Inaugural - Dissertation

zur Erlangung der Doktorwürde der
Naturwissenschaftlich-Mathematischen Gesamtfakultät
der Ruprecht-Karls-Universität Heidelberg

The Algebraic Diagrammatic Construction Method For High Performance Computing Environments

Using an Atomic Orbital Representation

vorgelegt von

Maximilien Alexandre Ambroise
aus Differdingen, Luxemburg

Juli 2021

Gutachter:
Prof. Dr. Andreas Dreuw
...

Licensing

(empty)

Notation for Orbital Representations

μ, ν, γ, λ, ...	atomic orbitals
i, j, k, l, ...	occupied molecular orbitals
a, b, c, d, ...	virtual molecular orbitals
$\underline{i}$, $\underline{j}$, $\underline{k}$, $\underline{l}$, ...	local occupied molecular orbitals
$\overline{a}$, $\overline{b}$, $\overline{c}$, $\overline{d}$, ...	local virtual molecular orbitals
I, J, K, L, ...	occupied projected atomic orbitals
A, B, C, D, ...	virtual projected atomic orbitals

Contents

List of Figures

List of Tables

Listings

Introduction

This is the introduction. Talk about sparsity. Show sparse matrix. How does sparsity arise, what can we do with it, where else does it emerge? State of arts: adc now, adc in the future

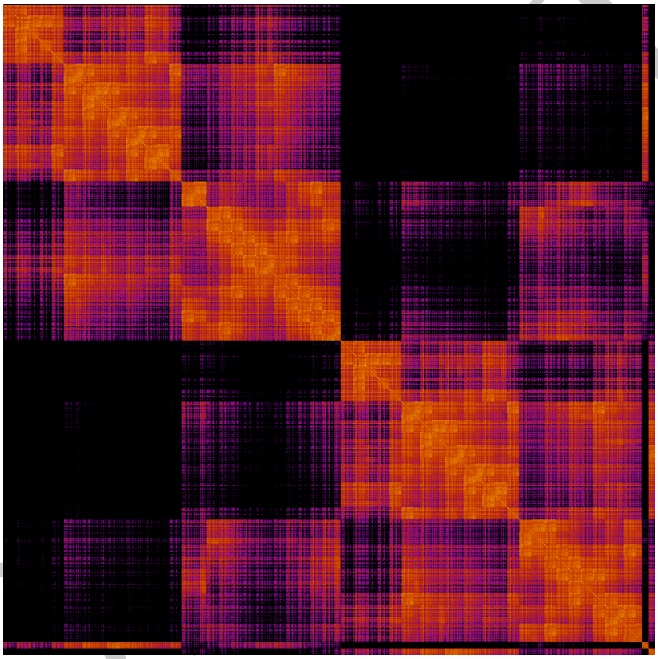


Figure 1: Adenine-guanine fock matrix Hartree FOck cc-pVTZ

Chapter 1

Theory: The Basics

Over the last decades, quantum chemistry has emerged as a crucial tool for investigating a wide variety of problems in chemistry. This, in combination with the increasing performance and widespread use of computers, has spawned a whole new branch of chemsitry known as *computational chemistry*. The field of computational chemistry uses methods developed in *theoretical chemistry* and incorporates them into efficient programs. Nowaydays, quantum chemical methods are routinely applied to assist in solving problems related to chemical structure, reactivity and spectroscopy. However, one of the main problems in computational chemistry is choosing a suitable level of theory for a given problem - the best choice is always a trade-off between speed and accuracy, and requires intricate knowledge of the methods' theoretical and computational limits. This chapter summarizes the core concepts of quantum chemsitry and introduces the most important methods used in modern computational chemistry. It is by no means complete, and the reader is referred to the original text book resources used by the author for more details (Szabo,Jensen,Helgaker,Schirmer,Norman).

1.1 Describing Dynamics in a Molecular System

In order to describe a molecular system, one needs to decide

- what the fundamental particles are
- what forces are governing them
- what the starting conditions are
- and what form the dynamical equations take.

The choice of particles dictates what properties the model is ultimately able to describe. For example, force field methods use atoms as the indivisible unit, which is sufficient to describe the molecular structure and dynamics of large molecules such as proteins, but does not provide any information on electron distribution. Using electrons and nuclei as the fundamental particles allows to get a better picture of the electron density and how it reacts to external perturbation, which is important for studies on reactivity and spectroscopic constants. To describe radioactive decay, it is necessary to further divide the nuleus into protons and neutrons.

Smaller subdivisions lead to a stricter limit on the size of molecules that can be treated. Force field methods may describe the dynamics of molecules with several tens of thousands of atoms, while a finer grained method involving electrons can often only desribe molecules one to two orders of magnitudes smaller depending on the approximantion used.

The mathematical form of the dynamical equations is determined by the size and mass of the particles. They can be divided into four regimes (Figure ...). Atoms are sufficiently heavy and slow that their trajectories can be described using classical (newtonian) mechanics. The time evolution of the positions $\mathbf{r}$ of the atoms in a potential $\mathbf{V}$ can be written as

$$-\frac{\partial \mathbf{V}}{\partial t} = m\frac{\partial^2 \mathbf{r}}{\partial t} \tag{1.1}$$

which is another form of Newton's second law $\mathbf{F} = m\mathbf{a}$. The potential $\mathbf{V}$ is also treated classicaly as the sum of contributions of particle-particle interactions.

For objects with velocities approaching the speed of light, it is necessary to introduce *relativistic effects*. Mass then becomes a function of velocity

$$m = \frac{m_0}{\sqrt{1 - v^2/c^2}} \tag{1.2}$$

Classical newtonian mechanics cannot be applied to very light particles, such as electrons, and quantum effects need to taken into considerations. For non-relativistic velocities, the dynamics are governed by the time-dependent Schrödinger equation:

$$\mathbf{H\Psi(r,t)} = \mathbf{i}\frac{\partial \mathbf{\Psi(r,t)}}{\partial \mathbf{t}} \tag{1.3}$$

where $\mathbf{H}$ is the Hamiltonian operator which is a sum of the kinetic and potential energy operators

$$\mathbf{H} = \mathbf{T} + \mathbf{V} \tag{1.4}$$

$$\mathbf{T} = -\frac{1}{2m}\nabla^2 \tag{1.5}$$

and Ψ is the wavefunction of the system, which is obtained as the solution to the Schrödinger Eqaution, and gives the probability of finding a particle at position $\mathbf{r}$ at time t. Here, atomic units are assumed.

For electrons moving at relativistic speed, for example in the core orbitals in super-heavy atoms, the Hamiltonian takes a more complex form, and the Schrödinger equation is then known as the *Dirac* equation:

$$\mathbf{H}_{Dirac} = \left(c\alpha\mathbf{p} + \beta mc^2\right) + \mathbf{V} \tag{1.6}$$

Here, only the solutions and aproximations to the non-relativistic Schrödinger equation will be discussed.

1.2 The Electronic Schrödinger Equation

> "Where did we get that [Schrödinger's equation] from? It's not possible to derive it from anything you know. It came out of the mind of Schrödinger."

