis set. Minimal basis like the STO-nG described above are considered nowadays to be too crude to describe even the most basic molecular properties, chemical bonds and other features with acceptable accuracy. One of the first approaches aimed at increasing the flexibility of the basis in molecules was to use ”split-zeta” basis sets, popularized by Pople-type basis sets such as 3-21G[60]. The main idea is to keep the minimal basis for the core electron shells, which are usually quite insensitive to the chemical environment, and add extra fuctions to the valence shell to improve the bond description in different environments. Some Pople-style basis sets, still in use today, add the further restriction of using the same primitives (with different contraction coefficients) for the s and p shells of the same energy level. This design was the key to accelerate the evaluation of 2-electron integrals in the Pople-Hehre algorithm[61], as will be later discussed in more detail.

Polarization and diffuse functions

A common practice is to add polarization shells to a basis set. Polarization shells[62] are functions with angular momentum higher than the highest occupied atomic orbital in the isolated atom. Polarization shells improve the description of some higher-order features of the molecular orbitals due to effects other than the inmediate chemical environment. Diffuse functions[63] are functions with low Gaussian exponent that add flexibility in describing the actual, slower decay of the atomic and molecular orbitals (compared to regular GTOs), important in some systems, like the previously mentioned intermolecular interactions.

Correlation-consistent basis sets

Many classic basis sets were obtained from HF energy optimizations, and give good results for HF calculations. However, these parameters do not provide the best basis for correlated methods, which are more expensive to compute and therefore need a basis with the best convergence possible to the basis set limit. Correlation-consistent basis sets[64] were developed to address this issue; the parameters of correlation-consistent basis sets are optimized directly from minimizing the atomic energy with some correlated calculation. Hylleraas[65] was the first to note in his analysis of the helium atom wavefunction that the expansion in Slater determinants necessarily included orbitals of higher angular momentum. Correlation-consistent basis sets also include extra shells for all the angular momentums of the atom and higher, to treat the Coulomb hole appearing at higher levels of theory. The convergence of the wavefunction with increasing angular momentum is nevertheless slow unless the wavefunction ansatz is modified to make explicit use of the interelectronic distance, like in the $R12$ and $F12$ approaches[66 67], or in many DFT functionals.

General contraction basis sets

General contraction basis sets were originally proposed by Rafenetti[68] as a systematic way to converge in accuracy to the basis set limit with every extra shell. In this approach, all GTO shells of a given angular momentum are constructed from a common set of primitives, as oposed to optimizing every shell separately. The minimal basis set reproduces the energy of the atom computed with the full set of primitives, and every extra shell correspond to virtual orbitals of the atom increasing in energy.

$$\chi^j_{lm}(\mathbf{r}) = N_{lm} R^m_l(\mathbf{r}) \sum_k^K D^j_k e^{-\alpha_k |\mathbf{r}|^2} \tag{2.42}$$

This idea was expanded further by Almlöf and Taylor[69] in their Atomic Natural Orbital (ANO) basis sets, naturally extending the method to correlation-consistent basis sets with higher momentum components. ANO basis are obtained from the natural orbitals (the basis diagonalizing the 1-reduced density matrix) obtained from a correlated calculation of the atom, originally CISD. The atomic wavefunction will converge towards the limit in the fastest possible way, with every new function added to the basis. ANO-type basis sets are used in calculations where very high accuracy is needed, so the primitive sets used are usually much larger than in other basis sets with similar number of functions. This has traditionally limited their use, due to the bad scaling of integral algorithms with the contraction degree of the basis.

Even tempered basis sets

The exponent optimization of the primitive Gaussians is a non-linear problem which tends to be time-consuming, numerically-unstable, and grows fast with the number of parameters. Even-tempered basis sets[70] bypass these problems using fixed geometric progressions of the primitive exponents $\alpha_k = \alpha_0 \kappa^k$ and optimizing only the coefficients D_{kj} in the linear expansion. This is based on the observation that optimized primitive exponents tend to approximately follow this trend, and span the exponent space evenly. This approach is also more systematic, easier to extend, and avoids potential bias. One computational advantage of using even-tempered basis sets in QC calculations is the appearence of close-formula solutions for the molecular integrals, with explicit dependence on the index of the primitives.

Some more sophisticated approaches, like the well-tempered basis set[71], replace the geometric progression with a more complex recursive definition of the exponents. However, even-tempered basis sets have been proven to span most evenly the Hilbert space in the Coulomb metric[72], a useful property in QC considering that the Coulomb metric is the metric of choice in Density Fitting[73] and other methods requiring the minimization of the energy error.

Chapter 3

Integrals in quantum chemistry

The computationally intensive core of a quantum chemistry calculation can be divided between the linear algebra and the molecular integral evaluation.

The linear algebra typically consists of matrix-vector and matrix-matrix products, and matrix decompositions like singular value decompositions, diagonalizations, Cholesky factorizations and inversions. These operations are the building blocks of many methods in science and engineering, and there is already a number of libraries[74 75 12 76 77] with efficient implementations of these basic routines, carefully tuned for performance by hardware vendors and research labs for all common hardware architectures. There is in general little use in writing new implementations of any of these routines, as it makes the code *less* portable and almost certainly worse-performing than the reference implementation. However, chosing the appropriate linear algebra routine and using it correctly is not always easy, given the variety of libraries, algorithms, sparse representations, preconditioners, etc. for every linear algebra problem. The correct choice for the problem at hand can have a huge impact in the elapsed wall-time of the calculation.

Some advanced methods (correlated methods, in particular), involve higher-rank tensor operations for which standard libraries do not always exist, problems may not always fit in memory, and/or the sparse patterns may be too intricate. It might be necessary in some case to write the implementation of the required algebra operations, although general approaches to the problem already exist[78]. As usual, any knowledge about the structure of the problem that can be used to improve the design is worth exploiting, and many well-known techniques used in the optimization of rank-2 linear algebra problems (blocking, vectorization, etc.) can also apply to higher-rank operations.

The most fundamental tensors in the calculation contain as elements the projections of the one- and two-electron operators in the given basis, more usually referred to as Molecular Integrals. The efficiency and maturity of the dense linear algebra routines, coupled with the elevated computational cost of evaluating many integrals, generally make integral evaluation the most time-consuming part in SCF methods and some more advanced methods, even when the algebra has a worse asymptotic scaling with the system size ($\mathcal{O}(N^3)$ vs. $\mathcal{O}(N^2)$). Since the beginning of quantum chemistry, a lot of efforts have been focused in finding faster integral algorithms, whether analytic solutions or numerical approximations, appropriate for the scale of the problems and the limitations of the available hardware at the time. GTOs are the almost universal basis in such computations, but despite their history and the efficiency of the integration and implementation techniques discussed below, integral evaluation is still the main bottleneck in many calculations.

3.1 Fundamentals of integral evaluation with GTOs

A set of standard cartesian Gaussian-type orbitals (CGTOs) is defined as a sum of weighted Gaussian primitives times a simple monomial of the cartesian coordinates:

$$\phi_{\mathbf{a}}(\mathbf{r}, \mathbf{A}) = N_{\mathbf{a}}(x - A_x)^{a_x}(y - A_y)^{a_y}(z - A_z)^{a_z} \sum_k^K D_k e^{-\alpha_k |\mathbf{r}-\mathbf{A}|^2} \tag{3.1}$$

A set of CGTOs is uniquely defined by its angular momentum L, and the contraction coefficients D_k and Gaussian exponents α_k of its primitives. The total number of primitives K is referred to as its contraction degree. The values of n_λ are positive (or zero) integers, which add up to the angular momentum L. Note that because $n_x + n_y + n_z = L$ is constant, the Gaussian expansion and the polynomial part can be normalized separately. A shell of CGTOs can be expressed in shorthand notation in its $\langle bra|$ form as:

$$(\mathbf{a}| = \sum_k^K D_k \, [\mathbf{a}_k| \tag{3.2}$$

where $\mathbf{a}$ is the integer vector of the powers of its monomials, and $(x|$ and $[x|$ are the contracted function and its uncontracted primitives, respectively. We can consider a general contraction scheme, and express the functions in shorthand notation with:

$$(\mathbf{a}^j| = \sum_k^K D_k^j \, [\mathbf{a}_k| \tag{3.3}$$

but in general the indices will be omitted for simplicity and only be used to resolve otherwise ambiguous formulas.

Alternatively, it is possible to define a batch of spherical Gaussian-type orbitals (SGTOs) by substituting the polynomial part with a set of regular solid harmonics of order L.

$$\phi_l^m(\mathbf{r}, \mathbf{A}) = R_l^m(\mathbf{r} - \mathbf{A}) \sum_k^K D_k e^{-\alpha_k |\mathbf{r}-\mathbf{A}|^2} \tag{3.4}$$

$$R_l^m(\mathbf{r}) = \sqrt{\frac{4\pi}{2l+1}} |\mathbf{r}|^l Y_l^m(\hat{r}) \tag{3.5}$$

The index m is bound as usual to the admissible values for spherical harmonics $-L \leq m \leq L$. Like in the case of spherical harmonics, the complex solid harmonics R_l^m and R_l^{-m} are often combined to give two purely real functions.

$$C_l^{|m|}(\mathbf{r}) = \frac{1}{2}(R_l^{|m|}(\mathbf{r}) + R_l^{-|m|}(\mathbf{r})) \tag{3.6}$$

$$S_l^{|m|}(\mathbf{r}) = \frac{1}{2i}(R_l^{|m|}(\mathbf{r}) - R_l^{-|m|}(\mathbf{r})) \tag{3.7}$$

The main advantage of GTOs over STOs or any other radial function is the many existing analytic relations involving Gaussians that simplify integral evaluation. As previously pointed out, it is possible to express STOs as a continuous transform with a Gaussian kernel; after changing the integration order, it is possible to express any STO integral as a GTO integral, itself integrated over the Gaussian exponents. This approach is practical for simpler integrals, but the final integration steps in multi-center electron-repulsion integrals are not analytic even with this approach. Using multidimensional quadrature techniques to evaluate them is equivalent to expressing the STOs as a weighted sum of Gaussians in the first place.

Following are some of the most important results, analytic identities and notations useful in the evaluation of molecular integrals over GTOs.

3.1.1 Gaussian product theorem

The Gaussian product theorem states that the product of two Gaussians centered around two different points in space is another Gaussian centered on a third point, lying on the segment connecting the original points. Because Gaussian functions in N-dimensions factor into a product of N 1-dimensional Gaussians, this simplifies the handling of multicenter integrals considerably.

Specifically, the Gaussian product theorem for GTO primitives can be written as:

$$e^{-\alpha|\mathbf{r}-\mathbf{A}|^2} e^{-\beta|\mathbf{r}-\mathbf{B}|^2} = e^{-\frac{\alpha\beta}{\eta}|\mathbf{A}-\mathbf{B}|^2} e^{-\eta|\mathbf{r}-\mathbf{P}|^2} \tag{3.8}$$

with:

$$\eta = \alpha + \beta \tag{3.9}$$

$$\mathbf{P} = \frac{\alpha\mathbf{A} + \beta\mathbf{B}}{\eta} = \mathbf{A} - \frac{\beta}{\eta}\mathbf{AB} = \mathbf{B} + \frac{\alpha}{\eta}\mathbf{AB} \tag{3.10}$$

3.1.2 Cartesian polynomial translation

The polynomial part of each GTO needs also to be translated to the new center $\mathbf{P}$. This is acomplished either by direct expansion or by recurrence. The direct expansion is a corollary of Newton's binomial formula:

$$(x_\lambda - A_\lambda)^n = ((x_\lambda - P_\lambda) + AP_\lambda)^n = \sum_{i=0}^{n} \binom{n}{i} (x_\lambda - P_\lambda)^i (AP_\lambda)^{n-i} \tag{3.11}$$

Alternatively, the recurrence is more useful to derive algorithms:

$$(x_\lambda - A_\lambda)^n (x_\lambda - P_\lambda)^m = (x_\lambda - A_\lambda)^{n-1} (x_\lambda - P_\lambda)^{m+1} + AP_\lambda (x_\lambda - A_\lambda)^{n-1} (x_\lambda - P_\lambda)^m \tag{3.12}$$

which is often expressed, in $\langle bra|$ form the following way:

$$[(\mathbf{a} + \mathbf{1}_\lambda)\mathbf{p}| = [\mathbf{a}(\mathbf{p} + \mathbf{1}_\lambda)| + AP_\lambda[\mathbf{ap}| \tag{3.13}$$

where the appearence of two integer vectors implies the simultaneous expansion of monomial powers around the two different centers. Any such translation affects only the polynomial part of the CGTO batch, so it is irrelevant whether it is applied primitive by primitive or to the contracted function. In fact, applying it to the contracted function gives the well-known *Horizontal Recurrence Relation*, introduced in the HGP algorithm[79] as a particular case for two-electron integrals, following an entirely different approach.

$$((\mathbf{a}+\mathbf{1}_\lambda)\mathbf{b}| = (\mathbf{a}(\mathbf{b}+\mathbf{1}_\lambda)| + AB_\lambda(\mathbf{ab}| \tag{3.14}$$

3.1.3 Hermite Gaussian functions

Hermite Gaussian functions are the eigenfunctions of the Fourier transform. One of the most fundamental and remarkable properties of the Fourier transform is the conversion of the $x\cdot$ operator in real space into the ∂_ω operator in Fourier space and vice-versa, simplifying the algebraic manipulation of many problems. Evaluating batches of molecular integrals over different combinations of powers of the cartesian axes can be similarly simplified with the use of Hermite Gaussian functions.

While it is possible to express the product of two primitives GTOs as a new Gaussian times a cartesian polynomial expansion, it is sometimes more convenient to find an expansion in terms of the corresponding Hermite Gaussian functions, defined as:

$$H_{\mathbf{n}}(\eta, \mathbf{r}-\mathbf{P}) = (\partial_{P_x})^{n_x}(\partial_{P_y})^{n_y}(\partial_{P_z})^{n_z}e^{-\eta|\mathbf{r}-\mathbf{P}|^2} \tag{3.15}$$

Every successive application of a partial derivative increments the total degree of the polynomial part by one. Any complete set of Hermite functions of order up to $n_x + n_y + n_z \leq n$ necessarily spans the same polynomial subspace as the corresponding cartesian GTO shell products. The benefit of using Hermite functions as intermediates in some derivations is that the solution to higher-order integrals becomes straightforward; the only analytic solution required is for the simplest case (that of zero-angular momentum), and the rest can be solved from there by simple derivation. This was first noted by McMurchie and Davidson in their landmark paper about integral evaluation[80].

A CGTO with exponent η and vector powers $\mathbf{e}$ can be expanded in HGFs (denoted by $\overline{\mathbf{p}}$) around the same center with the simple recurrence:

$$[(\mathbf{e}+\mathbf{1}_\lambda)\overline{\mathbf{p}}| = p_\lambda[\mathbf{e}(\overline{\mathbf{p}-\mathbf{1}_\lambda})| + \frac{1}{2\eta}[\mathbf{e}(\overline{\mathbf{p}+\mathbf{1}_\lambda})| \tag{3.16}$$

which is a restatement of the classical recurrence relation for Hermite polynomials of general exponent. A product of CGTOs with centers **A** and **B** (denoted by **ab**) can be expanded in HGFs with center in **P** using any of the following two equivalent recurrence relations:

$$[(\mathbf{a}+\mathbf{1}_\lambda)\mathbf{b}\overline{\mathbf{p}}| = p_\lambda[\mathbf{ab}(\overline{\mathbf{p}-\mathbf{1}_\lambda})| + AP_\lambda[\mathbf{ab}\overline{\mathbf{p}}| + \frac{1}{2\eta}[\mathbf{ab}(\overline{\mathbf{p}+\mathbf{1}_\lambda})| \tag{3.17}$$

$$[\mathbf{a}(\mathbf{b}+\mathbf{1}_\lambda)\overline{\mathbf{p}}| = p_\lambda[\mathbf{ab}(\overline{\mathbf{p}-\mathbf{1}_\lambda})| + BP_\lambda[\mathbf{ab}\overline{\mathbf{p}}| + \frac{1}{2\eta}[\mathbf{ab}(\overline{\mathbf{p}+\mathbf{1}_\lambda})| \tag{3.18}$$

named *transfer equations* by Gill, Head-Gordon and Pople.

3.1.4 Laplace transform of the Coulomb potential

The Coulom potential $1/r_{12}$ appearing in electron repulsion integrals and nuclear potential integrals among others has a pole at $r_{12}=0$, and the form of such integrals over Gaussian densities has no obvious immediate solution. Fortunately, the Laplace transform of the potential brings a much more useful form:

$$\frac{1}{|r_2 - r_1|} = \frac{2}{\sqrt{\pi}}\int_0^\infty e^{-s^2|r_2-r_1|^2} ds \tag{3.19}$$

Changing the integration order, integrating over the spacial coordinates first and leaving the integration over s for last, greatly simplifies the solution of such integrals. For instance, the integral:

$$V(\eta,\zeta,\vec{P},\vec{Q}) = \int_{V_1}\int_{V_2} \frac{e^{-\eta|\vec{r}_1-\vec{P}|^2} e^{-\zeta|\vec{r}_2-\vec{Q}|^2}}{|r_2 - r_1|} d^3\vec{r}_1 d^3\vec{r}_2 \tag{3.20}$$

between two Gaussian charges can be integrated over the volumes to

$$V(\eta,\zeta,\vec{P},\vec{Q}) = \frac{2}{\sqrt{\pi}}\pi^3 \int_0^\infty \frac{e^{-\frac{\eta\zeta s^2}{\eta\zeta+(\eta+\zeta)s^2}|\mathbf{PQ}|}}{(\eta\zeta+(\eta+\zeta)s^2)^{3/2}} ds \tag{3.21}$$

Substituting $t^2 = s^2/(\theta + s^2)$ with $\theta = \frac{\eta\zeta}{\eta+\zeta}$ and simplifying some terms, we arrive at:

$$V(\eta, \zeta, \vec{P}, \vec{Q}) = \frac{2\pi^{5/2}}{\eta^{3/2}\zeta^{3/2}} \Theta^{1/2} F_0\left(\Theta|\mathbf{PQ}|^2\right) \tag{3.22}$$

$$F_m(z) = \int_0^1 t^{2m} e^{-zt^2} dt \tag{3.23}$$

Where the function $F_m(z)$ is known as the Boys' function or the chemists' incomplete gamma function. An alternate, but ultimately equivalent formulation to the one presented here for ERI algorithms is the *Rys quadrature*[81]. Its approach is to construct an exact quadrature with weights and points w_n^* and t_n^* and directly evaluate the integrals of the type

$$\int_0^1 e^{-zt^2} P(t) dt = \sum_n w_n^* P(t_n^*) \tag{3.24}$$

The damped potential operator $erf(\omega|r_{12}|)/|r_{12}|$ or its complementary $erfc(\omega|r_{12}|)/|r_{12}|$ are related to the regular Coulomb potential by a trivial modification of the integration limits of its Laplace transform.