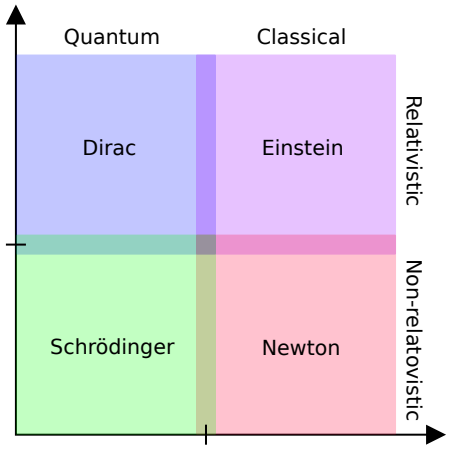


Figure 1.1: Adapted form (ref)

— *Richard Feynman, The Feynman Lectures on Physics*

If one is interested in describing the electron distribtion in detail, the Schrödinger Eqaution (SEQ) is the best starting point. There is no formal, rigorous proof for the Schrödinger equation, similar to how Newton's second law cannot really be "derived", more than simply "motivated" by observation.

As successful as the Schrödinger equation is, finding solutions to it is non-trivial. Different approximations may be applied to the SEQ to solve it more easily, without considerable loss of accuracy.

1.2.1 The Time-Independent Schrödinger Equation

The potential energy operator is the only time-dependent part of the Hamiltonian:

$$\mathbf{H}(\mathbf{r}, t) = \mathbf{T}(\mathbf{r}) + \mathbf{V}(\mathbf{r}, t) \tag{1.7}$$

For systems where the potential is time-independent, e.g. bound systems without external (electromagnetic) perturbation, the Hamiltonian is time-indepedent as well, which in turn allows to separate space and time variables. It can then be shown that the *time-independent* Schrödinger equation takes the form

$$\mathbf{H}\Psi(\mathbf{r}) = E\Psi(\mathbf{r}) \tag{1.8}$$

where E is the total energy of the system, and the eigenvalue of the wavefunction Ψ. The time-depedence is then simply reduced to a phase factor:

$$\Psi(\mathbf{r}, t) = e^{-iEt}\Psi(\mathbf{r}) \tag{1.9}$$

1.2.2 The Born-Oppenheimer Approximation

Atomic nuclei are much heavier than electrons ($m_{proton} \approx 1836 m_{electron}$), and move much slower. To a good approximation, the nuclei can be assumed to be stationnary from the point of view of electrons. This is known as the *Born-Oppenheimer approximation*. The

total Hamiltonian operator can be written in terms of the kinetic and potential operator of the nuclei (n) and electrons (e) as

$$\mathbf{H}_{tot} = \mathbf{T}_n + \mathbf{T}_e + \mathbf{V}_{ne} + \mathbf{V}_{ee} + \mathbf{V}_{nn} \tag{1.10}$$

In the Born-Oppenheimer approximation, the kinetic energy of the nuclei T_{nn} is neglected, and the nucleus-nucleus potential V_{nn} is taken as a constant, which corresponds to neglecting the coupling between electrons and nuclei. This allows a separation of the electronic and nuclar variables. The remaining terms of Equation ... form the electronic Hamiltonian $\mathbf{H_{elec}}$. The solutions to the *electronic Schrödinger eqaution*

$$\mathbf{H}_{elec}\Psi_{elec}(\mathbf{r}_i, \mathbf{R}_n) = E_{elec}(\mathbf{R}_n)\Psi_{elec}(\mathbf{r}_i, \mathbf{R}_n) \tag{1.11}$$

produce the electronic wavefunction which depends on the (fixed) *position* $\mathbf{R}_n$ of the nuclei and no longer on the *momentum* of the nuclei. The total energy

$$E_{tot}(\mathbf{R}_n) = E_{elec}(\mathbf{R}_n) + E_{nucl}(\mathbf{R}_n) \tag{1.12}$$

provides a *potential energy surface* (PES) on which the nuclei move. The PES can then be used to solve the nulcear Schrödinger equation to obtain information on vibrational, rotational and translational properties in the molecular system.

From this point onward, the subscript *elec* is dropped, and only the electronic Schrödinger Equation is considered.

1.3 Solutions to the Electronic Schrödinger Equation

1.3.1 Slater Determinants

It is first important to consider the wave function of the single electrons. In single-atom systems, these functions take the form of "atomic orbitals" (AOs). Correspondingly, "molecular orbitals" (MOs) are defined as the single electron wave functions in a molecular system. These spatial orbital functions form the basis of the total electronic wave function.

The Hamiltonian in ... only depends on the spatial coordinates. However, to fully describe an electron, one also needs to consider spin. This is done by introducing two orthonormal spin functions $\alpha(\omega)$ and $\beta(\omega)$ corresponding to spin-up ($\uparrow$) and spin-down ($\downarrow$), with the spin-coordinate ω. For one spatial molecular orbital, this gives two possible *spin-orbitals*

$$\phi(\mathbf{x}) = \begin{cases} \varphi(\mathbf{r})\alpha(\omega) \\ \varphi(\mathbf{r})\beta(\omega) \end{cases} \tag{1.13}$$

where φ are the *spatial orbitals*, and $\mathbf{x}$ are the combined spatial and spin coordinates. The spin orbitals therefore depend on four variables.

To a first approximation, one may consider a molecular system to consist of N *non-interacting*, independent electrons. The Hamiltonian is then written as a sum of one-particle Hamiltonians

$$\mathbf{H} = \sum_{i}^{N} \mathbf{h}_i \tag{1.14}$$

Electron correlation may then be included in some average way by using *effective* one-electron Hamiltonians, which is the basic working idea of the *Hartree* method. The solution to the SEQ can then be expressed as a product of the one electron wave functions

$$\Psi^{HP}(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N) = \phi(\mathbf{x}_1)\phi(\mathbf{x}_2)...\phi(\mathbf{x}_N) \tag{1.15}$$

which is also known as *Hartree product*.

However, the Hartree product does not take into account the *indistinguishability* of electrons. In what is known as the antisymmetry principle, which is a generilization of the Pauli exclusion principle, the wave function needs to fulfill

$$\Psi(\mathbf{x}_1, \mathbf{x}_2) = -\Psi(\mathbf{x}_2, \mathbf{x}_1) \tag{1.16}$$

upon exchange of any two electrons in the system. This is most easily achived by using *Slater determinants* (SD). For a molecular system consisting of N electrons distributed over N spin orbitals ϕ_i, the SD takes the form

$$\Psi_{SD}(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N) = \frac{1}{\sqrt{N!}} \begin{vmatrix} \phi_I(\mathbf{x}_1) & \phi_J(\mathbf{x}_1) & \ldots & \phi_P(\mathbf{x}_1) \\ \phi_I(\mathbf{x}_2) & \phi_J(\mathbf{x}_2) & \ldots & \phi_P(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_I(\mathbf{x}_N) & \phi_J(\mathbf{x}_N) & \ldots & \phi_P(\mathbf{x}_N) \end{vmatrix} \tag{1.17}$$

Or, using the diagonal of the SD as a short-hand notation

$$\Psi_{SD}(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N) = |\phi_I(\mathbf{x}_1), \phi_J(\mathbf{x}_1), \ldots, \phi_K(\mathbf{x}_1)\rangle \tag{1.18}$$

1.3.2 The Fock Space

A more generalized representation of the space of the antisymmetrized electron wave functions can be achieved by introducing the concept of occupation number (ON) vectors in the context of second quantization (Annex). In a system with M possible states (in this case spin molecular orbitals), the ON vectors take the form

$$|\mathbf{k}\rangle = |k_1, k_2, \ldots, k_M\rangle = \begin{cases} 1 if\ \phi_P\ occupied \\ 0 if\ \phi_P\ occupied \end{cases} \tag{1.19}$$

The occupation number is 1 if ϕ_P is present in the SD, and 0 if not. Together, all possible ON vectors in Equation ... form an orthonormal abstract vector space, known as Fock space. The Fock space formed by N electrons distributed over M orbitals is denoted as $F(M, N)$ with total dimension equal to the binomial coefficient $\begin{bmatrix} M \\ N \end{bmatrix}$. The sum of the occupation numbers in the ON vectors gives the total number of electrons

$$N = \sum_i^M k_i \tag{1.20}$$

The special fock space $F(0, M)$ contains a single vector known as the vacuum state with

$$|vac\rangle = |0_1, 0_2, \ldots, 0_M\rangle \tag{1.21}$$

$$\langle vac|vac\rangle = 1 \tag{1.22}$$

The ON vectors in $F(M,N)$ can be alternatively expressed in terms of the vaccum state from $F(M,0)$ using creation operators

$$|\mathbf{k}\rangle = \left[\prod_{P=1}^{M} (a_P^\dagger)^{k_P}\right] |vac\rangle \tag{1.23}$$

In second quantization, the antisymmetry principle of the wavefunction is guaranteed by the commutator relationship of the annihilation and creation operators a_P and $a_P^\dagger$ which act on the ON vectors.

1.3.3 Exact Solution and Standard Models

The simplest approach to solving the electronic SEQ is by approximating the exact wave function Ψ using a single Slater determinant where the electrons occupy the lowest lying molecular orbitals. The Hartree-Fock method is an example of a *single-determinant method* and finds *the* single best Slater determinant for $|\Psi\rangle$. In Fock space, the best possible determinant is represented by the ON vector where the N lowest lying orbitals are occupied.

As will be discussed in more detail later, a single-determinant treatment of the electronic wavefunction is insufficient to fully capture *electron correlation*. The electron correlation energy is formerly defined as the difference between the Hartree-Fock energy and the *exact* energy of $|Psi\rangle$

$$E_{correlation} = E_{HF} - E_{exact} \tag{1.24}$$

although the Hartree-Fock wave function does include correlation effects to some degree. In a more general sense, electron correlation is a broad term for any interactions between electrons that make their movement depend on each other, or *correlate* with each other (cf Section ...).

In order to improve on the HF approximation, it is important to add additional Slater determinats. These SDs can be generated from the HF reference wave function by replacing the occupied MOs ϕ_I in a reference Slater determinant ... by one or multiple orbitals ϕ_A which were previously unoccupied. This effectively corresponds to exciting one or more electrons from their occupied molecular orbitals $I, J, ..$ to unoccupied, or *virtual* orbitals $A, B, ..$ (Figure ...). These excited Slater determinants can be classified by the number of electrons they excite and are often referred to as singles, doubles, triples and so on.

$$\begin{aligned}
|S\rangle &= a_A^\dagger a_I \,|HF\rangle \\
|D\rangle &= a_A^\dagger a_I a_B^\dagger a_J \,|HF\rangle \\
|T\rangle &= a_A^\dagger a_I a_B^\dagger a_J a_C^\dagger a_K \,|HF\rangle \\
&\dots
\end{aligned}$$

Alternatively, singles states are known as 1-particle-1-hole (1p-1h or simply p-h) states, doubles as 2-particle-2-hole (2p-2h) and so on, due to the excitation operators effectively

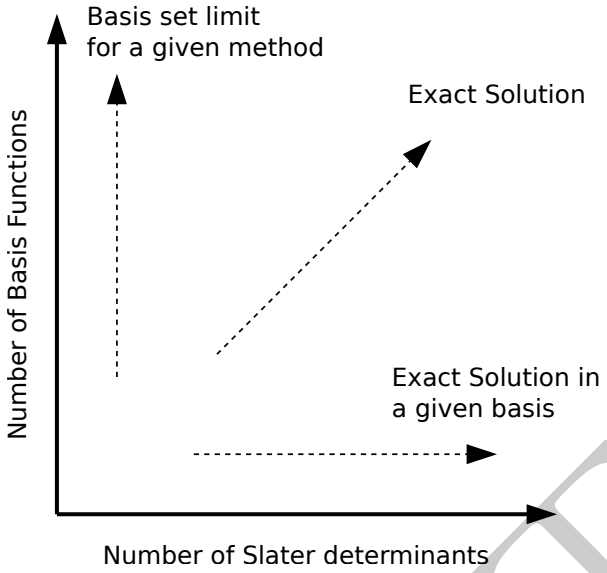


Figure 1.2: Standard model

creating *holes* in the reference determinant and adding *particles* in higher lying orbitals instead.

The exact solution to the electronic Schrödinger equation is then given by the sum of the Hartree Fock wave function and all possible ON vectors in $F(M, N)$

$$|\Psi\rangle = c_0 |HF\rangle + \sum_{i=1}^{\binom{M}{N}-1} c_i |i\rangle \tag{1.25}$$

Equation ... offers a systematic approach to improve on the Hartree Fock method, by increasing (1) the number of Sater determinats and (2) the basis set size M, and gives rise to a hierarchy of methods. Correlated electronic structure methods mainly differ by how they determine the expansion coefficients c. For all but the smallest systems, the full $F(M, N)$ cannot be used in calculations due to the total number of ON vectors which increases binomially with $\binom{M}{N}$, and hence the number of coefficients to be determined. In practice, the Fock space is *truncated* to some degree.

Multi-determinant methods like configuration interaction (CI) or coupled cluster (CC) use the Hartree Fock wave function as the reference from which they generate singles, doubles, triples ... SDs as in Eqaution .... By truncating at different excitation levels, one gets a hierarchy of CI/CC methods which recover different amounts of correlation energy (e.g. CIS, CISD, CISDT ...). Multi-determinant methods are mainly suited to recover dynamic correlation (cf. ...).

For systems with strong static correlation, additionally, a multi-reference approach is needed. Here, the excited SDs are generated from multiple reference states rather than only from HF. Methods include multireference CI (MRCI) and multireference CC (MRCC). The reference states are traditionally obtained from multiconfigurational self-consistent-field methods (MCSCF) like the complete active space SCF (CASSCF) or restrcited active space SCF (RASSCF). MCSCF is a combination of HF and CI, which both optimizes the HF molecular coefficients and the CI expansion coeffcients. Multi-

reference methods methods mainly recover *static correlation* (cf. Section ...). (Note: in limit, both static and dynamic recovered. No strict division between them)

A summary of all *ab-initio* standard models is given in Figure ...