3.1.5 Rotation of a solid harmonic expansion

Any rotation in 3D euclidean space can be uniquely determined by the axis of rotation and an angle. General rotations can be also expressed in terms of their three Euler angles (α, β, γ), which decompose the rotation matrix into three simpler rotations around cartesian axes. Of the many possible choices for the axes, the following is the most followed convention is quantum mechanics:

$$R(\alpha, \beta, \gamma) = R_z(\alpha) R_y(\beta) R_z(\gamma) \tag{3.25}$$

The rotation of a set of spherical harmonics is represented by the Wigner D-matrix:

$$D^j_{m'm}(\alpha, \beta, \gamma) = \langle jm'|R(\alpha, \beta, \gamma)|jm\rangle = e^{im'\alpha} d^j_{m'm}(\beta) e^{-im\gamma} \tag{3.26}$$

with $d^j_{m'm}$ being the Wigner d-matrix.

The rotation of the spherical harmonic (l, m) around its z-axis amounts to a linear combination of (l, m) and $(l, -m)$. Because of this simplicity, the z-axis is always the preferred axis for composing more complex rotations. To rotate a complete spherical harmonics expansion, it is necessary to generate the $(2l + 1) \times (2l + 1)$ rotation matrices, either implicitely or explicitely. Chosing $0, -1, +1, -2, +2, \dots, -l, +l$ for the order of coefficients the matrix becomes 2×2 block diagonal:

$$Z_l(\alpha) = \begin{vmatrix} 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & \cos(\alpha) & \sin(\alpha) & \dots & 0 & 0 \\ 0 & \sin(-\alpha) & \cos(\alpha) & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & \cos(l\alpha) & \sin(l\alpha) \\ 0 & 0 & 0 & \dots & \sin(-l\alpha) & \cos(l\alpha) \end{vmatrix}$$

To perform the next rotation around the (new) y-axis while preserving the relative simplicity of the previous z-rotation, it is also necessary to introduce an orthogonal matrix J_l known as the one corresponding to the $(xyz) \to (yzx)$ index permutation, or equivalently, its inverse $J_l^{-1} = J_l^T$. The rotation around the y-axis becomes simply:

$$Y_l(\beta) = J_l^T Z_l(\beta) J_l J_l^{m'm} = \langle lm' | P_{yzx} | lm \rangle \tag{3.27}$$

The last step is another rotation around the (new) z-axis. The total rotation of the expansion can be expressed as the product of five matrices[82]:

$$R_l(\alpha, \beta, \gamma) = Z_l(\alpha) J_l^T Z_l(\beta) J_l Z_l(\gamma) \tag{3.28}$$

The first (rightmost) rotation moves the axis z_R until it is made coplanar with the $xz-$ plane. The second, moves z_R and makes it colinear with the z'-axis. The final rotation makes the axes x'' and y'' coincide with x_R and y_R.

Any cartesian rotation has a constant $\mathcal{O}(1)$ computational cost, so the total cost of such rotations is $\mathcal{O}(L)$ or $\mathcal{O}(L^2)$, depending on whether the expansion includes only one shell of angular momentum or all angular momentums up to L. The cost of transforming the cartesian axes is that of a dense matrix-vector product (albeit many of the entries are zero), and therefore the total cost of rotating the expansion is $\mathcal{O}(L^2)$ or $\mathcal{O}(L^3)$, respectively. If the axis of rotation z_R is known in advance, but not the angle, it is possible to precompute the first four steps and perform the last $\mathcal{O}(L)/\mathcal{O}(L^2)$ rotation once the angle is known.

3.1.6 Translation of a solid harmonic expansion

The translation theorem of solid harmonics gives a closed formula for the translation of a 3D polynomial expressed as an expansion of regular solid harmonics[82].

$$R_l^m(\mathbf{r}+\mathbf{a}) = \sum_{\lambda=0}^{l} \sum_{\mu=-\lambda}^{\lambda} C_{l,\lambda}^{m,\mu} R_\lambda^\mu(\mathbf{r}) R_{l-\lambda}^{m-\mu}(\mathbf{a}) \tag{3.29}$$

with the expansion coefficients defined by:

$$C_{l,\lambda}^{m,\mu} = \binom{l+m}{\lambda+\mu}^{1/2} \binom{l-m}{\lambda-\mu}^{1/2} \tag{3.30}$$

Notice that translations involving only the z-component simplify the calculation of the above expression from $\mathcal{O}(L^4)$ to $\mathcal{O}(L^3)$.

$$R_l^m(\mathbf{r}+a\mathbf{e_z}) = \sum_{\lambda=0}^{l-|m|} C_{l,\lambda}^{m,0} a^\lambda R_{l-\lambda}^m(\mathbf{r}) \tag{3.31}$$

The better scaling of the second expansion makes it the method of choice in algorithms using extensive multipole expansions, as pointed by Greengard in[83]. To use the second method, the general translation has to be decomposed into a $\mathcal{O}(L^3)$ rotation of multipole expansion to make its z-axis coincide with the translation vector, followed by the $\mathcal{O}(L^3)$ translation along the z-axis, and a final $\mathcal{O}(L^3)$ rotation recovering the original orientation.

3.2 Screening

Screening methods are fundamental to achieve high performance in direct SCF methods. One fundamental observation is that the integrals over GTO products decay faster-than-exponentially with increasing separation of the centers. From the Gaussian product rule, it is clear that the higher the exponent of a primitive Gaussian (sharper/more localized function), the faster the products with primitives from other centers decay with distance and the sooner they become negligible. For large enough molecules, it is possible to discard a majority of GTO pairs on this basis alone. Even amongst surviving pairs, and since GTOs may and usually do contain primitives with exponents spanning several orders of magnitude, it is also beneficial performance-wise to test every primitive product of the pair and discard those below some threshold. This procedure reduces the effective contraction degree of the shell pairs, which is direcly related to the evaluation cost of its integrals. For most practical purposes, it is possible to discard the primitive pair using:

$$\begin{aligned} e^{-\frac{\alpha\beta}{\alpha+\beta}|AB|^2} &< \tau && (3.32) \\ &\text{or} \\ |AB|^2 &> -\log(\tau)(\alpha^{-1}+\beta^{-1}) && (3.33) \end{aligned}$$

with typical values of $\tau \sim 10^{-20}$. This threshold is not mathematically rigorous, considering that it omits the value of the contraction coefficients and the extent of the Gaussian density associated to the primitive product. However, even if the error due to it was of a few orders of magnitude, the cutoff distance would not be altered by more than a few percent of the theoretically correct distance. In practice, interatomic distances in a molecule tend to cluster around some particular values more than being spread in a continuum, and it is hard to imagine a situation where a few percent over or under some distance might create any significant numerical error.

Another useful screening method for discarding whole batches of integrals is based on the Cauchy-Schwarz inequality. It usually takes the form:

$$\langle a|\hat{O}|b\rangle^2 \leq \langle a|\hat{O}|a\rangle\langle b|\hat{O}|b\rangle \qquad (3.34)$$

where the operator $\hat{O}$ can be viewed as a metric of an inner product space, and is therefore required to be symmetric positive definite. The Cauchy-Schwarz inequality is rather general; it applies from vector products and Frobenius norms to convolutions and integrals over L^2 complex functions.

3.3 Efficient evaluation of molecular integrals

The resurgence of vector-capable hardware, the complex cache hierarchies and the adoption of radically different special-purpose architectures, combined with the evolution of integral algorithms requires a careful re-assessment of the traditional methods for integral evaluation. The rest of the chapter will introduce a few general key concepts used in the implementation of the integral codes presented in this thesis.

3.3.1 Elementary Basis Set

Contemporary hardware is ideally suited for integral batch processing designs based on *Elementary Basis Sets* and other related ideas. EBS is a way of classifying and packing the necessary integrals that has been described in several papers[84][85][86] under different names, or without any explicit name. The name employed in this work is derived from the term *Elementary Basis Algorithm*, which was introduced in the first paper[84] explicitely describing the idea.

The elementary basis set is the generic set of atom-centered basis functions defined for a given element in a calculation, not to be confused with the basis set itself, which is the set of all the functions in the system used to represent the wavefunction. The same way all atoms in the system are instances of some element, every basis function or shell in the system is an instance of some elementary basis function (or shell). This distinction is quite relevant to the implementation of the integral codes for the following reasons:

- A majority of simulations contain no more than a few different elements. It is rare for chemical systems (or at least the useful ones) to be composed by more than 5 or 6 elements. With very few exceptions, the norm in QC computations is to assign the same elementary basis to all atoms of the same element.

- Molecular integrals involving the same set of elementary shells share the same angular momenta, contraction degree, general contraction degree, weights, exponents, etc. Computing the integrals of the same EBS tuple together expose all the potential data-level parallelism amongst integrals. Many intermediate constants can be computed fewer times because they are reused for all the integrals computed in the same block. The same batch screening bounds and primitive screening methods are also applicable to all the integrals in the block.

- Vector and stream computing perform best when the data is accessed in regular patterns, and when the branch followed by the control instructions has a pattern that can be correctly predicted by the CPU, or in the case of GPUs, when all threads in a warp execute the same path.

The EBS approach is used together with the integral geometry classification described next.

3.3.2 Integral Geometry and Rotations

The geometry of an integral refers to the degeneracy of the centers that appear in the integral. QM/CMM and ECP integrals involve up to three centers, and their different possible geometries are, up to symmetry, $(AA|A)$, $(AA|B)$, $(AB|B)$ and $(AB|C)$. ERIs can involve up to four centers, and can be classified into seven geometries: $(AA|AA)$, $(AA|AD)$, $(AB|AB)$, $(AA|BB)$, $(AA|CD)$, $(AB|AD)$ and $(AB|CD)$. The algorithms presented here use the non-degenerate centers from the 3- and 4-center geometries to define a frame of reference in which several geometric parameters vanish, simplifying both the algebraic manipulation and the number of FLOPs necessary to evaluate them. The resulting integrals must be rotated to the lab frame of reference where the molecule is defined.

The classification of integrals according to their geometry is necessary to avoid numerical errors in the rotation matrices of degenerate geometries, and also because degenerate geometries can use simpler and faster versions of the integral routines. Having more than one function from one atom, same-center GTO products might have product densities and integrals many times larger than the average case. They are also likely to contribute to the block-diagonal of the tensor, where numerical errors propagate faster to other parts of the code.

In conclusion, classifying by EBS tuple and geometry accelerates integral evaluation and minimizes numerical errors.

3.3.3 Cache Line Vectorization

Cache Line Vectorization consists in packing (shuffling) the data corresponding to the variables of different integrals in the usual vector form, but using the number of elements that fits in exactly one line of L1/L2/L3 cache (x86: 64bytes) instead of matching the length of the processor's vector unit (SSE2: 16 bytes / AVX: 32 bytes) as it is commonly done. There are multiple benefits to this design:

- The data loaded in any line of cache is either all useful or all evictable. This reduces data fragmentation in the cache to practically zero.
- It simplifies design by collapsing one level of variable granularity (it depends on the vector instruction set) unto another that's nearly constant in all hardware implementations. With data being aligned to cache boundaries and

packed, there cannot be contention between processors for one line of cache. Vectors are automatically aligned to the most effective offset for loading and storing.

- Close to peak ALU utilization. Contemporary x86 vector ALUs have a latency of 4 cycles and throughtput of 1 cycle for ADD, MUL and FMA. Processing some operation on all the data in the cache line sequentially (one vector after another) guarantees that the SSE vector ALU will recieve new data every cycle, and that there are no chain dependencies between the data wasting CPU cycles (64 bytes = 4 * 2 doubles/SSE * 8 bytes/double). Routines with a high CCR and enough cache to store the required data achieve nearly peak FLOP performance with SSE2. AVX-capable processors consume twice as much data per cycle (32 bytes = 4 doubles * 8 bytes/double), while the cache line is unchanged, which can only guarantee half the occupation, although it is possible to regain full occupation in Intel processors by using two threads per core with Hyperthreading (at the cost of sharing the cache). This solution lies somewhere between the SSE2 vectorization and how the Intel MIC works. The Intel MIC supports the AVX-512 instruction set, which can initiate a vector operation per cycle using all 64 bytes (8 doubles). However, each physical core is concurrently executing up to 4 threads, which continuously feed the ALU, and effectively hide the latency. For CUDA GPUs, the data is packed in blocks of at least 32 elements instead, which corresponds to the size of a CUDA warp, as well as 32x8=256 bytes being the alignment of the GPU memory banks. Many kernels need at least twice to four times the number of threads per core to avoid starvation.

3.3.4 Automated code generation

Maximizing the performance of numerical software involves at some point aggressive code optimization of the innermost routines. Integral routines are ideally suited for some of these techniques: they involve multiple functions potentially spanning many combinations of angular momenta, but at the same time these values are low enough so that propagating their value at compile-time could cause big code simplifications. Furthermore, the state-of-the-art algorithms for GTO integrals introduced and referenced in this work involve for the most part convoluted recurrence relations with irregular memory access patterns. Integral routines capable of treating general cases cannot be efficient for any particular case.

If the program needs relatvely few calls to the integral routine, the potential increase in complexity may not be justified by an unnoticeable gain in performance, but if the call originates in one of the innermost loops, the difference in performance can be dramatic and such complexity may be more than justified.

When the parameter space of some of the input variables is relatively small with boundaries known in advance, the routine should be specialized for at least the most common cases. Automating this procedure (rather than doing it manually) is the best strategy to avoid hard to dectect errors and maintainance issues. The simplest cases can be handled by the compiler with function and class templates, or equivalent. The pointers to the specialized routines can be stored in an array, from where the appropriate specialization is retrieved for every call to the function.

Integral routines tend to benefit considerably from these techniques. Unrolled or specialized routines further expose a number of potential optimizations to the compiler such as constant folding and common subexpression elimination, fewer to no control structures improve the speed of small routines by avoiding potential branch mispredictions in the processor's pipeline, and a more compact code that fits better in the L1 instruction cache.

More complex cases require code generation to some degree, because compilers tend to perform rather poorly at generating code from complex recursive definitions (when they do not crash). There are also a practical limitations to the number of lines of code a compiler can handle. There are two solutions to this issue: to directly generate assembly/machine code or to use bytecode. Generating assembly is not difficult in itself, and unrolled intensive numerical code is expected to run at about the same speed regardless of whether it is optimized by the compiler or not. In case of stalling, the Out-of-Order execution will reorder the microop queue if necessary to deal with the instructions required for the control of the program flow in advance. The problem with assembly/machine code is that the length of machine x86 vector instructions, times the number of FLOPs per formula, times the number of vectors per cacheline likely exceeds the size of the L1 instruction cache, which has to compete with the routine's data for L2 cache space and bandwidth. The bytecode, on the other hand, can be made more compact than machine code, while only requiring a minimal interpreter code to execute it.

Chapter 4

Evaluation of 2-electron repulsion integrals

2-electron repulsion integrals (ERIs) are the most computationally demanding type of molecular integral commonly used in quantum chemistry simulations. The search for efficient ERI evaluation schemes has guided the evolution of basis sets[61 87] and QC codes alike[88]. ERIs are typically evaluated in batches of all the possible combinations of angular momenta of the four shells participating in the batch. This accelerates their evaluation due to the many common intermediate variables. Over the years, many algorithms have been developed to evaluate ERIs efficiently, either for particular special cases or as general approaches. However, every analytic solution requires in one way or another some summation over all four contraction indices K_A, K_B, K_C, K_D, which make high-quality basis sets with high contraction degrees prohibitively expensive in many cases.

$$(\mathbf{ab}|\mathbf{cd}) = \sum_{i}^{K_A}\sum_{j}^{K_B}\sum_{k}^{K_C}\sum_{l}^{K_D} D_i D_j D_k D_l \; [\mathbf{a}_i\mathbf{b}_j|r_{12}^{-1}||\mathbf{c}_k\mathbf{d}_l] \tag{4.1}$$

After the publication of the first algorithms able to treat general cases[81 80 89], a series of efforts initially lead by John A. Pople, Martin Head-Gordon and P.M.W., Gill focused on the developement of faster algorithms for treating high degrees of contraction that outperform the naïve method of accumulating the integrals inside the summation loops. Their investigations culminated in two related general algorithms with much lower prefactors for the asymptotic $\mathcal{O}(K^4)$ scaling, namely the PRISM[90] and BRAKET[91] algorithms. These and other fast algorithms based on similar ideas are nowadays referred as ”early contraction algorithms”, and lie at the core of many QC codes. Although their lower asymptotic prefactor made certain basis sets with higher contraction degrees more popular, the use of general contraction and other higher-quality basis sets remain still today a relative rarity due to their high cost of evaluation. One of the main issues of early contraction algorithms is that they tend to have a high constant computational cost, i.e. a computational penalization for integrals of relatively low contraction degree. This forces in many instances the inclusion of multiple algorithms optimized for different types of ERI, complicating the design of the ERI evaluation module.

Except for some notable exceptions[92], the focus in development ERI algorithms has nowadays shifted torwards finding new better approximations[93] or alternative ways to bypass their computation in practical calculations. Some of these alternatives deliver high performance with good typical errors, acceptable within the application. However, the complete ERI batches are still needed in many calculations of higher precision, such as some correlated computations, where every potential source of numerical error needs to be either well-bounded or eliminated.

4.1 New ERI algorithms

The K4+MIRROR algorithm is a new ERI algorithm developed and presented fit the first time in this thesis, designed for high contraction degrees. It works in two major steps: the *K4* contraction of ERI kernels and the *MIRROR* transformations. The first step performs the contraction of the kernels maximally exploiting the quadruple nested loop structure to avoid unnecessary computations; it has a complexity of $\mathcal{O}(K^4JL^2)$ for the general four-center case. The MIRROR (*Mixed and Interleaved Recurrence Relations Over Rotated axes*) steps are carried out outside the loops, and make effective use of frame rotations and some modified recurrence relations to keep the FLOP count down to a minimum; its complexity is $\mathcal{O}(J^4L^8)$, which contrary to most other early contraction algorithms, is usually low enough to treat low contractions at a negligible extra cost. Its details are discussed in the first paper.

The number of FLOPs in the K4 contraction step can be further reduced by applying the CDR/AERR recurrence relations, discussed in the last paper. Their use reduce the complexity of the innermost loop to $\mathcal{O}(K^4JL)$. In the (typical) case of spherical GTOs, it is also possible to skip the accumulation of several kernels, as well as to simplify or eliminate a few recurrence relations involving them later on.

The choice in the order of application of the MIRROR recurrence relations as well as the particular application of the CDR/AERR inside the K4 contraction loops are based purely on heuristics. Both happen to work reasonably well for all tested cases. This is not to say that they are the optimal application of the RRs for any case. Other RR block orders sometimes produce slightly lower FLOP counts, especially for low angular momentum routines. In particular, it is quite possible that mixing RRs from different blocks might produce better FLOP counts. The form of the CDR and 1-center RR are also strongly indicative that it is possible to combine both recurrences and reduce the cost of the contracted CDR + MIRROR steps.