Different Route: Moller plesset

1.3.4 The Variational Method

The time-independent Schrödinger equation takes the form of an eigenvalue problem

$$\mathbf{H}\,|\Psi_i\rangle = E_i\,|\Psi_i\rangle \qquad i = 0, 1, 2, ...\infty \tag{1.26}$$

where the infinite set of exact solutions $|\Psi_i\rangle$ with eigenvalues E_i forms an orthonormal basis

$$\langle\Psi_i|\Psi_j\rangle = \delta_{ij} \tag{1.27}$$

A trial wavefunction can be expanded in the basis of exact solutions with coefficients c as

$$|\widetilde{\Psi}\rangle = \sum_i a_i\,|\Psi_i\rangle \tag{1.28}$$

The *variation principle* states that the expectation value of the Hamiltonian of an approximate wave function of the form 1.28 is an upper bound to the exact ground state energy. This statement can be expressed as

$$\frac{\langle\widetilde{\Psi}|\,\mathbf{H}\,|\widetilde{\Psi}\rangle}{\langle\widetilde{\Psi}|\widetilde{\Psi}\rangle} \geq E_0 \tag{1.29}$$

The equality only holds when $|\widetilde{\Psi}\rangle$ is equal to the exact solution $|\Psi_0\rangle$. One can recast the eigenvalue problem 1.26 as a variational optimization problem where the energy is a functional of a trial wavefunction

$$E[\widetilde{\Psi}] = \frac{\langle\widetilde{\Psi}|\,\mathbf{H}\,|\widetilde{\Psi}\rangle}{\langle\widetilde{\Psi}|\widetilde{\Psi}\rangle} \tag{1.30}$$

The saddle points of the energy functional then correspond to the exact solutions of the Schrödinger equation. The variational approach to the SEQ provides a powerful tool to solve a wide variety of problems in electronic structure theory.

The trial wave function depends on a set of coefficients c, and hence the energy functional will also depend on these coefficients. In general, determining the coefficients which minimize the functional is very difficult. However, a more simple approach of the variational method can be obtained by using a linear ansatz where the trial function is expanded in a fixed N-dimensional set of orthonormal basis functions $|\phi\rangle$

$$|\widetilde{\Psi}\rangle = \sum_i^N c_i\,|\phi_i\rangle \tag{1.31}$$

By using Lagrange's method of undetermined multipliers

$$\mathcal{L}(\mathbf{c}, E) = \langle\widetilde{\Psi}|\,\mathbf{H}\,|\widetilde{\Psi}\rangle - E(\langle\widetilde{\Psi}|\widetilde{\Psi}\rangle - 1) \tag{1.32}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{c}} = 0 \tag{1.33}$$

it is possible to show that the variational problem corresponds to solving the eigenvalue problem involving the Hamiltonian matrix **H**:

$$\mathbf{H}\mathbf{c}_n = E_n \mathbf{c_n} \tag{1.34}$$

Or, in a more general form

$$\mathbf{HC} = \mathbf{CE} \tag{1.35}$$

Here, **C** is a N by N coeffcient matrix containing N column coefficient vectors $\mathbf{c}_n$ ($n = 0...N$) which describe N possible solutions for the trial wave function $|\widetilde{\Psi}\rangle$. **E** is a diagonal matrix containing the eigenvalues E_n. This approach of finding approximate solutions to the eigenvalue problem 1.26 is known as the *linear variational method*.

1.4 Hartree Fock

The Hartree Fock method is central to electronic structure method. It is computationally inexpensive, and is still routinely used in qualititative studies of large molecules, even if it does not accurately account for electron correlation. It also serves as the starting point for correlated methods. Only a few computational methods actually bypass the solution to the Hartree Fock equations, firmly cementing its place in quantum chemsitry.

1.4.1 The Hartree Fock Equations

Recall the structure of the electronic Hamiltonian

$$\mathbf{H} = \mathbf{T}_e + \mathbf{V}_{ne} + \mathbf{V}_{ee} + \mathbf{V}_{nn} \tag{1.36}$$

with

$$\mathbf{T}_e = -\sum_i^N \frac{1}{2}\nabla_i^2 \quad \text{kinetic energy of electrons} \tag{1.37}$$

$$\mathbf{V}_{ne} = -\sum_a^{N_{nuc}} \sum_i^N \frac{Z_a}{\mid \mathbf{R}_a - \mathbf{r}_i \mid} \quad \text{nuclei-electron repulsion} \tag{1.38}$$

$$\mathbf{V}_{ee} = \frac{1}{2}\sum_i^N \sum_{j\neq i}^N \frac{1}{\mid \mathbf{r}_i - \mathbf{r}_j \mid} \quad \text{electron-electron repulsion} \tag{1.39}$$

$$\mathbf{V}_{nn} = \frac{1}{2}\sum_a^{N_{nuc}} \sum_{b\neq a}^{N_{nuc}} \frac{Z_a Z_b}{\mid \mathbf{R}_a - \mathbf{R}_b \mid} \quad \text{nuclei-nuclei repulsion} \tag{1.40}$$

In Hartree-Fock theory, the electrons are treated as independent particles. One can therefore ignore the coupling between electrons in the $\mathbf{V}_{ee}$ and express the Hamiltonian as a sum of an effective one-electron operator **f**, also known as the *Fock* operator, of the form

$$\mathbf{H} = \sum_i \mathbf{f}_i = \sum_i \mathbf{h}_i + \frac{1}{2}\sum_i \sum_{j\neq i} \mathbf{g}_i j \tag{1.41}$$

$$\mathbf{h}_i = -\frac{1}{2}\nabla_i^2 - \sum_a^{N_{nuc}} \frac{Z_a}{\mid \mathbf{R}_a - \mathbf{r}_i \mid} \tag{1.42}$$

$$\mathbf{g}_{ij} = \frac{1}{\mid \mathbf{r}_i - \mathbf{r}_j \mid} \tag{1.43}$$

The one-electron operator $\mathbf{h}$, also known as the *core* Hamiltonian describes the movement of the electrons in the field of the nuclei. The two-electron operator $\mathbf{g}_{ij}$ gives the average potential (or "field") experienced by electron i due to the presence of all the other electrons j. For this reason, Hartree-Fock is also known the *mean-field* approximation. In second quantization, the fock operator takes the form

$$\mathbf{f} = \sum_{PQ}^{M} h_{PQ} a_P^\dagger a_Q + \frac{1}{2} \sum_{PQRS}^{M} g_{PQRS} a_P^\dagger a_R^\dagger a_S a_Q \tag{1.44}$$

with the matrix elements of the one and two electron operators given by

$$h_{PQ} = \langle \phi_P | \, \mathbf{h} \, | \phi_Q \rangle = \int \phi_P^*(\mathbf{x}) h(\mathbf{x}) \phi_Q(\mathbf{x}) d\mathbf{x} \tag{1.45}$$

$$g_{PQRS} = \langle PQ | \, RS = \int\int \phi_P^*(\mathbf{x}_1) \phi_R^*(\mathbf{x}_2) g(\mathbf{x}_1, \mathbf{x}_2) \phi_Q(\mathbf{x}_1) \phi_S(\mathbf{x}_2) d\mathbf{x}_1 d\mathbf{x}_2 \tag{1.46}$$

The elements g_{PQRS} are known as the 2-electron repulsion integrals. Calculating the expectation values for the fock operator in 1.44 using second quantization gives (ref):

$$\begin{aligned} f_{PQ} &= \langle \phi_P | \, \mathbf{f} \, | \phi_Q \rangle \\ &= h_{PQ} + \sum_I^N \frac{1}{2}(g_{PQII} - g_{PIIQ}) \\ &= h_{PQ} + \frac{1}{2}(J_{PQ} - K_{PQ}) \end{aligned} \tag{1.47}$$

The symmetric matrix with entries f_{PQ} is also known as the *Fock matrix*. $\mathbf{J}$ is the *coulomb matrix* and describes electron correlation due to the coulomb potential (coulomb correlation), and $\mathbf{K}$ is the exchange matrix describing the electron correlation which arises due to the Pauli exclusion principle (Fermi correlation). The exchange contributions have no classical counterpart and arise purely from quantum mechanical considerations.