The asymptotic complexity of contraction can be further reduced down to $\mathcal{O}(K^4 Jlog(L))$ for a particular case of even-tempered basis sets, but the more difficult implementation and the relative low values of L are likely to render the algorithm useless for practical purposes.

4.2 Implementation

The K4+MIRROR algorithm has been implemented in the *Quimera* library (a Query-based unified interface for managing electron repulsion algorithms) and optimized for both x86 and NVIDIA Fermi architectures. The latter is still in an earlier stage of development.

4.2.1 Steps of evaluation

In the COLD PRISM[94] nomenclature, the computation of integrals is divided in four steps: the O- step is the computation of the Boys' function and generation of 1-center integrals; the C-step is the contraction of 1-center integrals to ERI kernels (CDR/K4 step in the algorithm); the L-step is the transformation of ERI kernels to the ERI batches (MIRROR transformations); the final D- step is the digestion (use) of the integrals, not discussed in the current chapter. To make the best possible use of instruction caches and expose regularity, ERIs are processed in blocks, and every step is performed on the totality of the block before starting the next one. Between steps, a memory buffer is used to store the intermediate values.

4.2.2 Automatic code generation

The recurrence relations of the CDR/K4+MIRROR show little regularity in their memory accesses, which due to the many possible combinations of angular momenta would make an efficient general implementation hopeless. Instead, the recurrence relations and intermediate integrals necessary for solving each individual case of angular momenta and geometry are backtracked by the code generator. Once the dependency graph is generated, a simple procedure carries out some basic simplifications like eliminating identities or propagation of zeros. Because the memory is managed entirely by the code generator, it is important to minimize the number of simultaneously alive intermediate variables during the MIRROR step. One basic heuristic in this process is to group the recurrence relations in blocks according to their invariant indices (the indices not directly participating in the recurrence relation), which helps reduce the number of intermediate variables staying alive[95].

Once the evaluation order is generated, there are two alternative modes of execution. The generator can write a source with the optimized code for the specific routine, which can later be compiled and linked to the library. Alternatively, it can generate a compact bytecode and store it to a file. The bytecode files are read when the library is initialized and stored in memory, and the bytecode is interpreted whenever that specific routine is called.

VILIC

VILIC (Variable Instruction Lenght Interpreted Code) is a bytecode format used by Quimera to store and interpret the contraction step K4 or CDR/K4 of ERI routines, defined by a list of kernel contractions and recurrence relations. Each instruction starts with the address of the destination to be updated, followed by a variable number of sources and optionally ending with some integer. The variable length of the bytecode is due to the very different nature of the frequently accessed contractions (using 2 addresses + 1 integer) and AERR (3 addresses) on one side and the much less frequently used CDR recurrences, which vary from 3 to 9 addresses. Using the minimal length for every instruction helps minimize the memory footprint and minimize contention over cache utilization, since the cache is being simultaneously used to store intermediate integrals.

The recurrence relations of the MIRROR transformations are performed in blocks of the same basic kind, executed sequentially. Every individual RR starts with the destination address (relative to the position 0 of the buffer used), followed by the addresses of the more operands, and depeding on the class or RR, ending with the auxilliary integer.

No loss of performance is observed from the extra processing needed to compute addresses. The computation of addresses, bookkeeping of IC pointer and format conversion are probably overlapped in the CPU with the FLOP intensive job.

4.2.3 Optimization for x86

O-step

The Boys' function is computed inside a quadruple-nested loop over the primitives. Two quantities are computed: the value of the incomplete gamma of the total angular momentum and an exponential. The rest of the values can be computed from those by recurrence. The x86 implementation of the Boys' function uses a lookup table with derivatives for the short-range part and an inverse square-root followed by several FLOPs for the long-range part. The short-range calls are computed using a four-term Taylor expansion around the closest tabulated point. The table range is from 0 to 32, with 32 divisions per unit; it requires four doubles per point (the value + 3 derivatives), totalling 32Kb of memory. The exponential is computed from another Taylor expansion, which reuses the computed powers.

$$F_m(z^* + \Delta z) \simeq F_m(z^*) + \Delta z F_{m+1}(z^*) + \frac{(\Delta z)^2}{2} F_{m+2}(z^*) + \frac{(\Delta z)^3}{6} F_{m+3}(z^*) \quad (4.2)$$

$$e^{-(z^*+\Delta z)} = e^{-z^*} e^{-\Delta z} \simeq e^{-z^*} \left(1 + \Delta z + \frac{(\Delta z)^2}{2} + \frac{(\Delta z)^3}{6} \right) \quad (4.3)$$

To help reduce the memory needed for buffering beetween the routine computing the Boys' function and the K4 contraction, only the highest needed $F_m(z)$ is contracted with the usual weight and stored, along with the exponential used in the recursion:

$$[m_{ijkl}] = \frac{2^{1/2}\pi^{5/2}}{\eta_{ij}^{3/2}\zeta_{kl}^{3/2}} e^{-\frac{\alpha_i\beta_j}{\eta_{ij}}|AB|^2} e^{-\frac{\gamma_k\delta_l}{\zeta_{kl}}|CD|^2} (2\theta_{ijkl})^{m+1/2} F_m(\theta_{ijkl}|P_{ij} - Q_{kl}|^2) \quad (4.4)$$

$$[\overline{(m-1)}_{ijkl}] = \frac{2^{1/2}\pi^{5/2}}{\eta_{ij}^{3/2}\zeta_{kl}^{3/2}} e^{-\frac{\alpha_i\beta_j}{\eta_{ij}}|AB|^2} e^{-\frac{\gamma_k\delta_l}{\zeta_{kl}}|CD|^2} (2\theta_{ijkl})^{m-1/2} e^{-\theta_{ijkl}|P_{ij}-Q_{kl}|^2} \quad (4.5)$$

the rest of the one-center integrals are computed by downward recursion in the K4 contraction routine.

$$[m_{ijkl}] = \frac{1}{2m+1} \left(|PQ|^2 [(m+1)_{ijkl}] + [\overline{m}_{ijkl}] \right) \quad (4.6)$$

With the CDR/AERR recurrence relations, these integrals are used directly.

4.2.4 C and L steps

Perhaps surprisingly, the use of interpreted code is faster than compiled code for many cases, and both can reach up to 50% of peak performance in modern x86 CPUs, for relatively low total angular momentum. This is fundamentally due to the rational use of the CPU cache combined with the high instruction-level parallelism of the CPU, which is able to simultaneously execute the vector floating point arithmetic and compute the addresses of the following instructions. For high total angular momentum, bytecode is the only option, as the generated sources can contain up to millions of lines of code.

The implementation uses the L1/L2 cache basically as a vector file where old variables are being overwritten by new ones. The generator attempts to create a topological sort (instruction order) such that intermediate variables are used as soon as possible to free the memory for other variables. The destination in memory of a recently computed variable can be chosen to minimize the chance of the destination having been spilled from the cache since its last use, according to some model of cache eviction policy.

4.2.5 Optimization for CUDA

Several modifications over the basic implementation show substantial improvement in the CUDA architecture. The most notable ones relate to the evaluation of the Boys' function $F_m(z)$, using redundant computations to save bandwidth and/or avoid random memory accesses, and using the constant memory to accelerate some computations.

OC-step

To avoid unnecessary memory use and GPU bandwidth, the computation of the Boys' function and the K4 contraction are merged into a single OC kernel in the GPU implementation.

The GPU code uses the hardware implementation of the exponential, as it is optimized. The lookup table used for the CPU implementation could fit in the GPU's 64Kb of available constant memory, but it would exceed the only 8Kb of cached constant memory per multiprocessor. Moreover, different threads within a warp are almost certain to use different (albeit close) values from the table, which would force most accesses to constant memory to be serialized, impacting performance. In this design, the tables would need to be reloaded before every OC kernel call, because the total angular momentum might differ from one call to the next.

Instead, the CUDA version substitutes the Taylor expansion with a Chebyshev expansion, where the order and number of tabulated points of the table have been simultaneously optimized to minimize memory while keeping the maximum numerical error under 10^{-15}. An 8-th order expansion with a total of 64 points suffices, giving a total size of 4Kb per table. Such design uses more FLOPs than the CPU code, but has many advantages. First, it allows up to 16 tables in constant memory, enough to store the tables needed for up to $(ff|ff)$ angular momentum, avoiding the costly movement of small data. Second, it allows two complete tables to reside simultaneously in the constant cache, without spilling data during the kernel execution. Finally, dividing the range in 64 intervals increases the likelihood of multiple threads accessing the same values, reducing serial access. All of the above also permits the concurrent execution of the CUDA kernel by multiple streams with different L.

The values of $e^{-\frac{\alpha\beta}{\eta}|AB|^2}$ and $e^{-\frac{\gamma\delta}{\zeta}|CD|^2}$ are recomputed when needed. The problem with storing these values in memory is not in this case the saturation of memory bandwidth, but the irregularity of the memory access patterns required. The screening of primitive integrals is computed directly in the K4 loops, at the cost of few FLOPs. To achieve high performance in screening, the ERI batches need to be packed in any case, since the warps' ability to handle divergent execution is limited, and is done by executing in serial all paths.

Before every call to the OC-kernel, all the batch-independent constant data required is computed by an initialization function, and placed in the device's memory. It includes contraction coefficients, powers of Gaussian exponents, θ_{ijkl} , constants used in primitive screening, etc. Since all threads in the block use the same constants at a given time, these values can be loaded from the device memory with the optimized LDU (load uniform) instruction.

L-step

The MIRROR transformations tend to have a high computation-to-communication ratio ($\mathcal{O}(L^8)/\mathcal{O}(L^4)$), and are ideally suited for GPU processing, assuming there's enough local memory to store the intermediate results. Benchmarks done with up to d-functions ($L = 8$) show a negligible time spent in this step compared to the rest of steps of the computation.

Chapter 5

Evaluation of Coulomb and Exchange matrix elements

The Self-Consistent Field (SCF) methods previously discussed - Hartree-Fock and (hybrid) DFT - are the core of a majority of quantum chemical programs. The computation of the Coulomb and Exchange matrices is the principal bottleneck of SCF methods, response theory and some correlated methods, in particular those based on local schemes, density fitting or resoluton of identity. Due to the scale of many chemical problems and the widespread use of parallel machines, it is nowadays not only crucial to achieve high efficiency per-CPU core, but also to find strongly scalable solutions capable of utilizing multicore processors and multinode clusters.

In its most basic form, the computation of Coulomb and exchange matrices can be expressed as the contraction between a rank-4 tensor and a rank-2 tensor. Tensor contractions and products are considered "embarrasingly parallel" tasks, because most operations can be divided almost perfectly amongst any number of independent tasks (reasonable for the dimensions of the problem) and only required minimal communication at the very end, when the final reduction across nodes is performed.

$$J_{ij} = \sum_{k,l}^{N,N} \rho_{kl}\,(ij|kl) \tag{5.1}$$

$$X_{ik} = \sum_{j,l}^{N,N} \rho_{jk}\,(ij|kl) \tag{5.2}$$

In this sense, every contraction of a (rank-2) sub-block of the input density matrix D with a (rank-4) ERI batch is also independent of any other. The computation-to-communication ratio is very favorable in general because both input and output matrices are of size $\mathcal{O}(N^2)$, while computation is $\mathcal{O}(N^4)$. Even for large systems where typical integral screening discards nearly all but $\mathcal{O}(N^2)$ integrals, the prefactor hidden in the Big-$\mathcal{O}$ notation is large enough to hide communication costs.

In practice, however, good scaling is not so simple to achieve due to a number of factors, which can be roughly classified into algorithmic-related, hardware-related and problem-related. The algorithmic factors are due to the advances in the performance of state-of-the-art SCF algorithms, which tend inevitably to trade better theoretical performance for increased software complexity, making the performance problem more difficult. The hardware-related factors are those due to the advances in hardware architectures, accelerator cards, node-to-node communication, etc., which similar to the algorithmic category end up trading performance for programming simplicity. The last category encompasses everything related to the potential diversity of the spectrum of inputs, like the basis set and elements used in the calculation and the spatial distribution of atoms. For instance, finding an appropriate load balancing of the work for heterogeneous chemical systems such as metalloproteins or surface-adsorbed molecules, deciding sensible atom references for a z-matrix representation or reordering the atoms to achieve bandwidth minimization in sparse matrices.

5.1 Modern methods for Coulomb and Exchange formation

Many of the issues mentioned can indirecly affect other aspects of the code. Attempts to address all problems simultaneously result in design constraints interacting at different levels of the software design, some of which require some degree of compromise. However there are a few very relevant factors which impact the design at the largest possible scale and have to be considered before other more fine-grained details. The following considerations are central to modern hardware and SCF methods:

5.1.1 Direct methods

Because of the large number of basis functions, the number of ERIs in many problems exceed available memory and even disk space. Modern SCF methods[96] recompute, use and discard the Coulomb and Exchange matrices required at every SCF step. Due to the cost of ERI evaluation, this approach is almost invariably associated with integral screening methods or some more advanced technique. A useful alternative that has found its way up to correlated methods is to perform an incomplete Cholesky factorization of the ERI matrix, computing only enough Cholesky vectors to approximate the matrix within some predefined accuracy. The procedure can be carried out even with limited memory, storing the vectors to disk, because the Cholesky factorization algorithm only uses a subblock of the ERI matrix at any time, which can be computed as required.

5.1.2 Screening

It is common procedure to use techniques to ”screen” negligible ERIs and skip their computation entirely, usually by means of the Cauchy-Schwarz inequality[28]. This form of screening discards a significant amount of ERIs which would contribute to the final matrices below some predefined threshold. In direct SCF, the original inequality is usually extended to also take into account the magnitude of the elements in the density matrix subblocks. Other screening methods include distance-dependent estimates[97], in cases where distances in the molecule are considerable, but especially for screening contributions in canonical perturbation theory, which involve products of ERIs, and decay with distance becomes more relevant. These techniques on the other hand make the problem relatively sparse (irregular), and can no longer be split in equal-sized tasks with the same ease.

In some cases, the so-called differential density $\Delta D^{(n)} = D^{(n)} - D^{(n-1)}$ (the difference between density matrices of two consecutive iterations) is used instead of the full density matrix and the differential Fock operator obtained from it is added to previous operator. This approach, together with screening methods, has the advantage of further skipping many more ERI evaluations during the final cycles of the procedure, due to the presence of nearly-converged orbitals (typically core and other low-energy orbitals).

5.1.3 Convergence acceleration

There are several methods typically used in SCF calculations to accelerate the converge to the solution and decrease the number of iterations and overall computation. They also tend to stabilize the SCF procedure and increase the radius of convergence, which can otherwise be problematic in some systems. The immediate way to do it is to generate the best possible initial guess for the density matrix, typically based on a semiempirical method or some previous calculation.

The most widespread techniques used during the iterative procedure consist in extrapolating the density guesses for the next iteration. The Direct Inversion of Iterated Subspaces (DIIS)[98] keeps track of the last Fock and density matrices to generate a linear combination for the next iteration that minimizes the error residual. EIIS[99] is a similar approach that does the same for the energy, but its use is limited to a certain radius of convergence. Even with these techniques, convergence is not guaranteed. Shifting energy levels may sometimes be necessary to avoid non-converging oscillatory solutions, which usually occur when the LUMO and the HOMO are so close in energy that small perturbations change their ordering. The Optimal Damping Algorithm (ODA)[100] is a technique using implicit shifts in the Fock matrix generated by linear combinations of previous results, and can enforce absolute convergence to the solution.

Some other methods use approximations that reduce the computational complexity during the initial steps. A common family of approaches divide a large system into several subsystems or fragments, solve each one independently, and piece the solutions together[101] [102]. Another method suggested by Adamson and coworkers[103] is to replace the long-range $|1/r_{12}|$ operator with the short-range CASE operator $erfc(\omega|r_{12}|)/|r_{12}|$, an aproach that can be used for the first cycles of a SCF calculation, or even start with an initial ω (more local interaction) and decrease its value every cycle until $\omega = 0$, which recovers the original $|1/r_{12}|$ operator. The nuclear-electronic potential has to be modified accordingly, and computed every time the value of ω is modified. Both methodologies can reduce the number of ERIs necessary from $\mathcal{O}(N^2)$ to $\mathcal{O}(N)$ without substantially changing core orbitals, bonds, and other important chemical features, which tend to be localized.

A simple approach which I haven't found mentioned in the literature consists is using a smaller basis during the initial iterations, and add extra sets of shells every time the density matrix error norm of a given iteration converges below some threshold. This reduces not only the dimension of the algebra problem, but also skips the evaluation of the most time-consuming integrals until the last steps. This approach might be best suited for GC basis sets, which are augmented in a systematic way.

5.1.4 Advanced algorithms

Many implementations consider the Coulomb and Exchange problems independently, enabling a number of approaches that can accelerate their evaluation.

Coulomb matrix

The Coulomb matrix is a classical Coulomb problem, requiring the evaluation of the interactions between some charge densities and the potential generated by the total electron density of the molecule. The fast decay of the overlap between basis with the distance makes the Coulomb matrix sparse and block-diagonally-dominant. The different methods described next are not mutually exclusive.

While basis sets are designed to approximately span the subspace of molecular orbitals, the basis generated by their products are asymptotically degenerate, and in practice contain many near-linear dependencies contributing little to its flexibility to describe the electron density. Density-fitting methods[104] employ a smaller auxiliary basis set that is used to fit the total density or the basis products, reducing the total computation time. However, they also introduce new sources of numerical error, and new problems such as the choice of an adequate auxiliary basis for a particular problem and which is the appropriate fitting metric. Some approaches aimed at bounding the errors[105] use the ERI matrix' Cholesky vectors as fitting basis, or more commonly the Cholesky vectors from the atomic ERI matrices.

The J-engine[106] is another approach, based on the observation that many geometry-dependent recurrence relations are shell pair dependent only. In the J-engine, the $|ket\rangle$ basis products constituting the density are multiplied with their densities and expanded as Hermite Gaussian functions or contracted HGFs. The Coulomb matrix is computed by iterating over all the expanded $\langle bra|$'s for every ket, and applying the Hermite and horizontal recurrences to the contracted $\langle bra|$'s. The reduction of the number of pair transformations is from $\mathcal{O}(N^2)$ for all ERI batches to $O(N)$, a cost negligible in practice.

Linear-scaling methods are also used in SCF problems. The original Fast Multipole Method (FMM)[107] was designed for point particles, but it has been extended to treat densities in the Continuous Fast Multipole Method (CFMM)[108]. The KWIK method[109] uses a partition of the $1/r_{12}$ operator into a short-range operator containing the pole $erfc(\omega|r_{12}|)/|r_{12}|$ and a smooth, long-range operator $erf(\omega|r_{12}|)/|r_{12}|$. The short-range operator is evaluated in real space in $\mathcal{O}(N)$, and the long-range operator is evaluated in Fourier space.