In the special basis where the Fock matrix is diagonal

$$f_{PQ} = \delta_{PQ} \epsilon_P \tag{1.48}$$

the one-electron eigenfunctions of the Fock operator

$$\mathbf{f} \, | \phi_P \rangle = \epsilon_P \, | \phi_P \rangle \tag{1.49}$$

are known as the *canonical molecular spin orbitals*, and the eigenvalues are the *molecular orbital energies*. Solving the *Hartree-Fock equation* 1.49 gives the MOs which form the basis of Hartree-Fock wavefunction. It should be stressed that the total electronic Hartree

Fock energy is not the sum of the individual MO energies, but is given by the expectation value of the Hamiltonian

$$E_{HF} = \langle HF| \mathbf{H} |HF\rangle = \sum_{I}^{N} h_{II} + \frac{1}{2}\sum_{I}^{N} (J_{II} - K_{II}) \tag{1.50}$$

where the sum runs over the *occupied* orbitals I. For N electrons distributed over M MOs, there are N occupied orbitals with $\epsilon_I < 0$ and $M - N$ virtual orbitals with $\epsilon_A > 0$.

1.4.2 The Basis Set Approximation

Up until this point, the electronic wavefunction was constructed from Slater determinants of molecular spin orbitals. Virtually all applications use a basis set expansion to express the unknown MOs in terms of known functions, conventially called *atomic orbitals*. Any type of function can be used, e.g. exponentials, gaussians, polynomials or plane waves. Exponentials and gaussians are best suited to describe bound systems. The molecular orbitals are then expressed as a *linear combination of atomic orbitals* (LCAO)

$$|\phi_i\rangle = \sum_{i}^{M_{basis}} c_{i\mu}\chi_\mu \tag{1.51}$$

The basis set approximation is *exact* for an infinite number of basis functions. For more details, see ...

1.4.3 Working Equations for Restricted and Unrestricted Hartree-Fock

For reasons of efficient implementation, it is useful to separate out different electron spin components. The Fock matrix has four spin blocks: $F_{\alpha\alpha}$, $F_{\alpha\beta}$, $F_{\beta\alpha}$ and $F_{\beta\beta}$. The Fock matrix in the canonical basis is diagonal, and therefore only the diagonal blocks $F_{\alpha\alpha}$ and $F_{\beta\beta}$ are important. Introducing the notation $\overline{I}$ for MOs with spin σ', and I with opposite spin σ, the matrix elements of a spin block are given by

$$\begin{aligned} f^\sigma_{PQ} &= h^\sigma_{PQ} + \frac{1}{2}\left\{ \sum_{PQ}^{N_\sigma}\sum_{I}^{N_\sigma} (PQ \mid II) - (PI \mid QI) - \sum_{PQ}^{N_\sigma}\sum_{I}^{N_{\sigma'}} \left(PQ \mid \overline{JJ}\right) - \left(P\overline{J} \mid Q\overline{J}\right) \right\} \\ &= h^\sigma_{PQ} + \sum_{PQ}^{N_\sigma} J^\sigma_{PQ} - K^\sigma_{PQ} + J^{\sigma'}_{PQ} - K^{\sigma'}_{PQ} \end{aligned} \tag{1.52}$$

The opposite spin block $f^{\sigma'}_{PQ}$ is obtained by substituting indices with a bar by indices without a bar and vice-versa. Spin separation yields two coupled sets of equations for alpha and beta MOs

$$\begin{aligned} \mathbf{f}^\alpha |\phi^\alpha_I\rangle &= \epsilon^\alpha_I |\phi^\alpha_I\rangle \\ \mathbf{f}^\beta \left|\phi^\beta_I\right\rangle &= \epsilon^\beta_I \left|\phi^\beta_I\right\rangle \end{aligned} \tag{1.53}$$

These are known as the unrestricted Hartree-Fock equations (UHF). For closed-shell molecules with equal number of alpha and beta elecrons, the spatial part of the MOs is the same for both spins. The expression for the Fock matrix then further simplifies to

$$f_{ij} = h_{ij} + 2J_{ij} - K_{ij} \tag{1.54}$$

$$\mathbf{f}\,|\phi_i\rangle = \epsilon_i\,|\phi_i\rangle \tag{1.55}$$

The equations in 1.57 are known as the restricted Hartree-Fock (RHF) equations.
Using the linear variatonal method explained in the previous section for the MO trial functions expressed as a linear combination of N_{bas} AO basis functions, the eigenvalue problem for RHF can be recast in matrix form as

$$\mathbf{FC} = \mathbf{CE} \tag{1.56}$$

with the MO coefficient matrix $\mathbf{C}$ and the Fock matrix $\mathbf{F}$ in the AO basis given by

$$F_{\mu\nu} = H_{\mu\nu}^{core} + \sum_{\lambda\sigma}^{N_{basis}} \left(2\,(\mu\nu \mid \sigma\lambda)\,P_{\lambda\sigma} - (\mu\sigma \mid \nu\lambda)\,P_{\lambda\sigma}\right) \tag{1.57}$$

$$= H_{\mu\nu}^{core} + 2J_{\mu\nu} - K_{\mu\nu} \tag{1.58}$$

The symmetric matrix $\mathbf{P}$ is the so-called atomic orbital density matrix (DM) of the form

$$P_{\mu\nu} = \sum_{i}^{N_{occ}} C_{\mu i} C_{\nu i} \tag{1.59}$$

A similar expressions is found for UHF

$$\begin{aligned} F_{\mu\nu}^{\sigma} &= H_{\mu\nu}^{core} + \sum_{\lambda\sigma}^{N_{basis}} \left((\mu\nu \mid \sigma\lambda)\,P_{\lambda\sigma}^{T} - (\mu\sigma \mid \nu\lambda)\,P_{\lambda\sigma}^{\sigma}\right) \\ P_{\mu\nu}^{T} &= P_{\mu\nu}^{\sigma} + P_{\mu\nu}^{\sigma'} \end{aligned} \tag{1.60}$$

1.4.4 The Self-Consistent Field Method

In general, the atomic orbitals are not orthogonal. Defining the overlap matrix $\mathbf{S}$ with entries

$$S_{\mu\nu} = \int \chi_{\mu}^{*}(\mathbf{r})\chi_{\nu}^{*}(\mathbf{r})d\mathbf{r} \tag{1.61}$$

where $S_{\mu\mu} = 1$, the eigenvalue problem for RHF takes the more general form

$$\mathbf{FC} = \mathbf{SCE} \tag{1.62}$$

The equations in 1.65 are known as the *Roothan equations*. In the unrestricted case, they are called the *Pople-Nesbet equations*:

$$\mathbf{F}^{\alpha}\mathbf{C}^{\alpha} = \mathbf{S}\mathbf{C}^{\alpha}\mathbf{E}^{\alpha} \tag{1.63}$$

$$\mathbf{F}^{\beta}\mathbf{C}^{\beta} = \mathbf{S}\mathbf{C}^{\beta}\mathbf{E}^{\beta} \tag{1.64}$$

1.5 Electron Correlation

1.6 Post-Hartree Fock Ground State

1.6.1 Configuration Interaction

1.6.2 Perturbation Theory

1.6.3 Coupled Cluster

1.7 Post-Hartree Fock Excited State

1.7.1 Coupled Cluster Linear Repsonse

1.7.2 Equation-of-Motion Coupled Cluster

1.7.3 Algebraic Diagrammatic Construction

Chapter 2

Local 0

While computational chemistry has manifested itself as a popular and widely used tool, its inherently steep scaling limits its applicibality to cheaper methods like DFT, orto small or middle sized molecules for post-Hartree Fock methods. Even Hartree Fock with O(N^3) expensive when comparing to the rest of the world of computer science (examples?). Very early on, with the development of CI methods, (Pulay) effort has been put into reducing the prefactor and computational complexity for QC methods. Example: One of the most time-consuming steps in Post-Hartree Fock is the formation of the molecular orbitals, e.g. formation of the OVOV block of the MO integral tensor as encountered in CC and MP2:

$$(ia \mid jb) = \sum_{b}^{vir} C_{\sigma b} \sum_{a}^{vir} C_{\nu a} \sum_{j}^{occ} C_{\lambda j} \sum_{i}^{occ} C_{\mu i} \, (\mu\nu \mid \lambda\sigma) \tag{2.1}$$

By efficiently refactoring the sums, the MO-AO integral transformation scales as O(N^4). There are two reasons for the large cost: first, and quite obviously, a rank-4 tensor scales very fast, hence O(N4), also becomes a bottle neck for tensor contractions as many indices. Secondly, for large basis sets with triple-zeta quality or higher, or basis sets with diffuse functions, the virtual orbital space is very large, and can be multiple times the size of the occupied space. Attempts to reduce scaling can be grouped into two groups: screening-based methods and domain-based methods. Screening-based methods recast existing equations into the AO basis and use the sparsity and fast decay between AOs to establish highly efficient screening algorithms to lower the scaling of integral transformation. Domain-based methods stay in the MO basis, but a localized one, and attempt to assign domains of virtual molecular orbitals to a single LMO or a pair of LMOs, to obtain a more compact representation. Other attempts at mitigating the cost of MO-AO transformation is to exploit the rank sparsity of the AO ERI tensor. Density fitting and Cholesky decompositions can refactor the ERI tensor into a product of two 3-dim tensors. Tensor Hypercontraction goes even further and decomposes into 4 2-dim tensor. Density does not inherently lower scaling of methods, but rather reduces the prefactor associated with integral transformation. In special cases, decomposition techniques allow a refactoring of the working equations into lower scaling. Examples include the coulomb part of the Fock-build (O3 to O2) and SOS-MP2, SOS-CC2 or SOS-ADC(2) (O5 to O4). Density fitting and local approximations can be combined, to yield the best of both world in what is known as local density fitting. All of the above methods have their fair share of problems, some more than others. We will first address principles of density fitting,

before looking at possible orbital representations, and how they can be used for reduced scaling. Also go to local density fitting, and finally how the methods are implemented for ground state (HF,MP2,CCSD) and excited state computations (CI, CCLR, ADC).

2.1 Sparsity in Electronic Structure Theory

Sparsity is a core concept in electronic structure theory. Many of the most commonly encountered matrices and tensors exhibit some form of sparsity.

2.1.1 Element-Wise Sparsity of Electron Integrals

Molecular electron integral evaluation can become prohibitively expensive for large systems, espcially the four-dimensional ERI tensor which formerly scales as $\mathcal{O}(N^4)$. It is therfore imperative to exploit the exponential decay of the GTO basis.

Consider a model system consisting of n hydrogen atoms arranged in a line, with a distance of 1 a_0 between one another, and a primitive 1s Gaussian function attached to each atom. Figure ... shows the scaling behaviour for the overlap and electron repulsion integrals of this system. A full line is used to show the number of total elements, while the dotted line represents the number of significant integrals with magitude > 1e-10. From observing both graphs, it becomes apparent that for increasing number of atoms, many of the electron integrals can be ignored. Therefore, one only needs to store integrals above a certain threshold. This is also known as *element-wise sparsity*.

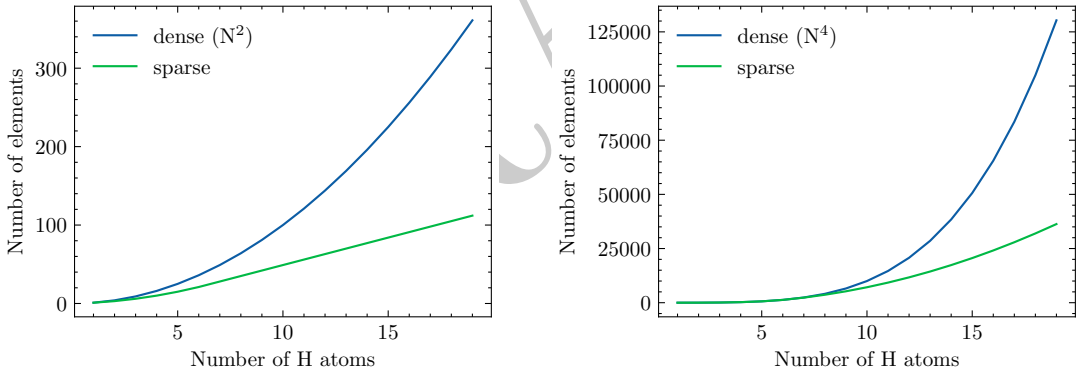
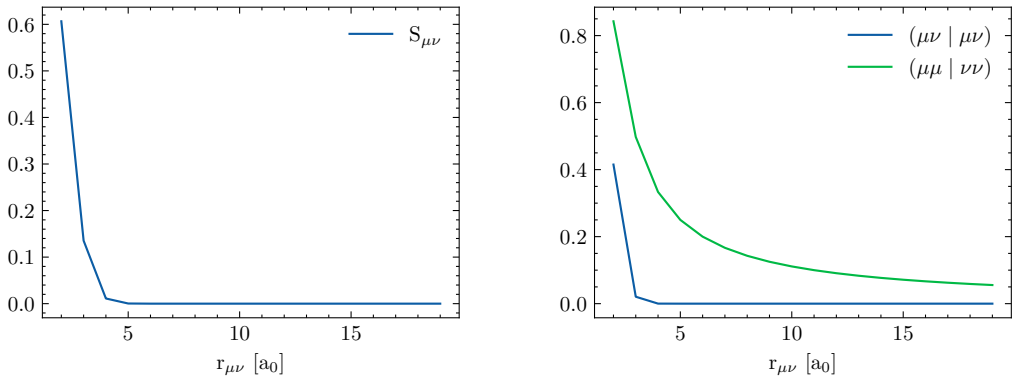


Figure 2.1: (a) Number of significant entries (full line) in the overlap matrix for a hydrogen atom chain, with a threshold of 1e-10. The dotted line shows the total number of elements for the dense matrix, which scale as N^2. (b) Number of significant entries (full line) in the electron repulsion integral tensor for a hydrogen atom chain, with a threshold of 1e-10. The dotted line shows the total number of elements for the dense tensor, which scale as N^4.