The worst-performing cases for ERIs are usually due to slowly-decaying GTOs, which not only couple to other GTOs within a larger radius, but tend also to be of higher angular momentum. Primitives of GTOs with low Gaussian exponents delay the onset of performance gains in CFMM and other reduced-scaling techniques, because they impose larger cutoff distances before the problem can be approximated. The Fourier Transform Coulomb (FTC) method by Fusti-Molnar and Pulay[110] computes ERIs numerically in Fourier space, and can be used to evaluate integrals over diffuse GTOs efficiently, while using any other method(s) for the remaining ERIs.

Exchange matrix

The techniques described above are not applicable to the computation of the Exchange matrix, which is not expressable as a Coulomb problem. Nevertheless, and because according to Kohn's conjecture the density matrix $\rho(r, r')$ of an insulating system (system with non-vanishing HOMO-LUMO gap) exhibits an exponential decay with the distance $|r - r'|$, it is possible to combine this with pre-screening to construct $\mathcal{O}(N)$ methods for evaluating the Exchange contribution in such systems. These include the ONX[111] and LinK[112] algorithms. Some recent attempts have been directed towards reducing the prefactor. The Chain-of-Spheres algorithm[113] acomplishes this by mixing analytic and numerical integration, and pseudospectral methods[114] evaluate the integrals numerically on a spatial grid.

5.1.5 Parallelization

The cost of computing an ERI batch and contract it with the associated DM blocks can vary by orders of magnitude according to the angular momentums and contraction degrees (the elemental batch) of that batch. The number of batches of the same elemental batch can vary greatly depending fundamentally on the chemical system and basis set used, but also on the screening threshold and other details.

Such variability can lead in naïve parallel implementations to poor load balancing (asymetric computation effort between processors or nodes), translating into very inefficient use of the computational resources. The problem cannot in general be solved with static scheduling algorithms because integral pre-screening prunes the ERI batch lists on-the-fly and in an irregular manner. One possible solution is to use dynamic scheduling in a master-slave parallel architecture, where the master node dynamically computes lists of integrals to be evaluated and sends them to each node, but which introduces a sequential bottleneck in both the computation and the communication, which may not be noticeable for small clusters and/or problems, but can kill scalability in larger computations.

Modern processors are multicore, and multiple threads in the same node using shared memory for communication is much faster than inter-node communication, which makes a hybrid OpenMP+MPI (or similar) implementation desirable. The latest MPI standard has included shared memory features, but the new standard is still not fully supported, and like everything else in the standard, the actual mechanism is left to the implementation.

5.1.6 Cache hierarchy

Modern CPU cache hierarchies are the principal reason for the CPU performance, and are the key to high-performance codes. With cache miss latencies approaching 100 CPU cycles, the penalty of overlooking cache optimization in numerical software can in some regimes be many times higher than any improvement from using an algorithm with a better FLOP count. CUDA-capable GPUs also have their own slightly different cache hierarchy, but in many cases the programming model makes their details irrelevant.

5.1.7 Vector and stream processing

The vector unit of x86-64 CPUs has to be explicitely programmed, since efficient vectorization cannot in general be trusted to the compiler. To fully exploit the SIMD vector instructions available in modern processors and simultaneoulsy preserve tsome degree of data locality, it is necessary to use vector data structures properly aligned to the cache line's boundaries, and use the AoSoA (array of struct of arrays) construct to pack the data in a format such that the processor can direcly load the vector variables efficiently, as opposed to the typical AoS (array of structs) packing typical of the *C* family of languages or the SoA (struct of arrays) used in FORTRAN. A similar reasoning applies to warps in stream processing. This technique is somewhat analogous to matrix blocking, common in numerical linear algebra, whereby the matrices are sliced into small square blocks in order to simultaneously optimize data locality and vectorization.

To maximally exploit SIMD instructions and stream processors, the ERI batches "packed" in the same struct have to be chosen to be as similar as possible[115][116] (extrinsic vectorization), to be able to use the same control flow on the whole vector and perform the minimal unnecessary calculations. This is specially relevant for ERIs, where prescrening is also applied to the primitive Gaussian products, and GTO products of the same elemental basis pair can have very different effective contraction degrees depending on the distance between its centers. To make effective use of the primitive product prescreening with vectorization, all batches in the "pack" have to share reasonably similar contraction degrees in both pairs.

5.2 Design of the Echidna Fock Solver

The design of the *Echidna* (Exchange+Coulomb Hybrid Implementation of Direct Novel Algorithms) library is based in all previous considerations. The Echidna solver is an efficient, hybrid OpenMP+MPI parallel implementation of a Coulomb and Exchange matrix solver. Some of its core routines are also implemented for GPU, permitting hybrid CPU+GPU computations. It requires a molecule geometry and a basis set at initialization. The main routine admits a number of density matrices as input, which may be non-symmetric (for use in response theory) and outputs a set of Fock matrix contributions which may equal the Coulomb matrix only (for pure DFT functionals) or Coulomb plus any fraction of the Exchange contribution (for hybrid DFT functionals), including 1 (for Hartree-Fock).

5.2.1 Initialization

At initialization, an exhaustive list of all ERI 'kinds' required for the input molecule is generated. Here, the term 'kind' refers to the previously defined Elementary Basis Shell quadruplet and geometry of the atomic centers. A number of batch constants are precomputed and stored for each elemental basis quadruplet. A local memory pool is initialized per thread and per GPU stream, and an extra memory pool is initialized per node. Once the list with all 'kinds' is known, Echidna makes a call to initialize Quimera, the ERI evaluation library, and passes a list of the geometries and angular momenta of the ERI 'kinds' so Quimera can check that it can link to every routine needed, or load the necessary bytecode. A list of basis function products is generated, including their effective contraction degree and the Gaussian prefactors of their primitive products. The last step is to compute the Cauchy-Schwarz screening parameter for every product and store it for later use.

Echidna internally uses its own matrix storage format, with a recursive pattern that preserves memory locality for blocks of the same shell pair, GC function pair (if applicable), and atom pair. Every thread and CUDA stream keeps local output matrices (Coulomb and/or Exchange) where it performs its own partial updates. This circumvents the need for inter-thread synchronization to avoid race conditions between the threads, that could cause unnecessary contention.

When the Fock update routine is called, the input density matrices for the system are broadcasted to all nodes, and all thread-local output matrices are reset to zero.

5.2.2 The task scheduler

The list of ERI 'kinds' is sorted in dicreasing order of estimated computation weight and fed to each node's task scheduler, which in essence is a parallel (multithreaded) producers-consumers problem. Using a hybrid scheme is a big advantage over using MPI alone, because shared memory allows faster communication between threads, something that can be used to dynamically adjust to different workloads and different tasks. The threads switch between the task of adding (producing) Lists of Matrix Updates (LMUs) to a task pool and executing (consuming) the lists, depending on the filling level of the task pool. If GPUs are available and some CUDA stream is idle, the threads' priority is to make all asynchronous data copies necessary and fill the kernel queue of the stream with the matching asynchronous kernel launches. This prevents the GPU scheduler from starving once the other stream finishes, but returns control almost inmediately to the thread to continue working. GPU performance depends on its workload in a non-linear way, being notoriously poor for small tasks. To avoid any performance drop due to short LMUs outnumbering the computationally-intensive ones, the LMU pool is implemeted as a min-max priority queue, where the LMUs for the GPU streams are served from the front (max), and threads' LMUs are served from the back (min).

The principal philosophy followed during the parallelization is to avoid any form of synchronization between components as much as possible, making sure individual work threads can progress regardless of the global state of the computation. At the moment no task scheduling or any other synchronization is done across nodes. Synchronization introduces some overhead, and without knowing the specifics of the system (topology, bandwidth, latency, etc.) it's impossible to use a synchronization protocol that is generally efficient but that wouldn't potentially kill the scalability of the solver in some other cases. As a first approach it appears to be better to use a mechanism with which every node can generate its own list of integrals on its own, independently of other nodes. In the implementation the workload betweem nodes is balanced statistically through a simple mechanism: every node goes through the same 'kind' lists, but from every list only shell pairs or quadruplets with index congruent to the node's index (modulo the total number of nodes) are considered as candidates for evaluation. In the limit of large systems, all nodes evaluate LMU lists that are similar in number and composition to every other node's.

This approach can be easily extended in the future if it becomes necessary to support task migration between nodes. For reasons discussed next, it would be preferrable in terms of data-level parallelism to not shuffle the lists between nodes as it is currently done, and instead compute ranges of consecutive shell pairs within the same node, with very short LMUs being computed entirely in the same node. The communication is needed to avoid duplicated Matrix Updates in the system. An alternative strategy is to collect profiling data of the ERI evaluation (CPU time and number of processed batches for each ERI kind) in each node and broadcast it after every cycle. The collected data can be used to generate an educated guess of which tasks can be moved to another node to improve the balance of the next iteration.

5.2.3 Prescreening

The prescreening mechanism implemented is a modified Cauchy-Schwarz algorithm. Whether a given ERI and its related matrix updates is scheduled for execution or discarded depends on estimates of the integral batch and the density matrix sub-blocks which it is going to be contracted with. This prescreening can be partialy done in advance, discarding many updates below a predetermined threshold, but the integrals surviving the crude first-pass screening are tested again with a more accurate test.

The criteria to accept an ERI batch is that it contributes to the expected energy above some threshold $\kappa = 10^{-\tau}$, or equivalently, the batch is discarded if it has a negligible expected contribution to the final energy. Up to a constant factor, the condition is equivalent to simultaneously satisfying the following three inequalities:

$$\kappa > \Delta J^{ABCD} = \sum_{a,b}\sum_{c,d} D^{a,b}_{AB}\, D^{c,d}_{CD}\, (ab|cd) \tag{5.3}$$

$$\kappa > \Delta K_1^{ACBD} = \sum_{a,c}\sum_{b,d} D^{a,c}_{AC}\, D^{b,d}_{BD}\, (ab|cd) \tag{5.4}$$

$$\kappa > \Delta K_2^{ADBC} = \sum_{a,d}\sum_{b,c} D^{a,d}_{AD}\, D^{b,c}_{BC}\, (ab|cd) \tag{5.5}$$

where the summations indices span all elements of the ERI batch. These sums over the four indices can be reinterpreted as the inner product of two rank-4 tensors, wich can be "flattened" and recast into two rank-2 tensors (matrices). This allows the use of

and the CS inequality can be used to derive the proper upper bond. The norm of the external product of the matrix subblocks is simply the product of the norms. A new inequality appears for the Coulomb contractions:

$$\Delta J^{ABCD} \leq \|D_{AB} \otimes D_{CD}\|_F \; \|(ab|cd)\|_F \tag{5.6}$$

$$|D_{AB} \otimes D_{CD}\|_2 = \|D_{AB}\|_2 \; \|D_{CD}\|_2 \tag{5.7}$$

$$\|(ab|cd)\|_2 \leq \|(ab|cd)\|_F \tag{5.8}$$

and similar expressions appear for exchange. The Frobenius norm of the batch can be further manipulated using the typical CS inequality for ERIs:

$$\begin{aligned}
\|(ab|cd)\|_F &= \sqrt{\sum_{a,b}\sum_{c,d}(ab|cd)^2} \leq \sqrt{\sum_{a,b}\sum_{c,d}(ab|ab)(cd|cd)} = \\
&= \sqrt{\sum_{a,b}(ab|ab)}\sqrt{\sum_{c,d}(cd|cd)} = \\
&= tr\left((ab|ab)\right)^{1/2} tr\left((cd|cd)\right)^{1/2} = \\
&= \|(ab|ab)\|_*\|(cd|cd)\|_*
\end{aligned}$$

where $\|A\|_*$ is the so-called *nuclear norm* of the batch tensor. Finally, it is possible to express the criteria for accepting a batch as:

$$\mu_{AB} + \mu_{CD} + d_{AB} + d_{CD} \leq \tau \tag{5.9}$$

$$\mu_{AB} + \mu_{CD} + d_{AC} + d_{BD} \leq \tau \tag{5.10}$$

$$\mu_{AB} + \mu_{CD} + d_{AD} + d_{BC} \leq \tau \tag{5.11}$$

For J, K_1 and K_2, respectively, and with:

$$\mu_{IJ} = -\frac{1}{2}\log\left(\sum_{i,j}(ij|ij)\right) \tag{5.12}$$

$$d_{IJ} = -\log(\|D_{IJ}\|_F) \tag{5.13}$$

$$\tau = -\log\kappa \tag{5.14}$$

where μ_{IJ} is a new shell pair parameter -similar to the usual CS parameter- that is computed at the beginning of the calculation, and the d_{IJ} values are computed for every block at the beginning of each cycle.

A slightly better bound can be derived for the Coulomb interaction, interpreting the contraction as a bilinear form.

$$\Delta J^{ABCD} \leq \|D_{AB}\|_F D_{CD}\|_F \;\|(ab|cd)\|_2 \tag{5.15}$$
$$\tag{5.16}$$

where $\|A\|_2$ is the spectral norm of the matrix, equal to its largest singular value. An analogous manipulation to the previous derivation gives a result identical in everything except the value of μ_{IJ}:

$$\mu^C_{IJ} \;\; = \;\; -\frac{1}{2}\log\left(\lambda_{max}\left(IJ|IJ\right)\right) \tag{5.17}$$

where λ_{max} is the spectral norm (maximal eigenvalue) of the IJ shell pair's ERI matrix. it is also possible to derive bounds for Coulomb directly from the CS inequality as follows:

$$\upsilon_{AB} + \upsilon_{CD} \leq \kappa \tag{5.18}$$
$$\upsilon_{IJ} = -\frac{1}{2}\log\left(\sum_{ij}\sum_{kl} D^{ij}_{IJ} D^{kl}_{IJ}(ij|kl)\right) \tag{5.19}$$

which are lengthier to compute, but better than the two previous bounds. They can also be computed at a negligible cost by moving all contributions from equal pairs J^{ABAB} to the beginning of the computation and contracting every individual contribution with the density matrix subblock D_{AB}.

These approaches might be better suited than the general one if Exchange is being computed separately (or not needed in the first place); otherwise, the difference in the estimates tend to be small enough to prefer storing only one value of μ for every pair. The equivalent approaches for Exchange estimates results in a matrix with a "flattening" of the indices that is unsuitable for obtaining good CS estimates (their CS inequalities are trivially satisfied).

The prescreening presented here has two remarkable properties worth commenting:

- Rotational invariance: This is a missing property of both the traditional CS ERI screening and other methods, even though its significance has been recently pointed out by Maurer, Lambrecht et al. in the paper[97]. Using rotationally-invariant screening is necessary to obtain consistent results independent of the chosen orientation of the molecule.

- Symmetry: This property is also absent in the typical CS screening and is relevant for two reasons, one conceptual and one practical. Conceptually, because the interaction between two subsystems in the molecule is always symmetric, so the decision of including certain batches for its computation should also be independent from which subsystem's perspective the method is applied, i.e. if density P interacts strong enough with density Q it stands to reason that the converse is automatically true. The practical reason stems from computational savings. Coulomb, Exchange and density matrices and ERI batches obey certain permutational symmetries, which are implicitly exploited throughout the evaluation of matrix elements, cutting down the total amount of work. In particular, ERI batches can have up to 8 other batches equivalent by symmetry, and can be contracted with up to 6 different density blocks (2 for Coulomb and 4 for Exchange). This stesses the importance of using a consistent method that will simultaneously either accept or reject the contraction with all the density blocks of a certain type. Otherwise, the computation of many spurious ERI batches needed only for digesting a few elements of one density block can add up to a significant increase in computation.

Efficiently screening the Coulomb interactions with this method is rather simple. One generates the lists $\mu_{AB} + d_{AB}$ and $\mu_{CD} + d_{CD}$ (or equivalent) and sorts them by increasing value, which can be efficiently done in $\mathcal{O}(NlogN)$. The last interacting CD pair of the list given an AB pair can be retrieved in $\mathcal{O}(\log N)$ with a simple binary search of the list for the value $\tau - \mu_{AB} - d_{AB}$, and viceversa.

Screening of Exchange can be performed in $\mathcal{O}(N)$, assuming the density matrix $\rho(r, r')$ decays fast with the distance $|r - r'|$. Of the several alternatives, perhaps the simplest is iterating over all AB pairs of the list, constructing four $\mathcal{O}(1)$ lists of all the shells C and D with $d_{AC} < \tau - \mu_{AB} - min(\mu_{CD})$ and $d_{BD} < \tau - \mu_{AB} - min(\mu_{CD})$ and the analogous lists for d_{AD} and d_{BC}. The pairs of lists are iterated over C and D, and the CD pairs are checked with the K_1 and K_2 inequalities. Notice that the locality of exchange is manifested through a large overlap in the results of the K_1/K_2 inequality tests. This approach is in essence equivalent to the ONX algorithm, only exploiting the advantages of the EBA and the geometric simplifications.

5.2.4 Integral packing

Once the list of acceptable shell quadruplets is generated (along with the type of contractions it is involved into), it is divided in tiles according to the CPU cache line size or the GPU warp size, and the tiles are packed in LMUs. The quadruplets are grouped according to their similitude (effective contraction degree of its constituents) as much as possible, and the groups are sorted by increasing distance between the shell products. A special algorithm, the EVPA2, handles the packing of the remainders of every group.

There are different factors to consider when deciding the limit of tiles per LMU, but in general it is better to generate many short lists instead of a larger one, especially for integral 'kinds' that are expected to appear towards the end of the calculation, because the small overhead due to extra function calls and memory management is compensated by the better load-balance between threads as a consequence of the finer-grained parallelism. The opposite is true for LMUs that are to be computed on the GPU. When the calculation is done on a GPU, the performance of the CPU numerical code is secondary to obtaining high occupation on GPU. The threads' part in consuming the shortest LMUs before the GPU streams run out of compute-intensive tasks is also vital to high combined performance, but due to the low computational load of the LMUs, this is accomplished automatically without requiring extra attention.

The effectiveness in screening and packing by similitude comes at the cost of loss of the locality in the chemical system, since statistically, the more "similar" integral batches will most likely be in different parts of the molecule. This also imposes some design constraints to parallelism, since it is desirable to compute the similar batches together and not scatter them in different threads, streams or nodes. Of course these issues with task granularity tend to diminish as the input system becomes larger and exposes more potential data-level parallelism.

5.2.5 EVPA2

The integrals are grouped in tiles of 8, 16 or 32, so after removing all the tiles exactly filled, it is likely that a considerable number of integrals will remain, with contraction degree combinations scattered throughout the range of admissible values. Moreover, this will be the case for the major part of the integrals of the class in smaller systems and/or for some special combinations of values with low populations.