Linear Scaling Overlap Integrals

While the overlap integrals formely scale with $\mathcal{O}(N^2)$, it can be shown that the number of significant elements scales *linearly*. First, consider the product of two 1s GTOSs χ_A



(a) (b)

Figure 2.2: (a) Magnitude of the overlap integral between two Gaussian 1s orbitals as a function of distance r (exponential decay). (b) Magnitude of the electron repulsion integral betwwen two Gaussian 1s orbitals as a function of r. The short range interaction $(\mu\nu \mid \mu\nu)$ decays at a muchfaster rate with e^{-r^2}, compared to the long range interaction with $1/R$.

and χ_B, centred at $\mathbf{A}$ and $\mathbf{B}$, with exponents α and β. The Gaussian product theorem (GPT) states that the result is itself also a (scaled) Gaussian function

$$\chi(A,\alpha)\chi(B,\beta) = e^{-\alpha|\mathbf{r}-\mathbf{A}|^2} e^{-\beta|\mathbf{r}-\mathbf{B}|^2} = \kappa\chi(P,\alpha+\beta) \tag{2.2}$$

with the scaling factor κ

$$\kappa = e^{-\frac{\alpha\beta}{\alpha+\beta}|\mathbf{A}-\mathbf{B}|^2} \tag{2.3}$$

and the centre-of-charge coordinate P

$$\mathbf{P} = \frac{\alpha\mathbf{A}+\beta\mathbf{B}}{\alpha+\beta} \tag{2.4}$$

Spatioal integration yields the expression for the overlap between χ_A and χ_B

$$S_{AB} = \int \kappa\chi_P dr = \kappa\left(\frac{\pi}{\alpha+\beta}\right)^{3/2} \tag{2.5}$$

The magnitude of the overlap integral is proportional to the scaling factor κ which decays exponentially with the distance between GTO centres. In the case of the model system given above, where $\alpha = \beta$, the distance at which the integral falls below a certain threshold ϵ is given by

$$d_s = \sqrt{\alpha^{-1} ln\left[\left(\frac{\pi}{2\alpha}\right)^3 \epsilon^{-1/2}\right]} \tag{2.6}$$

Which in our case is equal to 6.9 a_0. Each hydrogen atom therefore only has significant overlap with a finite number n_{max} of other centres. For atom chains with $n > n_{max}$, the number of non-zero elements in the overlap matrix will no longer scale as n^2, but *linearly* with nn_{max}. For more realistic, three-dimnensional molecular systems, the crossover is less clearly defined due to the non-uniform distribution of atoms and different GTO exponents. Nonetheless, if a system grows sufficiently large, the overlap integrals still scale linearly. Similar arguments can be brought forth for the kinetic-energy integrals as well.

Quadratic Scaling Electron Repulsion Integrals

Using the Guassian product theorem established above, we can express the two-electron repulsion integrals of four primtive 1s Gaussian functions $s(A,\alpha)$, $s(B,\beta)$, $s(C,\gamma)$ and $s(D,\delta)$ as

$$\begin{aligned} g_{ABCD} &= \int s(A,\alpha)s(B,\beta)\frac{1}{|\mathbf{r_1}-\mathbf{r_2}|}s(C,\gamma)s(D,\delta)dr \\ &= \int \kappa s(P,\alpha+\beta)\frac{1}{|\mathbf{r_1}-\mathbf{r_2}|}\lambda s(Q,\gamma+\delta) \end{aligned} \tag{2.7}$$

where $s(P,p)$ and $s(Q,q)$ are Gaussian distributions with

$$\mathbf{P} = \frac{\alpha\mathbf{A}+\beta\mathbf{B}}{\alpha+\beta}; \quad \mathbf{Q} = \frac{\gamma\mathbf{C}+\delta\mathbf{D}}{\gamma+\delta} \tag{2.8}$$

$$\kappa = e^{-p|\mathbf{A}-\mathbf{B}|^2}; \quad \lambda = e^{-q|\mathbf{C}-\mathbf{D}|^2} \tag{2.9}$$

$$p = \frac{\alpha\beta}{\alpha+\beta}; \quad q = \frac{\gamma\delta}{\gamma+\delta} \tag{2.10}$$

The coulomb integrals can then be evaluated as

$$g_{ABCD} = \sqrt{\frac{4\eta}{\pi}}S_{AB}S_{CD}F_0\left(\eta\left|\mathbf{P}-\mathbf{Q}\right|^2\right) \tag{2.11}$$

with the Boys function F_0 and the reduced exponent η given by

$$\eta = \frac{pq}{p+q} \tag{2.12}$$

The Boys function is an important function appearing in many expressions for molecular integral evaluation. There are two expressions that bound the Boys function

$$\begin{aligned} F_n(x) &\leq \frac{1}{2n+1} \quad \text{for small } x \\ F_n(x) &\leq \frac{(2n-1)!!}{2^{n+1}}\sqrt{\frac{\pi}{x^{2n+1}}} \quad \text{for large } x \end{aligned} \tag{2.13}$$

Using the Boys function's upper bounds, we can derive an upper bound for the electron repulsion integrals of our model system

$$g_{ABCD} \leq min\left\{\sqrt{\frac{4\eta}{\pi}}S_{AB}S_{CD}, \frac{S_{AB}S_{CD}}{|\mathbf{P}-\mathbf{Q}|}\right\} \tag{2.14}$$

The left-hand upperbound represents the short-range limit of the Boys function, and the right-hand one the long-range limit. In the short-range limit, i.e. for increasing distance R_{AB} or R_{CD}, the magnitude of g decreases *expoentially*. As shown in the previous section, the non-zero elements of the overlap integrals S_{AB} and S_{CD} scale linearly with system size, and therefore the number of significant electron repulsion integrals scales with N^2 in total. It should be noted, that in the long-range limit whith increasing distance R_{PQ} between

product densities, the number of elements in g will eventually scale linearly. However, the *algebraic* $1/R$ decay of the long-range interactions is so slow that it practically useless for the size of moleculaes that can be tackled with current technologies. In the case of the hydrogen atom chain, the integrals $(\mu\mu \mid \nu\nu)$ only fall below 1e-10 for R_{PQ} greater than 10^{10} a_0. While the long-range decay is impractical for use in the case of the electron repulsion integrals, there are instances such as in AO-MP2 where *bra* and *ket* decay as $1/R^4$. Knowing that the electron repulsion integrals are sparse is only the first step. One also has to develop a *screening* method to avoid computing small integrals, by finding a general upperbound. It has been shown (Roo1951) that g is positive-definite, fullfilling the relationship