Packing every remaining group and generating incomplete tiles would be a waste of computational resources, given that the tiles are evaluated as a whole regardless of the number of useful elements they contain. The solution is to promote some integrals to a higher precision (higher contraction degree) to fill the tiles. The problem is that the contraction step (L step) scales as $K_{AB}^{eff} K_{CD}^{eff}$, and while it might be the only option in some situations, the mechanism of promotion should reuse existing tiles and avoid creating new tiles of higher K^{eff}. Preferably, the repacking should minimize the cost of the contraction, to the extent of the possibilities.

The *Efficient Vector Packing Algorithm 2* (EVPA2 for short) is an algorithm designed to solve this problem. The EVPA2 algorithm starts with individual integrals and groups them recursively into $2^m - tiles$ of different sizes. Then it applies a recursive merging strategy, which terminates with a number of $2^n - tiles$. In the algorithm's pseudocode, $2^m - tiles$ have also a *group*, determined by the pair $(K^{eff}_{AB}, K^{eff}_{CD})$. A tile is *unpromotable* if there are no tiles of a higher group in the table, meaning with both K^{eff} equal or higher. The algorithm proceeds as follows:

```
for m=2^n,...,1
    for all groups G in table
        if integrals in G >= m
            I  <- m integrals of G
            t  <- m-tile of I
            G  <- G\I
            Tm <- t in group G
        end if
    end loop
end loop

for m=1,...,2^n

  while not stalled
    U <- unpromotable m-tiles of Tm
    N <- unpromotable m-tiles of Tm\U
    n <- find m-tile in N with highest Kab*Kcd
    P <- m-tile parents of n
    p <- find m-tile in P with lowest Kab*Kcd
    s <- 2m-tile (p,n)
    T2m <- s in group(p)
    Tm  <- (Tm\p)\u
  end loop

  while size(Tm) > 1
    n,m <- find m-tiles of Tm with lowest |Kab-Kab'| + |Kcd+Kcd'|
    s   <- 2m-tile (n,m)
```

```
      T2m <- s in group( max(Kab,Kab'),max(Kcd,Kcd' )
      Tm  <- (Tm\n)\m
    end loop

    if size(Tm') > 0
      n <- only m-tile left
      s <- 2m-tile (n,0)
      table <- s in group(n)
    end if

  end loop
```

5.2.6 Geometry and rotations

When the LMU is processed, the first step consists in finding the four (or less) centers of the shells and computing rotation matrix between frames of reference. The vectors $(\vec{AB}, \vec{CD}, \vec{AC})$ describing the geometry of the integral are rotated accordingly and stored vector-packed in an array for use next. The matrices for rotating spherical harmonics are also packed and stored for the digestion step. The CPU code uses some precomputed rotations along the shell pairs to speed up the process. In GPUs, the bandwidth and memory consumed by storing make the approach slower than regenerating the matrices when needed.

5.2.7 OCL steps

In the COLD PRISM nomenclature[94], the O-, C-, and L- steps refer to the generation of the one center integrals $[m]$, the contraction of kernels (CDR/K4 step), and the transformation to ERI batches (MIRROR step). These are handled entirely by the Quimera library, and a more detailed explanation can be found in the corresponding chapter.

5.2.8 D step

The last step of the path is the D- step, also known as digestion or contraction with the density matrix. The CDR/K4+MIRROR algorithm of the previous step leaves the ERI batches in the rotated frame of reference. Instead of rotating the whole batch in $\mathcal{O}(L^5)$ before digestion, Echidna rotates the density blocks to the frame of reference of the ERI, contracts, and rotates the Fock block update to the molecular frame.

The routines in this step, both the CPU and the GPU versions, are metaprogrammed with regular templates, which generate routines for all combinations of angular momenta needed. There are different contraction routines for the Coulomb-like and Exchange-like contractions, and depending on the geometry of the centers. The CPU code just adds the contributions to the locations in the Coulomb and Exchange matrices where they belong. The GPU version demands a little more commentary.

The digestion step is an example of reduction; there is a considerable amount of generated data being used to update only a few memory positions. In cases whith only few threads running in parallel, either locking the memory or using atomic operations tends not to be problematic performance-wise, because collisions are rare. However, in GPUs there might be many warps of threads executing. Using a scatter to update the matrices' blocks is in this case slow not only because of the uncoalesced memory accesses, but also because atomic operations are needed to avoid race conditions. The solution is to use a scatter-to-gather transform: instead of mapping the warp's threads to iterate over the list of matrix updates, it is possible to use them to map the iteration over the positions of the matrix of destination. This supposes a complication in the design due to sorting, transposing and the remapping of threads, but the final update can be performed atomic-free.

Chapter 6

Evaluation of QM/CMM matrix elements

The QM/CMM method iteratively solves for the charge distributions and dipoles in the CMM region, and the electron density in the QM region until convergence. The interaction between the CMM metallic atoms and the electrons requires evaluating the corresponding 1-electron molecular integrals over two GTO shells and the potential generated by possibly many diffuse (Gaussian) charges and dipoles. These integrals are essentially a Coulomb problem, not much different from the evaluation of electron-nuclear and 2-electron interactions. The potential generated by the metallic part of the CMM region is:

$$V(\mathbf{r}) = \sum_m^M q_m^{ind} \frac{erf(|\mathbf{r} - \mathbf{C}_m|/R_q)}{|\mathbf{r} - \mathbf{C}_m|} + \sum_m^M \mathbf{p}_m^{ind} \cdot \frac{(\mathbf{r} - \mathbf{C}_m)}{|\mathbf{r} - \mathbf{C}_m|^2} \times \left[\frac{erf(|\mathbf{r} - \mathbf{C}_m|/R_q)}{|\mathbf{r} - \mathbf{C}_m|} - \frac{2}{\sqrt{\pi} R_p} exp(-|\mathbf{r} - \mathbf{C}_m|^2/R_p^2)\right] \tag{6.1}$$

where q_m^{ind} and $\mathbf{p}_m^{ind}$ are the partial charges and dipoles of the m-th atom. Note that the typical Gaussian radiuses R_q and R_p may be different for charge and dipole contributions.

Other than the use of Gaussian charge distributions in lieu of point charges, there are two important factors to consider in the design of an efficient solution. First, the nanoparticles or metallic surfaces (the metallic part of the CMM region) may contain thousands of atoms; many times more the atoms of a typical QM region. Second, the charge distribution in the CMM region is being solved simultaneously with the QM part in a SCF-fashion, and updated at every cycle until both subsystems converge. Obtaining an overall good efficiency is therefore more critical than for the nuclear potential matrix, which is computed once at startup and stored for all later uses.

The matrix elements can be expressed as a reduction over (at most) three-center intregrals involving two GTOs and one atomic charge/dipole distribution. Carrying this naïve reduction leads to $\mathcal{O}(MN^2)$ scaling, where M is the number of atoms in the CMM region and N is the number of basis functions in the QM region. The first important factor to consider in the design is the typical input of the module. For large N (extended QM region) it is expected that GTO products will decay fast with increasing distance, so it is possible to implement some variant of the typical GTO product screening and reduce the complexity to $\mathcal{O}(MN)$, as will be discussed in detail later. Second, a heavy-load input will typically consist of a much larger number of atoms in the CMM region than in the QM region ($M >> N$) or else it would be preferable to use a fully-QM treatment.

Several techniques reduce the computational complexity of Coulomb problems from quadratic to loglinear or better for large number of particles. These include Ewald summation techniques and multipole expansions like FMM and related algorithms. However these algorithms are cucumbersome to implement, require careful tuning of several parameters that control the order of the approximate expansions (determining the error bounds) for the application and, more importantly, do not actually outperform the $\mathcal{O}(N^2)$ direct summation of pair interactions for the systems we are interested in computing. The specifics can of course vary depending on the hardware and the implementation, but the asymptotic prefactors of $\mathcal{O}(N)$ Coulomb methods are large enough to make them an inferior choice for systems below a few thousand particles. Another reason to avoid them is that other steps of the method have at the moment worse complexity than $\mathcal{O}(N)$ because they do not exploit the possible sparse structure of the so-called *relay matrix* for larger systems.

6.1 Evaluation of integrals

Equation 6.1 can be rewritten in a more compact form as follows:

$$V(\mathbf{r}) = \sum_{m}^{M} q_m U^{(0)}(\mathbf{r}, \mathbf{C}_m, R_p) + \sum_{m}^{M} \mathbf{p}_m \cdot \mathbf{U}^{(1)}(\mathbf{r}, \mathbf{C}_m, R_q) \tag{6.2}$$

where the individual contributions to the potential are given by:

$$
\begin{aligned}
U^{(0)}(\mathbf{r},\mathbf{C},R_q) &= \frac{erf(|\mathbf{r}-\mathbf{C}|/R_q)}{|\mathbf{r}-\mathbf{C}|} = \frac{2}{\sqrt{\pi}}\int_0^{1/R_q} exp(-s^2|\mathbf{r}-\mathbf{C}|^2)ds \qquad (6.3)\\
\mathbf{U}^{(1)}(\mathbf{r},\mathbf{C},R_p) &= \frac{(\mathbf{r}-\mathbf{C})}{|\mathbf{r}-\mathbf{C}|^3}(erf(|\mathbf{r}-\mathbf{C}|/R_p) - \frac{2}{\sqrt{\pi}}(|\mathbf{r}-\mathbf{C}|/R_p)e^{-|\mathbf{r}-\mathbf{C}|^2/R_p^2}\\
&= \nabla_{\mathbf{C}}U^{(0)}(\mathbf{r},\mathbf{C},R_p) \qquad (6.4)
\end{aligned}
$$

a useful simplification for the second term comes from realizing it can be expressed as a derivative of the first.

The potential terms of the first kind are due to the contributions we refer to as "Gaussian charges", and the terms of the second kind are due to what we will refer to as "Gaussian dipoles". Except for their partial charge or dipole and possibly different radii (if there is more than one metallic element in the CMM region), they all correspond to the same basic Gaussian charge distribution and its spacial derivatives. Consider a (normalized) Gaussian distribution of unit charge, with center in **C** and characteristic radius R, and a corresponding distribution(s) of unit dipole moment:

$$
\begin{aligned}
\sigma_s(\mathbf{r},\mathbf{C},R) &= (\pi^{-3/2}R^{-3})e^{-|\mathbf{r}-\mathbf{C}|^2/R^2} \qquad (6.5)\\
\sigma_p(\mathbf{r},\mathbf{C},R) &= \nabla_{\mathbf{C}}\sigma_s(\mathbf{r},\mathbf{C},R) \qquad (6.6)
\end{aligned}
$$

The charge density of the CMM region is generated by the superposition of the densities of the individual atoms:

$$
\sigma(\mathbf{r}) = \sum_m^M (q_m\sigma_s(\mathbf{r},\mathbf{C_m},R_m) + \mathbf{p}_m\cdot\sigma_p(\mathbf{r},\mathbf{C_m},R'_m)) \qquad (6.7)
$$

The QM/CMM matrix elements can be expressed with the shorthand notation used for 2-electron integrals:

$$
(\mathbf{ab}|\sigma) = \int_{V_1}\int_{V_2}\phi_{\mathbf{a}}(\mathbf{r}_1,\mathbf{A})\phi_{\mathbf{b}}(\mathbf{r}_1,\mathbf{B})|\mathbf{r}_1-\mathbf{r}_2|^{-1}\sigma(\mathbf{r}_2)d\mathbf{r}_1d\mathbf{r}_2 \qquad (6.8)
$$

where the contracted $\langle bra|$ refers as usual to the summation over its primitive products and the contracted $|ket\rangle$ implies of course summation over all CMM Gaussian charges and dipoles:

$$
\begin{aligned}
(\mathbf{ab}| &= \sum_{i,j}^{K_i,K_j} D_i D_j e^{-\frac{\alpha_i \beta_j}{\eta_{ij}} \mathbf{AB}^2} [\mathbf{a}_i \mathbf{b}_j| & (6.9)\\
|\sigma) &= \sum_{m}^{M} (q_m |\sigma_m] + \mathbf{p}_m \cdot |\pi_m]) & (6.10)
\end{aligned}
$$

Most techniques applicable to ERIs can also be used for QM/CMM integrals, which can be considered as the 3-center subcase of the former, with the caveat that the ket contraction can imply arbitrary centers. In particular, the usual McMurchie-Davidson approach holds for the $\langle bra|$. Expanding the product of cartesian polynomials around the centers $\mathbf{A}$ and $\mathbf{B}$ in Hermite polynomials around the centers $\mathbf{P}$ lying on the segment connecting $\mathbf{A}$ and $\mathbf{B}$, and integrating the product of the potential with the one center primitive function products, after some manipulation, we find we need to solve integrals of the type:

$$
[\overline{\mathbf{p}}|\sigma] = (\partial_{P_x})^{p_x} (\partial_{P_y})^{p_y} (\partial_{P_z})^{p_z} I(\mathbf{PC}, \eta, R) \tag{6.11}
$$

$$
I\left(\mathbf{PC}, \eta, R\right) = \frac{2\pi}{\eta^{3/2}} \int_0^{(1+\eta R^2)^{-1/2}} e^{-\eta |\mathbf{PC}|^2 t^2} dt \tag{6.12}
$$

where $\eta = \alpha + \beta$ is the sum of the primitive exponents and $\mathbf{P} = (\alpha \mathbf{A} + \beta \mathbf{B})/\eta$ is the center of the Hermite function expansion. The integrals over Gaussian dipoles can be evaluated straightforward using the previously mentioned gradient relation:

$$
\begin{aligned}
[\overline{\mathbf{p}}|\pi_x] &= [\overline{\mathbf{p}}|\partial_{C_x} \sigma')] = \\
&= (\partial_{C_x})(\partial_{P_x})^{p_x} (\partial_{P_y})^{p_y} (\partial_{P_z})^{p_z} I(\mathbf{PC}, \eta, R) = \\
&= -(\partial_{P_x})^{p_x+1} (\partial_{P_y})^{p_y} (\partial_{P_z})^{p_z} I(\mathbf{PC}, \eta, R) = \\
&= -[\overline{\mathbf{p} + \mathbf{1}_x}|\sigma'] & (6.13)
\end{aligned}
$$

and equivalent expressions for the y and z components. This simplifies every integral to a modified 1-center integral, which is the reason why accumulation over the ket can be directly performed at this stage, using the following expression:

$$
[\overline{\mathbf{p}}|\sigma) = \sum_{m}^{M} \left(q_m [\overline{\mathbf{p}}|\sigma_m] - \sum_{\lambda = x,y,z} \mathbf{p}_{m,\lambda} [\overline{\mathbf{p} + \mathbf{1}_\lambda}|\sigma'_m)] \right) \tag{6.14}
$$

The well known 1-center McMurchie-Davidson recurrence relations[80] can be applied to simplify the result to:

$$[\overline{\mathbf{p}+\mathbf{1}_\lambda}|\sigma]^{(m)} = \mathbf{PC}_\lambda[\overline{\mathbf{p}}|\sigma]^{(m+1)} - p_\lambda[\overline{\mathbf{p}-\mathbf{1}_\lambda}|\sigma]^{(m+1)} \tag{6.15}$$

Until reduction to the fundamental auxilliary integrals.

$$[\overline{\mathbf{0}}|\sigma]^{(m)} = \frac{2^{1/2}\pi}{\eta^{3/2}}\left(\frac{2\eta}{1+\eta R^2}\right)^{m+1/2} F_m\left(\frac{\eta}{1+\eta R^2}|\mathbf{PC}|^2\right) \tag{6.16}$$

where $F_m(z)$ is the standard Boys' function[117].

6.2 Algorithm

The algorithm resulting from applying the formulas presented in the previous section has a computational complexity of $\mathcal{O}(MK^2L^4)$. It can be extended straightforwardly to treat general contraction basis sets at a negligibe $\mathcal{O}(K^2J^2)$ extra cost.

```
loop over GTO primitives of centers a and b
  check primitive pair screening
  loop over CMM centers m
    compute [0] integrals
    apply 1-center MMD recurrence relation
    contract [p] integrals
  end loop
  transform hermite to cartesian
  loop over GC functions
    contract primitive pair brackets
  end loop
end loop
loop over GC functions
  transform GTOs from cartesian to spherical
end loop
```

This design is practically identical to the J-engine algorithm[106] used for Coulomb. Because the total angular momentum L tends to remain lower than for two-electron integrals, the arguably bad (quartic) scaling with the angular momentum is not much of an issue. It is possible nevertheless to design an algorithm with a better bound using techniques that parallel the early-contraction schemes for 2-electron integrals. Contrary to 2-electron integrals, however, the CMM distributions do not necessarily follow any particular pattern that can be used to reduce computation. This implies the only alternative left to design an early contraction method is to accumulate over the CMM charges first. The algorithm is outlined next:

```
loop over CMM centers m
  loop over GTO primitives of shell b
    loop over GTO primitives of shell a
      check primitive pair screening
      compute [m] integrals
      loop over GC functions of shell a
        contract v{m] integrals
      end loop
    end loop
    loop over GC functions of shell b
      loop over GC functions of shell a
        contract uv(m] integrals
      end loop
    end loop
  end loop
  loop over GC functions of shell b
    loop over GC functions of shell a
      apply bra-contracted 1-center MMD RR to generate uv(r]
      contract ket with CMM charge/dipole
    end loop
  end loop
end loop
loop over GC functions of shell b
  loop over GC functions of shell a
    apply bra-contracted H2C transformation to generate (e0|s)
    apply HRR to generate (ab|s]
    transform GTOs from cartesian to spherical
  end loop
end loop
```

The resulting algorithm has complexity $\mathcal{O}(MK^2L^2J)+O(MKL^3J^2)+O(ML^4J^2)$, but it is more a curiosity than an actual practical method for the present discussion, because in practice the more convoluted structure of the innermost loop depending explicitely on J incurs in a large penalization for low-L. Moreover, the prescreening over GTO pair primitives has to be done close to the innermost loop, complicating an otherwise regular loop structure as well as the paralellization scheme described next.

6.3 Implementation

The QM/CMM integral module makes full use of the general implementation techniques previously described. The GTO pairs are classified according to their elemental basis pair tag. The pairs within every group are sorted by distance (or equivalently, by decreasing CS parameter), with all pairs under the predefined CS threshold being discarded. Consecutive GTO pairs within every group are packed in tiles according to cache line size, which are sent to the evaluation routine. Due to its comparative simplicity and the limited number of combinations of angular momentum, the generation of specialized code can be accomplished using regular templates.

The module also implements a simple hybrid parallelization scheme: the pair lists are split evenly between all nodes to try to maintain a balanced load without resorting to more complex schemes involving inter-node synchronization which could potentially kill scalability. Within a node, the lists are further divided amongst processors in a multithreaded producers-consumers loop. For very unbalanced loads it is possible to generate a larger amount of smaller work tiles by splitting the CMM lists. This incurs in some computational penalization due to a partial duplication of work in the bra transformations and ket-contractions, but this is in general affordable for large M.

A screening over the primitive pairs is done in the evaluation using a modified CS scheme, but the values are precomputed and only need a relatively simple check within the function.

6.4 Screening at the GTO-product level

Two levels of integral screening are employed in the module. The simplest one is based in the fast decay of the GTO products with distance. This can inmediately reduce the overall computational scaling from $\mathcal{O}(MN^2)$ to $\mathcal{O}(MN)$ for a negligible cost, and enables savings even for very small QM regions.

The Cauchy-Schwarz inequality provides an easy and robust way of implementing such screening. Consider the CS inequality for two charge densities $\rho(r)$ and $\sigma(r)$ interacting through a Coulomb potential:

$$\left|(\rho(r_1)|r_{12}^{-1}|\sigma(r_2))\right| \;\leq\; (\rho(r_1)|r_{12}^{-1}|\rho(r_2))^{1/2}\;(\sigma(r_1)|r_{12}^{-1}|\sigma(r_2))^{1/2} \tag{6.17}$$

This form of CS is useless for nuclei-electron potential integrals because point charges make one of the rhs integrals diverge to infinity and the inequality holds trivially. However, it is very useful for screening two-electron integrals, where it usually gives upper bounds that are not far from the true values. Because QM/CMM integrals only involve distributed charge densities, the CS inequality is an attractive starting point for screening. Moreover, the CS factors corresponding to the GTO products are usually readily available, since they are computed and stored for use in ERI screening. The CS factor of the CMM density requires in principle computing the full electrostatic self-interaction potential of the CMM region. This is expressed (omitting dipole contributions for brevity) as:

$$(\sigma(r_1)|r_{12}^{-1}|\sigma(r_2)) = \sum_i^M \sum_j^M q_i q_j \;(\phi_s(\mathbf{r}_1, \mathbf{C}_i, R_i)|r_{12}^{-1}|\phi_s(\mathbf{r}_2, \mathbf{C}_j, R_j)) \tag{6.18}$$

The interaction potential of a superposition of Gaussian densities can be easily evaluated with a subset of the techniques needed for the GTO products, already discussed in chapter 3. However, computing all pair-interactions of the CMM region with a $\mathcal{O}(M^2)$ method seems unnecessarily wasteful (and a potential future bottleneck) when the interest is exclusively in obtaining a useful approximate upper bound for a given matrix entry. Using linear-scaling techniques for such a small issue looks like overkill. Fortunately, it is possible to produce an upper bound to the self-interaction in $\mathcal{O}(M)$ time at the cost of increasing the bound only slightly.

The previously introduced CS inequality holds because the Coulomb operator is hermitian, and it is possible to construct a normed inner product space using the densities as the elements[28]. The more general, algebraic form of the CS inequality can be written:

$$\langle a|b\rangle \leq \|a\|\cdot\, \|b\| \tag{6.19}$$

When the element can be ifself decomposed as a sum, the triangle inequality provides an upper bound directly from the norms of the terms in the sum.

$$\|a+b+c+\dots\| \le \|a\|+\|b\|+\|c\|+\dots \tag{6.20}$$

The CS parameter of the CMM region is estimated in such way in the module, by simply adding the CS parameters of the individual CMM atomic densities. The individual CS parameters involve a simple 1-center integral, and are computed trivially. The solutions (in atomic units) are:

$$\kappa_s(R) = \sqrt{\int_V \phi_s^2(\mathbf{r},0,R)d\mathbf{r}} = \left(\frac{2}{\pi}\right)^{1/4} R^{-1/2} \tag{6.21}$$

$$\kappa_p(R) = \sqrt{\int_V \phi_p^2(\mathbf{r},0,R)d\mathbf{r}} = \left(\frac{8}{9\pi}\right)^{1/4} R^{-3/2} \tag{6.22}$$

These terms are multiplied by the norm of the partial atomic charges or dipole vectors, respectively:

$$\kappa_{CMM} = \sum_m^M |q_m|\kappa_s(R_m) + \sum_m^M \|\mathbf{p}_m\|\kappa_p(R'_m) \tag{6.23}$$

and the final screening formula is derived:

$$\mu_{AB} + d_{AB} + \mu_{CMM} \le \tau \tag{6.24}$$

with terms defined in section 5.2.3 and $\mu_{CMM} = -\log(\kappa_{CMM})$.

6.5 Screening of GTO product primitives

A more advanced screening can be performed at the primitive product level. Using amplitude thresholds, which are common in nuclear integrals, is in this case arbitrary at best. A more robust methodology based again on the Cauchy-Schwarz inequality has been developed. The general CS screening previously introduced makes use of the properties of matrix norms to generate a rotationally-invariant method based on the trace of the 2-electron tensor block corresponding to the $(ab|ab)$ integral batch. It can be generalized to treat primitive pair screening, as described next. In practice, this screening and the previous GTO screening are integrated in the same method.

The primitive pairs are sorted as usual in order of increasing total exponent. The primitives of the pair are included to a list one by one by decreasing order of Gaussian exponent, and for each step, the trace CS μ_{AB}^K of the ERI for the incomplete primitive list is computed and stored. The last primitive product added completes the list, and the trace CS parameter μ_{AB} is recovered. The screening procedure relies in discarding as many sub-threshold pairs as possible, while keeping the error bounded:

$$\|\rho_{AB}\|_F \sqrt{\sum_{a,b} (\Delta^K \mathbf{ab}|\rho)^2} \leq \|\rho_{AB}\|_F \kappa_{CMM} \left(\sum_{a,b} (\Delta^K \mathbf{ab}|\Delta^K \mathbf{ab}) \right)^{1/2} \leq \kappa \quad (6.25)$$

Where Δ^K means that only the last K primitive products of the shell pair are considered. During integral evaluation, the list of primitive pairs to discard is chosen to be the maximal K list such that the inequality:

$$\mu_{AB}^K + d_{AB} + \mu_{CMM} \leq \tau \quad (6.26)$$

$$\mu_{AB}^K = -\frac{1}{2} \log \left(\sum_{a,b} \sum_{i,j}^{\in K} \sum_{k,l}^{\in K} [\mathbf{a}_i \mathbf{b}_j | \mathbf{a}_k \mathbf{b}_l] \right) \quad (6.27)$$

holds. This screening methodology requires in principle $\mathcal{O}(K^2)$ evaluations of the trace; if the whole batch of 2-electron integrals is computed, even with the CDRK4+MIRROR method the computational cost can scale quite badly as $\mathcal{O}(K^6 L)+ O(K^2 L^8)$, and become an uexpected bottleneck od the program. Fortunately a number of simplifications can be performed. First, only needing the trace of the 2-electron batch reduces substantially the number of operations. Second, using the $(AB|AB)$ center degeneracy of the integrals exposes only the distance between centers $|AB|$ as the only dependence with the geometry. Third, the accumulation loops can be modified to yield directly every partial contribution, reducing the dependence with the contraction degree to $\mathcal{O}(K^4)$. The details of the method are nevertheless lengthy and of minor importance to this discussion given that these integrals are computed only once. The final cost is in any case less than that of the evaluation of a regular general 2-electron integral.

Chapter 7

Evaluation of ECP matrix elements

Relativistic effects play an important role in the chemistry of heavy elements. The correct description of such effects require the use of scalar relativistic corrections, or approximations to the many-electron relativistic Hamiltonian operator, such as the Dirac-Coulomb-Breit Hamiltonian. However, such effects are mainly concentrated to the inner shells, where electrons have a higher kinetic energy, and play a much smaller role in the outer valence shells. The differences in the core configuration, however, indirectly affect the valence shells. Effective core potentials (ECPs), also refered to as pseudopotentials (PP) are a popular way to incorporate relativistic effects without the need of relativistic Hamiltonians. Similar constructions are used in calculations with plane-wave basis sets to reduce the number of functions needed. ECPs replace the $\hat{V} = Z/|r|$ potential operator of one nucleus and the core electrons associated with that atom with the following 1-electron potential:

$$\hat{V}^{PS} = \hat{V}_L + \hat{V}_{SL} \tag{7.1}$$

The first term is the so-called local operator, which is long-ranged and is defined as a function of the radius only, often expanded as a sum of optimized Gaussians multiplied with a monic polynomial of the radius.

$$\hat{V}_L = V_L(r) = \sum_n \sum_k c_{n,k} |r|^n e^{-\alpha_{n,k} r^2} \tag{7.2}$$

The resulting 1-electron integrals are also known as type-I ECP integrals, and can be evaluated generally with little complication with use of the Gaussian product theorem. Cases involving odd powers of l are not common, so the $|r|^l$ part usually admits exact expansion as polynomials of its cartesian variables, as $|r|^l = (x^2 + y^2 + z^2)^{l/2}$. For odd cases, one can make the substitution

$$|r - C|^l = 2\pi^{-1/2}|r - C|^{l+1} \int_0^\infty exp(-s^2|r - C|^2)ds \tag{7.3}$$

compute the rest of the integral over the cartesian coordinates analytically, and integrate over s as the last step, in a procedure that mirrors the evaluation of nucleus-electron potential integrals.

The second term is the so-called the semilocal operator, and is significantly more complex to evaluate:

$$\hat{V}_{SL} = \sum_{l=0}^{\infty} V_l(r) \sum_{m=-l}^{l} |lm\rangle\langle lm| \tag{7.4}$$

with $V_l(r)$ expanded similarly as the local operator. The resulting 1-electron integrals are known as type-II ECP integrals. The operator involves radial functions centered in some atom, and an operator projecting out the angular momentum of the left and right functions, as "seen" from the pseudopotential center. The treatment of the projection operator is far from trivial for plane waves and for GTOs when their center does not coincide with that of the pseudopotential.

In calculations with plane waves, it is usual to bypass their computation by using a separable representation of the radial function due to Kleinman and Bylander[118], which allows to compute the integrals as the product of the overlap of the basis functions with the functions that make the factorization possible. Due to the incompleteness of the basis used for the separation, this approach can sometimes lead to the appearence of "ghost" orbitals showing an incorrect number of nodes.

An integration scheme for GTO functions suitable for all cases has not yet been presented in the literature[119]. All approaches to solve type-II ECP integrals so far have been focused in carrying the integrations over the angular momentum analytically, and solving the radial integral in a variety of ways. McMurchie and Davidson[120] attempted three different methods for the radial integration, including Gauss-Hermite quadrature, and expanding one or both Bessel functions with Taylor series and carrying the radial integration analytically. The last two attempts have the inconveniency of requiring an undefined number of terms to achieve numerical convergence, which is a function of the distances and Gaussian exponents.

More modern attempts, like[121], are usually based on the quadrature idea, which have the computational advantage of being able to pretabulate the radial function for each ECP-GTO pair (at the cost of $\mathcal{O}(N)$ memory, with N the number of quadrature points), but the inconveniency that it is impossible to predefine a grid with the resolution necessary to compute all integrals which is again related to the wide range of parameters that can possibly enter the integral. This requires the use of adaptative quadrature methods, which iteratively refine the integration grid until the estimate of the remaining numerical error falls below some predefined value.

It is clear from a computational perspective that a general analytic solution is desirable. One such solution is presented for the first time in the next section. An efficient contraction algorithm based on this solution is described in the section following.

7.1 Evaluation of type-II integrals

We will assume the use of spherical GTOs throughout the section. It is possible to extend the analysis to cartesian GTOs, but we are trying to get rid of bad ideas, not encourage their use. The solid harmonics are assumed to be complex for simplicity of the formulation. The expressions for real solid harmonics can be easily derived from the present ones.

As with any molecular integral, type-II integrals over GTOs can be expanded in terms of their primivites. The primitive integrals are expressed with the following formula:

$$[\chi_l^m|\hat{V}_\lambda|\psi_{l'}^{m'}] = \sum_{\mu=-\lambda}^{\lambda} [\chi_l^m(\mathbf{A})|\lambda\mu, \mathbf{C}\rangle r^{n_\lambda} e^{-\gamma r^2} \langle\lambda\mu, \mathbf{C}|\psi_{l'}^{m'}(\mathbf{B})] \tag{7.5}$$

where the GTO primitives are centered at $\mathbf{A}$ and $\mathbf{B}$ and the ECP is centered at $\mathbf{C}$. The notation employed in the brackets refer to the idea that the angular momentum projection operator being used with the specified center.

The typical analysis of type-II ECP integrals involve carrying the integral over the angular degrees of freedom and finding some way to numerical quadrature for the radial integral, which becomes very difficult to treat analytically.

$$I^{\mu,m}_{\lambda,l}(r) = \int_0^{2\pi}\int_{-\pi/2}^{\pi/2} Y^{*\mu}_{\lambda}(\theta_1,\phi_1)\chi^m_l(r\hat{u}(\phi_1,\theta_1)-\mathbf{AC})\sin(\theta_1)d\theta_1 d\phi_1 \quad (7.6)$$

$$I'^{\mu,m'}_{\lambda,l'}(r) = \int_0^{2\pi}\int_{-\pi/2}^{\pi/2} Y^{*\mu}_{\lambda}(\theta_2,\phi_2)\psi^{m'}_{l'}(r\hat{u}(\phi_2,\theta_2)-\mathbf{BC})\sin(\theta_2)d\theta_2 d\phi_2 \quad (7.7)$$

$$\left[\chi^m_l|\hat{V}_\lambda|\psi^{m'}_{l'}\right] = \sum_{\mu=-\lambda}^{\lambda}\int_0^{\infty} r^{2+n_\lambda}e^{-\gamma r^2} I^{\mu,m}_{\lambda,l}(r) I'^{\mu,m'}_{\lambda,l'}(r)dr \quad (7.8)$$

As a first step, we choose two new systems of coordinates to evaluate the integrals over the solid angle. One choice that simplifies the solution is to keep the ECP at the origin and define the z-axis as the line containing the first or second GTO center, respectively, while the x-axis as the one containing the other center. Other than a trivial rotation of the GTOs, this attempt requires finding the base-change coefficients between the two reference frames S_1 and S_2:

$$C^{\mu\mu'}_{\lambda} = \langle S_1, \lambda\mu|\lambda\mu', S_2\rangle \quad (7.9)$$

which couple the spherical harmonics. This has to be accounted for in the angular momentum expansion of the integral:

$$[\chi^m_l|\hat{V}_\lambda|\psi^{m'}_{l'}] = \sum_{\mu=-\lambda}^{\lambda}\sum_{\mu'=-\lambda}^{\lambda} C^{\mu\mu'}_{\lambda}[\chi^m_l(\mathbf{A})|\lambda\mu,\mathbf{C}\rangle r^{n_\lambda}e^{-\gamma r^2}\langle\lambda\mu',\mathbf{C}|\psi^{m'}_{l'}(\mathbf{B})] \quad (7.10)$$

In such systems, the angular integrals are much easier to evaluate:

$$I^{\mu,m}_{\lambda,l}(r) = \int_0^{2\pi}\int_{-\pi/2}^{\pi/2} \sin(\theta)Y^{*\mu}_{\lambda}(\theta,\phi)R^m_l(\mathbf{r}-\mathbf{AC})e^{-\alpha(|r|^2-2a|r|\cos(\theta)+a^2)}d\theta d\phi \quad (7.11)$$

where $a = |\mathbf{AC}|$ and $R^m_l(\mathbf{r}) = |r|^l Y^m_l(\hat{r})$ refers to the usual solid harmonic multiplying the GTO Gaussian expansion. These can be expanded in the center of coordinates by a simple translation of the harmonics in the z-axis. Because such translations in z preserve the m component, it is straightforward to check that the integral over ϕ will be 0 unless $\mu = m$. The change of variable $t = \cos(\theta)$ further simplifies the expression:

$$I'^{\mu,m}_{\lambda,l}(r) = \delta_{m\mu} \sum_{l'=|\mu|}^{l} \binom{l}{l'} R^0_{l-l'}(a) N^\mu_l N^\mu_{l'} J^\mu_{\lambda,l'}(r) \tag{7.12}$$

$$J^\mu_{\lambda,l'}(r) = r^{l'} \int_{-1}^{1} P^\mu_\lambda(t) P^\mu_{l'}(t) e^{-\alpha(r^2-2art+a^2)} dt \tag{7.13}$$

where $P^m_l(z)$ are the usual associated Legendre polynomials, the new index l' corresponds to the summation over the new solid harmonics, translated to the origin, and $a = |AC| = AC_z$ is the distance of the GTO from the origin. An analogous treatment is done for the sibling GTO. Because the associated Legendre polynomials inside the integral share the same value of m, their product is always a polynomial it t, free from the term $(1-t^2)^{1/2}$. Instead of finishing the integration over the solid angle for both functions, which would result in a Bessel function expansion, and attempt the integration over r we switch the order of integration and use a different approach. To avoid complicating the notation with constant terms and indices, let's focus in the following auxilliary integrals, from which every other integral can be expressed as a linear combination of:

$$Q^{m,l}_n = \int_0^\infty \int_{-1}^{+1} \int_{-1}^{+1} r^n s^m t^l e^{-\gamma r^2} e^{-\alpha(r^2-2ars+a^2)} e^{-\beta(r^2-2brt+b^2)} dr ds dt \tag{7.14}$$

from which it is possible to reconstruct the previous integrals, given that:

$$[J^{\mu_A}_{\lambda,l_A}(r)|r^n e^{-\gamma r^2}|J'^{\mu_B}_{\lambda,l_B}(r)] = \sum_{u,v} L^{\mu_A,\lambda,l_A}_u L^{\mu_B,\lambda,l_B}_v Q^{u,v}_{n+l_A+l_B} \tag{7.15}$$

$$P^m_{l_1}(z) P^m_{l_2}(z) = \sum_{u=0}^{l_1+l_2} L^{m,l1,l2}_u z^u \tag{7.16}$$

By deriving the expression over r, applying the chain rule, integrating both sides and rearranging terms, we obtain a useful recurrence relation:

$$2(\alpha+\beta+\gamma) Q^{m,l}_{n+1} = Q^{m,l}_n + n Q^{m,l}_{n-1} + 2\alpha a Q^{m+1,l}_n + 2\beta b Q^{m,l+1}_n \tag{7.17}$$

This relation can be applied until exhaustion of the index n, reducing the set of integrals needed to just the subset $Q^{m,l}_0$. The integration over r becomes simple for $n = 0$ and results in:

$$Q_0^{m,l} = \frac{\sqrt{\pi}}{2\sqrt{\alpha+\beta+\gamma}} e^{-\alpha a^2-\beta b^2} M_{m,l} \tag{7.18}$$

$$M_{m,l} = \int_{-1}^{+1}\int_{-1}^{+1} t^m s^l H(s,t) ds dt \tag{7.19}$$

$$H(s,t) = e^{\frac{(\alpha a s+\beta b t)^2}{\alpha+\beta+\gamma}} \left(1 + \mathrm{erf}\left(\frac{\alpha a s + \beta b t}{\sqrt{\alpha+\beta+\gamma}}\right)\right) \tag{7.20}$$

To compute the double integral, a linear transformation of the variables s, t is done using the following rotation/scaling:

$$\begin{aligned} u &= (\alpha+\beta+\gamma)^{-1/2}(\alpha a s + \beta b t) && (7.21)\\ v &= (\alpha+\beta+\gamma)^{-1/2}(-\beta b s + \alpha a t) && (7.22) \end{aligned}$$

The integral boundaries must be modified accordingly. The symmetry of the function (its behaviour with respect to the center of inversion) is also useful to simplify the resulting expression. After some manipulation, we found two general solutions:

$$\begin{aligned} M_{m,l} &= \left(\int_{-p-q}^{-p+q}\int_0^{(p/q)u+(p^2+q^2)/q} F_{m,l}(u,v) du dv \right. && (7.23)\\ &+ \left.\int_{-p+q}^{p+q}\int_0^{-(q/p)u+(p^2+q^2)/p} F_{m,l}(u,v) du dv\right) 2(p^2+q^2)^{-m-l-1} \\ F_{m,l}(u,v) &= e^{u^2}(pu-qv)^m(qu+pv)^l \qquad m+l = even && (7.24)\\ F_{m,l}(u,v) &= e^{u^2}\mathrm{erf}(u)(pu-qv)^m(qu+pv)^l \qquad m+l = odd && (7.25)\\ p &= (\alpha+\beta+\gamma)^{-1/2}\alpha a && (7.26)\\ q &= (\alpha+\beta+\gamma)^{-1/2}\beta b && (7.27) \end{aligned}$$

Expanding the powers in l and m and integrating over v yields auxilliary integrals of the type:

$$\begin{aligned} E_n &= \int_{u_0}^{u_f} u^n e^{u^2} du && (7.28)\\ O_n &= \int_{u_0}^{u_f} u^n e^{u^2} erf(u) du && (7.29) \end{aligned}$$

For the $m+l=2k$ and $m+l=2k+1$ cases, respectively, with $u_0 = -p \pm q$ and $u_f = \pm p + q$. Integrating by parts and rearranging terms provides two useful recurrence relations for these integrals.

$$E_{n+1} = \frac{1}{2}\left[u^n e^{u^2}\right]_{u_0}^{u_f} - \frac{1}{2} n E_{n-1} \tag{7.30}$$

$$O_{n+1} = \frac{1}{2}\left[u^n e^{u^2} \mathrm{erf}(u)\right]_{u_0}^{u_f} - \frac{\pi^{-1/2}}{n+1}\left[u^{n+1}\right]_{u_0}^{u_f} - \frac{1}{2} n O_{n-1} \tag{7.31}$$

These recurrence relations terminate for $n = 2k+1$ cases without a last term left to evaluate. For $n = 2k$, they terminate with the $n = 0$ integrals, which evaluate to:

$$E_0 = \left[e^{u^2} D_+(u)\right]_{u_0}^{u_f} \tag{7.32}$$

$$O_0 = \pi^{-1/2}\left[u^2 {}_2F_2(1,1;2,3/2;u^2)\right]_{u_0}^{u_f} \tag{7.33}$$

Where $D_+(u)$ is the Dawson function, and ${}_2F_2(1,1;2,3/2;u^2)$ is a generalized hypergeometric function. The potential divergence of these integrals due to their asymptotic $u^{-1}e^{u^2}$ behaviour is cancelled for large values with the larger $e^{-\alpha a^2 - \beta b^2}$ prefactor from the $Q_0^{m,l}$ integral.

The integrals appearing in E_0 and O_0 can be evaluated similarly to the incomplete gamma function and its derivatives. The (approximate) interval $[0,6]$, can be divided in equal-length segments, on each of which the functions are approximated using Chebyshev polynomials. For arguments $u > 6$, the two functions coincide to machine precision, and can be evaluated numerically using the following Taylor series:

$$\pi^{-1/2} u^2 e^{-u^2} {}_2F_2(1,1;2,3/2;u^2) \simeq D_+(u) = \frac{1}{2z} + \frac{1}{2z^3} + \frac{3}{8z^5} + \dots \tag{7.34}$$

which converges after a few terms.

The analytic solution here presented is very general, and can be applied to all sorts of semilocal pseudopotentials expressable as sums of $|r|^\lambda e^{-\gamma r^2}$ terms, including $\lambda \geq -2$ and for integer λ.

7.2 Algorithm optimization

Since all spherical harmonic rotations, translations and associated Legendre polynomial expansions are independent of the Gaussian exponents and weights, they can be carried out outside any accumulation loops. The computational bottleneck is therefore in the evaluation of all necessary integrals of the class $Q_n^{m,l}$ inside a triple loop running for all combinations of primitives.

One potential issue of using the formulas presented without further refinement is the possibility of loss of numerical precision due to partial cancellations of large magnitudes. A possible approach to this problem that can be made computationally efficient is to use an exact Gaussian quadrature for the $H(s,t)$ kernel, which is an approach reminiscent of the Rys quadrature method for 2-electron integrals[81]. This is accomplished by generating a set of orthogonal polynomials over the kernel and computing the necessary weights and abcissas from them. The e^{u^2} and $e^{u^2}erf(u)$ kernels are also good candidates for exact quadrature rules.

Let's turn our attention again to equation 7.17. The recurrence relation involves, other than the integrals, some Gaussian exponents and two geometric parameters, $a = |\mathbf{AC}|$ and $b = |\mathbf{BC}|$. Let's introduce a braket-like notation for the sake of simpler manipulation:

$$[m_i|n_k|l_j] = Q_n^{m,l}(\alpha_i, \beta_j, \gamma_k) \tag{7.35}$$

Equation 7.17 is expressed as:

$$\begin{aligned} 2(\alpha_i + \beta_j + \gamma_k)[m_i|(n+1)_k|l_j] &= [m_i|n_k|l_j] + n[m_i|(n-1)_k|l_j] + \\ &+ 2\alpha_i a[(m+1)_i|n_k|l_j] + \\ &+ 2\beta_j b[m_i|n_k|(l+1)_j] \end{aligned} \tag{7.36}$$

Following the usual early-contraction methodology, and introducing the contracted quantities:

$${}^{u}(m|n^w|l)^v = \sum_i^{K_a}\sum_j^{K_b}\sum_k^{K_c} D_i D_j D_k \frac{(2\alpha_i)^u (2\beta_j)^v}{(2(\alpha_i+\beta_j+\gamma_k))^w}[m_i|n_k|l_j] \tag{7.37}$$

the recurrence relation can be "contracted":

$$\begin{aligned} {}^{u}(m|(n+1)^{w}|l)^{v} &= {}^{u}(m|n^{w+1}|l)^{v} + n\ {}^{u}(m|(n-1)^{w+1}|l)^{v} \\ &+ |\mathbf{AC}|\ {}^{u+1}(m+1|n^{w+1}|l)^{v} \\ &+ |\mathbf{BC}|\ {}^{u}(m|n^{w+1}|l+1)^{v+1} \end{aligned} \tag{7.38}$$

and be performed outside the accumulation loops. A possible algorithm using the method and techniques here presented is described next:

```
loop over angular momentum L of SL operator
  loop over GTO primitives of shell a
    loop over GTO primitives of shell b
      loop over primitives of the ECP
        compute [m|0|l] integrals
        contract integrals over w-index
      end loop
      contract integrals over v-index
    end loop
    contract integrals over u-index
  end loop
  apply CRR to generate (m|n|l) integrals
  reconstruct associated legendre polynomials
  contract integrals
end loop
translate the solid harmonic expansions to the GTO centers
rotate the solid harmonics of the batch to the lab frame of reference
```

7.3 Screening

Since ECP integrals involve the product of three functions that are localized in space, efficient screening is key to both achieve $\mathcal{O}(N)$ scaling for the molecule as well as to increase the overall performance of a batch by discarding Gaussian triplets with very small amplitude. Absolute upper bounds for ECP type 2 integrals -and rigorous criteria for the inclusion of Gaussian triplets- can be assessed by use of the usual Cauchy-Schwarz inequality and the Frobenius norm. Given a threshold τ, it suffices

$$\kappa_a \kappa_b \leq \tau \tag{7.39}$$

$$\kappa_a = \left(\sum_{m_a} (a_{l_a}^{m_a} | \hat{V}_{SL} | a_{l_a}^{m_a}) \right)^{1/2} \tag{7.40}$$

$$\kappa_b = \left(\sum_{m_b} (b_{l_b}^{m_b} | \hat{V}_{SL} | b_{l_b}^{m_b}) \right)^{1/2} \tag{7.41}$$

to guarantee that the Frobenius norm of the tensor block will be less or equal than τ:

$$\| (a_{l_a}^{m_a} | \hat{V}_{SL} | b_{l_b}^{m_b}) \|_F \leq \tau \tag{7.42}$$

Moreover, by defining $\kappa_a^{(k)}$ and $\kappa_b^{(k')}$ as the previous traces computed only for the last $K_A - k$ and $K_B - k'$ primitives of the shell, where K_A and K_B are the corresponding contraction degrees, an effective GTO primitive screening can be implemented by selecting for every GTO pair the minimal values of k and k' for which the following inequality still holds:

$$\kappa_a^{(k)} \kappa_b + \kappa_a \kappa_b^{(k')} \leq \tau \tag{7.43}$$

This test can be performed for each individual term of the sum of the semilocal potential expansion to compute only the relevant Gaussian triplets.

Chapter 8

Summary and outlook

The present thesis includes five papers, summarized next.

Paper I describes the K4+MIRROR algorithm, an efficient algorithm to evaluate electron-repulsion integrals over contracted Gaussian-type orbitals. The algorithm uses an array of techniques to simplify many steps over the previous best-performing algorithms, some of which are novel. The FLOP count for all combinations of angular momenta is the best published in the literature for the high contraction limit. The implementation referenced in the text and its performance correspond to early versions of the ERI algorithm and the EFS, not as efficient as the later versions interfaced to *Dalton*, and served to get a tentative idea of what the performance could be.

Paper II introduces the QM/CMM method, a new multiscale approach designed to compute the properties of molecules embedded in heterogeneous systems containing metallic and nonmetallic regions, such as nanoparticles in solution or metal surfaces in contact with a solvent. The molecule is described with DFT, the nonmetallic region with a polarizable force field, and the metallic region with the polarization-capacitance model. The properties of the system are predicted through linear response theory. The method is tested by computing the spectra of thymidine fisisorbed on a gold surface in contact with an aqueous environment, for gold surfaces defined by different Miller indices.

Paper III extends the QM/CMM method to quadratic response theory, allowing the computation of non-linear optical properties and other mixed properties to molecules in complex heterogeneous environments. The method is applied to study the chromophore para-nitroanilinne (PNA) adsorbed in gold surfaces with different solvents. In particular, the variations in the principal components of the non-linear susceptibility tensor are monitored for different solvents.

Paper IV (submitted) analyzes the feasibility of performing TD-DFT (Time-Dependent Density Functional Theory) computations with ANO basis sets. Results show that *aug-ANO* basis sets produce results of quality comparable to *def2* basis with an extra unit of angular momentum. The implemention of the K4+MIRROR algorithm in the EFS library proves to be fast enough to perform SCF and TD-DFT (linear response) calculations with the *aug-ANO* basis at a fraction of the cost of the much simpler *def2* basis, computed with the traditional Hermit module in Dalton.

Paper V (in manuscript) introduces new recurrence relations for ERIs, which are used to generate a more efficient version of the K4+MIRROR algorithm, with reduced FLOP count. The CDR/AERR recurrence relations reduce the number of FLOPs and asymptotic complexity of the K4 contraction step, while the SKS can be used to reduce the number of kernels and FLOPs needed for spherical GTOs.

The Echidna Fock Solver library, wherein the CDR/K4+MIRROR algorithms and the previously discussed methods to compute and accelerate the formation of the Coulomb and Exchange matrices have been implemented, is also the product of the research and work done in the current thesis. The EFS + Quimera libraries are GPL version 3, and have been interfaced to the Dalton 2013[122] quantum chemistry program suite, as also has the QM/CMM integral module.

There are three major areas in which the work here presented can be improved or extended. First, it would be interesting to extend the automation methodology of generating ERI routines to a more general code capable of synthetizing specialized molecular integral libraries for different target architectures, accelerating thus the development of new features in a QC code. This could be accomplished through a library of general recurrence relations and heuristics and a simple front-end for specifying the form of the integral required and the valid range of parameters for which the specialized functions will be generated. Libraries for computing ERI derivatives are of special interest, even if a less general approach is used for their generation. Likewise, the three- and four-electron integrals appearing in some explicitely correlated methods are a good candidate, considering these methods need of other integrals of lower order, too.

Second, some preliminary results not published in the present thesis suggest that a universal, even-tempered, general contraction basis sets could have a big computational advantage over other GC basis sets in that they admit a particularly fast ERI algorithm, due to the many degenerate centers for the Gaussian product expansion. It might be interesting to study the accuracy of such basis for some typical benchmark set, and the real-case performance of the ERI algorithms. Independent of the last point, it might be possible to speed-up the computation of regular ERIs within some bounded error with the K4+MIRROR algorithm by commputing the Singular Value Decomposition of the $[_{ij}m_{kl}]$ kernel tensors of an ERI batch and applying the contraction steps and most of the transformations to the reduced number of unitary vectors.

A question that remains unanswered is whether it is possible to transform the $O(L^4)$ CDR kernels into the $O(L^4)$ integrals of a spherical ERI batch through an asymptotically faster method than the $O(L^8)$ FLOPs due to the MIRROR step. Some analysis suggests that such method does indeed exist, and might potentially reduce the number of operations of the CDR/K4+MIRROR algorithm below many non-early-contraction algorithms, making it a universal scheme for all ERIs.

As for the QM/CMM and ECP integrals, computing the forces is greatly simplified by considering the translation invariance of the integrals. The ECP integral code is in an early implementation phase, but provides numerical results corroborated by computer algebra software.

REFERENCES

[1] The nobel prize in chemistry 1998. http://www.nobelprize.org/nobel_prizes/chemistry/laureates/1998/.

[2] The nobel prize in chemistry 2013. http://www.nobelprize.org/nobel_prizes/chemistry/laureates/2013/.

[3] B. Odom, D. Hanneke, B. D'Urso, and G. Gabrielse. New measurement of the electron magnetic moment using a one-electron quantum cyclotron. *Phys. Rev. Lett.*, 97:030801, 2006.

[4] S. F. Boys, G. B. Cook, C. M. Reeves, and I. Shavitt. Automatic fundamental calculations of molecular structure. *Nature*, 178:1207, 1956.

[5] John Backus. The history of FORTRAN. http://www.softwarepreservation.org/projects/FORTRAN/paper/p25-backus.pdf, 1981. IBM Corporation, Research Division.

[6] Parallel Virtual Machine. http://www.csm.ornl.gov/pvm/pvm_home.html.

[7] Ewing Lusk, Nathan Doss, and Anthony Skjellum. A high-performance, portable implementation of the MPI message passing interface standard. *Parallel Computing*, 22:789–828, 1996.

[8] Russell Fish. The future of computers - Part 1: Multicore and the memory wall. http://www.edn.com/design/systems-design/4368705/The-future-of-computers--Part-1-Multicore-and-the-Memory-Wall, 2011.

[9] David P. Rodgers. Improvements in multiprocessor system design. *SIGARCH Comput. Archit. News*, 13:225–231, 1985.

[10] Robert H. Dennard, Fritz Gaensslen, Hwa-Nien Yu, Leo Rideout, Ernest Bassous, and Andre LeBlanc. Design of ion-implanted MOSFET's with very small physical dimensions. *IEEE Journal of Solid State Circuits*, SC–9 (5), 1974.

[11] Zhe Fan, Feng Qiu, Arie Kaufman, and Suzanne Yoakum-Stover. GPU cluster for High Performance Computing. In *Proceedings of the 2004 ACM/IEEE Conference on Supercomputing*, SC '04. IEEE Computer Society, 2004.

[12] Matrix Algebra on GPU and Multicore Architectures. http://icl.cs.utk.edu/magma/.

[13] George Chrysos. Intel Xeon Phi coprocessor - the architecture. https://software.intel.com/en-us/articles/intel-xeon-phi-coprocessor-codename-knights-corner, 2012. Intel Corporation.

[14] DARPA Robert Colwell, Director of Microsystems Technology Office. The chip design game at the end of Moore's law. In *Proceedings of the Hotchips conference 25*, 2013.

[15] Peter Kogge et al. Exascale computing study: Technology challenges in achieving exascale systems, 2008.

[16] Tianhe-2 (milkyway-2) - TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P. http://www.top500.org/system/177999. National Super Computer Center in Guangzhou, China.

[17] Joe Cross (Program Manager). Power efficiency revolution for embedded computing technologies (PERFECT). DARPA Micosystems Technologies Office.

[18] TSUBAME-KFC - LX 1U-4GPU/104Re-1G cluster, intel xeon e5-2620v2 6c 2.100ghz, Infiniband FDR, NVIDIA K20x. http://www.green500.org. GSIC Center, Tokyo Institute of Technology.

[19] GRAPE-DR cluster - GRAPE-DR accelerator cluster, infiniband. http://www.top500.org/176224. National Astronomical Observatory of Japan.

[20] Junichiro Makino and Hiroshi Daisaka. GRAPE-8 - An accelerator for gravitational N-body simulation with 20.5 Gflops/W performance. In *Proceedings of the SC conference 12*, 2012.

[21] David E. Shaw, Martin M. Deneroff, Ron O. Dror, Jeffrey S. Kuskin, Richard H. Larson, John K. Salmon, Cliff Young, Brannon Batson, Kevin J. Bowers, Jack C. Chao, Michael P. Eastwood, Joseph Gagliardo, J. P. Grossman, C. Richard Ho, Douglas J. Ierardi, István Kolossváry, John L. Klepeis, Timothy Layman, Christine McLeavey, Mark A. Moraes, Rolf Mueller, Edward C. Priest, Yibing Shan, Jochen Spengler, Michael Theobald, Brian Towles, and Stanley C. Wang. Anton, a special-purpose machine for molecular dynamics simulation. *Commun. ACM*, 51:91–97, 2008.

[22] Matteo Frigo and Steven G. Johnson. The design and implementation of FFTW3. *Proceedings of the IEEE*, 93:216–231, 2005.

[23] CUDA 6.0 performance report, 2014. NVIDIA.

[24] The breakthrough advantage for FPGAs with tri-gate technology (white paper). ALTERA.

[25] K. Nakamura, H. Hatae, M. Harada, Y. Kuwayama, M. Uehara, H. Sato, S. Obara, H. Honda, U. Nagashima, Y. Inadomi, and K. Murakami. ERIC: A special-purpose processor for ERI calculations in quantum chemistry applications. In *Proceedings of the 5th International Conference on High Performance Computing and Grid in Asia Pacific Region*, 2002.

[26] T. Ramdas, G.K. Egan, D. Abramson, and K. Baldridge. Converting massive TLP to DLP: A special-purpose processor for molecular orbital computations. In *Proceedings of the 4th International Conference on Computing Frontiers*, CF '07, pages 267–276, 2007.

[27] President's Council of Advisors on Science and Technology. Designing a digital future: Federally funded research and development in networking and information technology, 2010.

[28] M. Häser and R. Ahlrichs. Improvements on the direct SCF methods. *J. Comput. Chem.*, 10:104, 1989.

[29] Robert A. Van De Geijn and Jerrell Watts. SUMMA: Scalable universal matrix multiplication algorithm. Technical report, Department of Computer Sciences, University of Austin and Scalable Concurrent Programming Laboratory, California Institute of Technology, 1997.

[30] Erwin Schrödinger. An undulatory theory of the mechanics of atoms and molecules. *Physical Review*, 28:1049–1070, 1926.

[31] P. A. M. Dirac. A new notation for quantum mechanics. *Mathematical proceedings of the Cambridge Philosophical Society*, 35:416–418, 1939.

[32] M. Born and R. Oppenheimer. Zur Quantentheorie der Molekeln. *Annalen der Physik*, 389:457–484, 1927.

[33] E. Teller. The crossing of potential surfaces. *The Journal of Physical Chemistry*, 41:109–116, 1937.

[34] Michael Dolg. *Modern Methods and Algorithms of Quantum Chemistry, Proceedings, Second Edition*, chapter Effective Core Potentials, pages 507–540. John-von-Neumann-Inst. for Computing, 2000.

[35] M. Plischke and B. Bergersen. *Equilirium Statistical Physics, 3rd edition.* World Scientific Publishing Co. Pte., 2006.

[36] A. Szabo and N. S. Ostlund. *Modern Quantum Chemistry: Introduction to Advanced Electronic Structure Theory.* MacMillan, New York, 1982.

[37] C. C. J. Roothaan. New developments in molecular orbital theory. *Rev. Mod. Phys.*, 23:69–89, 1951.

[38] R. McWeeny. Some recent advances in density matrix theory. *Rev. Mod. Phys.*, 32:335–369, 1960.

[39] Chr. Møller and M. S. Plesset. Note on an approximation treatment for many-electron systems. *Phys. Rev.*, 46:618–622, 1934.

[40] Jíří Čížek. On the Correlation Problem in Atomic and Molecular Systems. Calculation of Wavefunction Components in Ursell-Type Expansion Using Quantum-Field Theoretical Methods. *The Journal of Chemical Physics*, 45:4256–4266, 1966.

[41] A.O. Mitrushenkov, G. Fano, F. Ortoladi, R. Lingueri, and P. Palmieri. Quantum chemistry using the Density Matrix Renormalization Group. *J. Chem. Phys.*, 115:6815, 2001.

[42] J. A. Pople and R. K. Nesbet. Self-Consistent Orbitals for Radicals. *The Journal of Chemical Physics*, 22:571–572, 1954.

[43] C. C. J. Roothaan. Self-consistent field theory for open shells of electronic systems. *Rev. Mod. Phys.*, 32:179–185, 1960.

[44] P. Hohenberg and W. Kohn. Inhomogeneous electron gas. *Phys. Rev.*, 136:B864–B871, 1964.

[45] David A. Mazziotti. Structure of fermionic density matrices: Complete n-representability conditions. *Phys. Rev. Lett.*, 108:263002, 2012.

[46] Enrico Fermi. Un metodo statistico per la determinazione di alcune prioprietà dell'atomo. *Rend. Accad. Naz. Lincei*, page 602–607, 1927.

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

[48] A. Warshel and M. Levitt. Theoretical studies of enzymic reactions: dielectric, electrostatic and steric stabilization of the carbonium ion in the reaction of lysozyme. *Journal of molecular biology*, 103:227–249, 1976.

[49] Z. Rinkevicius, X. Li, J.A.R. Sandberg, K.V. Mikkelsen, and H. Ågren. A hybrid density functional theory/molecular mechanics approach for linear response properties in heterogeneous environments. *Journal of Chemical Theory and Computation*, 10:989–1003, 2014.

[50] L.L. Jensen and L. Jensen. Electrostatic interaction model for the calculation of large noble metal nanoclusters. *J. Phys. Chem.*, 112:15697–15703, 2008.

[51] T. Kjaergaard, P. Jörgensen, J. Olsen, S. Coriani, and T. Helgaker. Hartree-Fock and Kohn-Sham time-dependent response theory in a second-quantization atomic-orbital formalism suitable for linear scaling. *J. Phys. Chem.*, 129:054106, 2008.

[52] R. P. Feynman. Forces in molecules. *Phys. Rev.*, 56:340–343, 1939.

[53] J. E. Lennard-Jones. The electronic structure of some diatomic molecules. *Trans. Faraday Soc.*, 25:668–686, 1929.

[54] Jansen H.B. and P. Ros. Non-empirical molecular orbital calculations on the protonation of carbon monoxide. *Chem. Phys. Lett.*, 3:140, 1969.

[55] B. Liu and A.D. McLean. Accurate calculation of the attractive interaction of two ground state helium atoms. *J. Chem. Phys.*, 59:4557, 1973.

[56] I. Mayer. Towards a chemical Hamiltonian. *Intern. J. of Quantum Chem.*, 23:341, 1983.

[57] S.F. Boys and F. Bernardi. The calculation of small molecular interactions by the differences of separate total energies. some procedures with reduced errors. *Mol. Phys.*, 19:553, 1970.

[58] P. Pulay. Ab initio calculation of force constants and equilibrium geometries in polyatomic molecules. *Mol. Phys.*, 17:167, 1969.

[59] W.J. Hehre, R. F. Stewart, and J. A. Pople. Self-consistent molecular orbital methods. I. Use of Gaussian expansions of Slater-type atomic orbitals. *The Journal of Chemical Physics*, 51:2657–2664, 1969.

[60] R. Ditchfield, W. J. Hehre, and J. A. Pople. Self-consistent molecular orbital methods. IX. An extended Gaussian-type basis for molecular orbital studies of organic molecules. *The Journal of Chemical Physics*, 54:724–728, 1971.

[61] J.A. Pople and W.J. Hehre. Computation of electron repulsion integrals involving contracted Gaussian basis functions. *J. Comput. Phys.*, 27:161, 1978.

[62] B. Roos and P. Siegbahn. Polarization functions for first and second row atoms in gaussian type MO-SCF calculations. *Theoretica chimica acta*, 17:199–208, 1970.

[63] Huzinaga S. Gaussian-type functions for polyatomic systems. I. *J. Chem. Phys.*, 42:1293–1302, 1965.

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

[65] E.A. Hylleraas. Neue Berechnung der Energie des Heliums im Grundzustande, sowie des tiefsten Terms von Orto-Helium. *Zeitschrift für Physik*, 54:347, 1929.

[66] K. Szalewicz, B. Jeziorski, H.J. Monkhorst, and J.G. Zabolitzky. A new functional for variational calculation of atomic and molecular second- order correlation energies. *Chem. Phys. Lett.*, 91:169–172, 1982.

[67] C. Hättig, D.P. Tew, and A. Köhn. Accurate and efficient approximations to explicitly correlated coupled-cluster singles and doubles, CCSD-F12. *J. Chem. Phys.*, 132:231102, 2010.

[68] R.C. Raffenetti. General contraction of Gaussian atomic orbitals: Core, valence, polarization, and diffuse basis sets; molecular integral evaluation. *J. Chem. Phys.*, 58:4452, 1973.

[69] J. Almlöf and P.R. Taylor. General contraction of Gaussian basis sets. i. atomic natural orbitals for first and second row atoms,. *J. Chem. Phys.*, 86:4070, 1987.

[70] C.M. Reeves and M.C. Harrison. Use of Gaussian functions in the calculation of wavefunctions for small molecules. ii. the ammonia molecule. *J. Chem. Phys.*, 39:11, 1963.

[71] S. Huzinaga and M. Klobukowski. Well-tempered GTF basis sets for the atoms K through Xe. *Chem. Phys. Lett.*, 120:509, 1985.

[72] I. Cherkes, S. Klaiman, and N. Moiseyev. Spanning the Hilbert space with an even tempered basis set. *Intern. J. of Quant. Chem.*, 109:2996, 2009.

[73] B.I. Dunlap. Robust variational fitting: Gáspár's variational exchange can accurately be treated analytically. *THEOCHEM*, 501-502:221, 2000.

[74] Basic Linear Algebra Subprograms (BLAS). http://www.netlib.org/blas/.

[75] LAPACK - Linear Algebra PACKage. http://www.netlib.org/lapack/.

[76] ScaLAPACK - Scalable Linear Algebra PACKage. http://www.netlib.org/scalapack/.

[77] SLEPc - Scalable Library for Eigenvalue Problem Computations. http://www.grycap.upv.es/slepc/.

[78] G. Baumgartner, A. Auer, D.E. Bernholdt, A. Bibireata, and al. Synthesis of high-performance parallel programs for a class of ab initio quantum chemistry models. *Proceedings of the IEEE*, 93:2, 2005.

[79] M. Head-Gordon and J.A. Pople. A method for two-electron Gaussian integral and integral derivative evaluation using recurrence relations. *J. Chem. Phys.*, 89:5777, 1988.

[80] L.E. McMurchie and E.R. Davidson. One- and two-electron integrals over cartesian Gaussian functions. *J. Comput. Phys.*, 26:218, 1978.

[81] M. Dupuis, J. Rys, and H.F. King. Evaluation of molecular integrals over Gaussian basis functions. *J. Chem. Phys.*, 65:111, 1976.

[82] E.O. Steinborn and K. Ruedenberg. Rotation and translation of regular and irregular solid spherical harmonics. *Adv. Quant. Chem.*, 7:1, 1973.

[83] R. Beatson and L. Greengard. A short course on Fast Multipole Methods. http://math.nyu.edu/faculty/greengar/shortcourse_fmm.pdf.

[84] H. Nakai and M. Kobayashi. New algorithm for the rapid evaluation of electron repulsion integrals: Elementary Basis Algorithm. *Chem. Phys. Lett.*, 388:50, 2004.

[85] H. Nakai and M. Kobayashi. New recurrence relations for the rapid evaluation of electron repulsion integrals based on the Accompanying Coordinate Expansion formula. *J. Chem. Phys.*, 121:4050, 2004.

[86] T. Ramdas, G.K. Egan, D. Abramson, and K.K. Baldridge. Using extrinsic vectorization and shell structure for efficient SIMD evaluation of electron repulsion integrals. *Chem. Phys.*, 249:147, 2008.

[87] S. Huzinaga, J. Andzelm, E. Radzio-Andzelm, Y. Sakai, H. Tatewaki, and M. Klobukowski. *Gaussian Basis Sets for Molecular Calculations.* Elsevier, Amsterdam, 1984.

[88] W.J. Hehre, W.A. Lathan, M.D. Newton, R. Ditchfield, and J.A. Pople. Program no. 136.

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

[90] P.M.W. Gill and J.A. Pople. The PRISM algorithm for two-electron integrals. *Int. J. of Quant. Chem.*, 40:753, 1991.

[91] P.M.W. Gill, M. Head-Gordon, and J.A. Pople. Efficient computation of two-electron-repulsion integrals and their n-th order derivatives using contracter Gaussian basis sets. *J. Phys. Chem.*, 94:5564, 1990.

[92] T. Yanai, K. Ishida, H. Nakano, and K. Hirao. New algorithm for electron repulsion integrals oriented to the general contraction scheme. *Int. J. of Quant. Chem.*, 76:396, 2000.

[93] S. Reine, E. Tellgren, A. Krapp, T. Kjaergaard, T. Helgaker, and et al. Variational and robust density fitting of four-center two-electron integrals in local metrics. *J. Chem. Phys.*, 129:104101, 2008.

[94] T.R. Adams, R.D. Adamson, and P.M.W. Gill. A tensor approach to two-electron matrix elements. *J. Chem. Phys.*, 107:124, 1997.

[95] B. Gibb, R. Gibb, and M. Gibb. Saturday night fever. RSO, 1977.

[96] J. Almlöf, K. Faegri, and K. Korsell. Principles of a direct SCF approach to LCAO-MO ab-initio calculations. *J. Comput. Chem.*, 3:385, 1982.

[97] S.A. Maurer, D.S. Lambrecht, D. Flaig, and C. Ochsenfeld. Distance-dependent Schwarz-based integral estimates for two-electron integrals: reliable tightness vs. rigorous upper bounds. *J. Chem. Phys.*, 136:144102, 2012.

[98] P. Pulay. Convergence acceleration of iterative sequences. the case of SCF iteration. *Chem. Phys. Lett.*, 73:393, 1980.

[99] K.N. Kudin, G.E. Scuseria, and E. Cancès. A black-box self-consistent field convergence algorithm: One step closer. *J. Chem. Phys.*, 116:8255, 2002.

[100] E. Cancès and C. Le Bris. Can we outperform the DIIS approach for electronic structure calculations? *Int. J. Quant. Chem.*, 79:82, 2000.

[101] P.G. Mezey. Macromolecular density matrices and electron densities with adjustable nuclear geometries. *J. Math. Chem.*, 18:141, 1995.

[102] Luis Seijo and Zoila Barandiarán. Parallel, linear-scaling building-block and embedding method based on localized orbitals and orbital-specific basis sets. *J. Chem. Phys.*, 121:6698, 2004.

[103] Ross D. Adamson, Jeremy P. Dombroski, and Peter M.W. Gill. Chemistry without coulomb tails. *Chemical Physics Letters*, 254:329–336, 1996.

[104] Vahtras O., J. Almlöf, and M.W. Feyereisen. Integral approximations for lcao-scf calculations. *Chemical Physics Letters*, 213:514–518, 1993.

[105] F. Aquilante, L. Gagliardi, T.B. Pedersen, and R. Lindh. Atomic Cholesy decompositions: A route to unbiased auxiliary basis sets for density fitting approximation with tunable accuracy and efficiency. *The Journal of Chemical Physics*, 130:154107, 2009.

[106] G.R. Ahmadi and J. Almlöf. The Coulomb operator in a Gaussian product basis. *Chem. Phys. Lett.*, 246:364–370, 1995.

[107] L. Greengard and V. Rokhlin. A fast algorithm for particle simulation. *Journal of Computational Physics*, 73:325–348, 1987.

[108] C.A. White, B.G. Johnson, P.M.W. Gill, and M. Head-Gordon. The continuous fast multipole method. *Chemical Physiscs Letters*, 230:8–16, 1994.

[109] J.P. Dombroski, S.W. Taylor, and P.M.W. Gill. KWIK: Coulomb energies in O(N) work. *J. Chem. Phys.*, 100:6272, 1996.

[110] László Füsti-Molnár and Peter Pulay. The fourier transform coulomb method: Efficient and accurate calculation of the Coulomb operator in a Gaussian basis. *The Journal of Chemical Physics*, 117:7827–7835, 2002.

[111] V. Weber and M. Challacombe. Parallel algorithm for the computation of the Hartree-Fock exchange matrix: Gas phase and periodic parallel ONX. *J. Chem. Phys.*, 125:104110, 2006.

[112] C. Ochsenfeld, C.A. White, and M. Head-Gordon. Linear and sublinear scaling formation of Hartree-Fock-type exchange matrices. *J. Chem. Phys.*, 109:1663, 1998.

[113] F. Neese, F. Wennmohs, A. Hansen, and U. Becker. Efficient, approximate and parallel Hartree-Fock and hybrid DFT calculations. a 'chain-of-spheres' algorithm for the hartree-fock exchange. *Chem. Phys.*, 356:98, 2009.

[114] R.A. Friesner. Solution of the self-consistent field electronic structure equations by a pseudospectral method. *Chem. Phys. Lett.*, 116:39, 1985.

[115] Tirath Ramdas, Gregory K. Egan, David Abramson, and Kim K. Baldridge. On {ERI} sorting for {SIMD} execution of large-scale hartree–fock {SCF}. *Computer Physics Communications*, 178:817 – 834, 2008.

[116] M.F. Saunders, V.R. and; Guest. Applications of the CRAY-1 for quantum chemistry calculations. *Comput. Phys. Commun.*, 26:389, 1982.

[117] S.F. Boys. Electronic wave functions. i. a general method of calculation for the stationary states of any molecular system. *Prog. R. Soc.*, A 200:542, 1950.

[118] L. Kleinman and D.M. Bylander. Efficacious form for model pseudopotentials. *Phys. Rev. Lett.*, 48:1425, 1982.

[119] M. Dolg and X. Cao. Relativistic pseudopotentials: Their development and scope of application. *Chem. Rev.*, 112:403, 2012.

[120] L.E. McMurchie and E.R. Davidson. Calculation of integrals over ab initio pseudopotentials. *J. Comput. Phys.*, 44:289, 1981.

[121] C.K. Skylaris, L. Gagliardi, N.C. Handy, A.G. Ioannou, S. Spencer, A. Willetts, and A.M. Simper. An efficient method for calculating effective core potential integrals which involve projection operators. *Chem. Phys. Lett.*, 296:445, 1998.

[122] K. Aidas, C. Angeli, K.L. Bak, Vebjørn Bakken, R. Bast, L. Boman, O. Christiansen, R. Cimiraglia, S. Coriani, P. Dahle, E.K. Dalskov, U. Ekström, T. Enevoldsen, J.J. Eriksen, P. Ettenhuber, B. Fernández, L. Ferrighi, H. Fliegl, L. Frediani, K. Hald, A. Halkier, C. Hättig, H. Heiberg, T. Helgaker, A.C. Hennum, H. Hettema, E. Hjertenæs, S. Høst, I.M. Høyvik, M.F. Iozzi, B. Jansík, H.J. Jensen, D. Jonsson, P. Jørgensen, J. Kauczor, S. Kirpekar, T. Kjærgaard, W. Klopper, S. Knecht, R. Kobayashi, H. Koch, J. Kongsted, A. Krapp, K. Kristensen, A. Ligabue, O.B. Lutnæs, J.I. Melo, K.V. Mikkelsen, R.H. Myhre, C. Neiss, C.B. Nielsen, P. Norman, J. Olsen, J.M.H. Olsen, A. Osted, M.J. Packer, F. Pawlowski, T.B. Pedersen, P.F. Provasi, S. Reine, Z. Rinkevicius, T.A. Ruden, K. Ruud, V.V. Rybkin, P. Sałek, C.C..M. Samson, A.S. de Merás, T. Saue, S.P.A. Sauer, B. Schimmelpfennig, K. Sneskov, A.H. Steindal, K.O. Sylvester-Hvid, P.R. Taylor, A.M. Teale, E.I. Tellgren, D.P. Tew, A.J. Thorvaldsen, L. Thøgersen, O. Vahtras, M.A. Watson, D.J.D. Wilson, M. Ziolkowski, and H. Ågren. The Dalton quantum chemistry program system. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 4:269–284, 2014.