$$\sum_{abcd} c_{ab} g_{abcd} c_{cd} > 0 \tag{2.15}$$

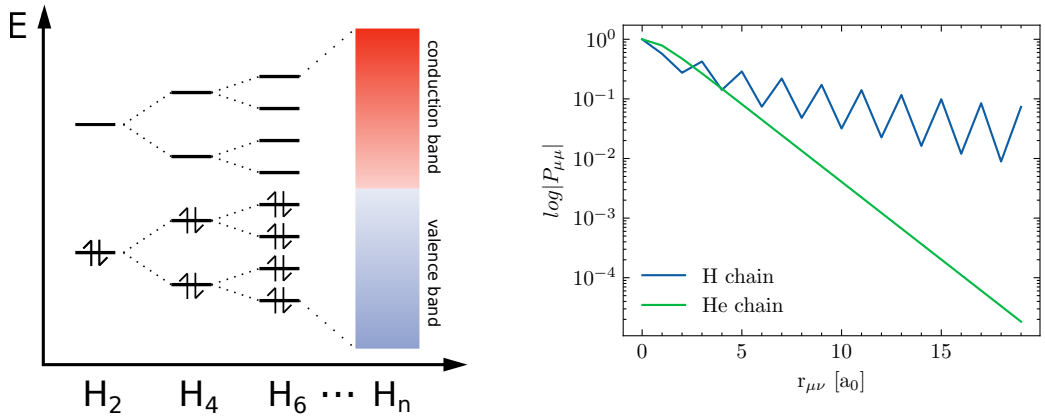
where c are one-electron orbital distrbutions. One can then apply the Schwarz inequality (Has1989) to obtain an upper bound expression for g

$$(\mu\nu \mid \sigma\lambda) \leq (\mu\nu \mid \mu\nu)\,(\sigma\lambda \mid \sigma\lambda) = Q_{\mu\nu} Q_{\sigma\lambda} \tag{2.16}$$

The matrix $\mathbb{Q}$ contains the square root of the short-range diagonal entries of g, and is also known as the Schwarz matrix. $\mathbf{Q}$ can be evaluated quickly with $\mathcal{O}(N^2)$ effort and inidividual integrals can be efficiently screened. It should be noted that Schwarz screening does not take into account the $1/R$ decay between product densities, which makes the method less useful in methods like AOMP2.

2.1.2 Element-Wise Sparsity of the Density Matrix

The decaying behaviour of the density matrix has been extensively studied in solids for atom-centred Bloch and Wannier functions. It was shown that for insulators, i.e. systems with large band-gaps, the contributions $P_{\mu\nu}$ decay exponentially with increasing distance $R_{\mu\nu}$, while for systems with small or no band gaps, such as metals, the elements decay algebraically. The same observations have been made for non-periodic systems using atomic orbitals as basis. For molecules with a large HOMO-LUMO gap, e.g. alkanes, the number of non-zero elements in the atomic orbital density matrix scales linearly with increasing system size. On the other hand, molecules with strong electron delocalization, such as conjugated polyenes, have have a small HOMO-LUMO gap, and the density matrix elements decay much slower. Consider again a chain of hydrogen atoms, equally spaced by a_0, each with one 1s Gaussian function, this time with N_{atom} atoms. Figure ... shows the MO diagram for increasing chain length. In the limit where $N_{atom} \to \infty$, the system takes on a band structure, similar to how they are encountered in a metal, with a smooth transition between occupied (valence) and virtual (conductance) band. In other words, the HOMO-LUMO gap becomes increasingly small. For hydrogen atom which each have one electron, the band is half filled, and the system is a conductor. If the Hydrogen atoms are replaced by Helium atoms, with two electrons per site, the band is fully filled and the system becomes an insulator. The magnitude of the density matrix elements $P_{\mu\nu}$ is plotted in Figure ... as a function of increasing distance between 1s functions. The elements decay much slower for the conducting hydrogen chain, while a rapid exponential decay can be observed in the case of the insulating helium chain.



(a) (b)

2.1.3 Diagrammatic Notation

Hollmann et al. (ref) have introduced a simple graphical representation to show contributing factors to the sparsity of a given matrix, tensor or tensor contraction. Each tensor index is represented as a vertex which contributes. Non-connected vertices each contribute $\mathcal{O}(N)$ elements to the overall expression. A sparsity relationship between two indices is represented as an *edge* connecting two vertices. In this case, the number of *pairs* scales as $\mathcal{O}(N)$. Consider the two electron integral tensor $(\mu\nu \mid \sigma\lambda)$. From the previous section, we know that the index pairs μ, ν and λ, σ are related by overlap. The diagrammtic representation takes the form:

$$\mu \overset{\mathrm{S}}{\longleftrightarrow} \nu \qquad \sigma \overset{\mathrm{S}}{\longleftrightarrow} \lambda$$

There are two pairs of connected vertices, which indicates that the integrals can be evaluated with $\mathcal{O}(N^2)$ effort, which is in agreement with the findings above. The S denotes the overlap relationship between vertices. For another example, consider the Hartree Fock expression for the exchange matrix

$$K_{\mu\nu} = (\mu\sigma \mid \nu\lambda)\, P_{\lambda\sigma} \tag{2.17}$$

Diagrammtically, the expression for $\mathbf{K}$ can be represented as

$$\mu \overset{\mathrm{S}}{\longleftrightarrow} \sigma \overset{\mathrm{P}}{\longleftrightarrow} \lambda \overset{\mathrm{S}}{\longleftrightarrow} \nu$$

The connection between σ and λ is also known as a "P-junction" (Nee2009), which represents the sparsity relationship arising due to the exponential decay of density matrix elements. The sparsity graph is fully connect, which suggests that $\mathbf{K}$ can be evaluated with $\mathcal{O}(N)$ effort. This is indeed the case, as shown by the ONX or LinK method. For linear scaling to emerge, indices of an expression therefore need to be fully linked. This simple but important fact is also known the linked index rule (LIR). Diagrams also show which factors can influence the performance of the scaling, such as diffuseness of the atomic orbitals (slower S decay) or size of of the HOMO-LUMO gap (slower P decay). One can therefore conclude that the expression for $\mathbf{K}$ as given above, is less suitable for large basis sets and non-insulators.

2.1.4 Rank Sparsity

A positive semi-definite matrix **A** has the property that it can be decomposed as a product

$$\mathbf{A} = \mathbf{B}\mathbf{B}^T \tag{2.18}$$

where **A** has dimensions N by N, and **B** has dimensions N by $rank(A)$. The rank represents the number of linearly independent column vectors in matrix, and for $rank(A) < N$, the matrix is said to be rank-deficient. The decomposition matrix **B** therefore is more compact and needs less storage space than **A**. There are different ways to compute **B**, such as Cholesky decomposition or QR decomposition. The tensor $(\mu\nu \mid \sigma\lambda)$ can be represented as a N_{AO}^2 by $N_{AO}^2 0$ matrix with combined row indices $I = \mu + N_{AO} * \nu$ and column indices $J = \sigma + N_{AO} * \lambda$. Because the tensor has been shown to be postive semi-definite, there also exists a decomposition, such that

$$(\mu\nu \mid \sigma\lambda) = A_{(\mu\nu)(\sigma\lambda)} = B_{(\mu\nu)X} B_{(\sigma\lambda)X} \tag{2.19}$$

The rank of **A** is in general much smaller than the combined index range N_{AO}^2, and scales linearly rather than quadratically with the number of basis sets. The decomposition tensor **B** is therefore 3-dimensional, rather than 4-dimensional, which reduces the storage needed by an order of magnitude from $\mathcal{O}(N^4)$ to $\mathcal{O}(N^3)$, but only in the case where $(\mu\nu \mid \sigma\lambda)$ is dense. In the limit of large molecules, the NZEs of **B** also scale with $\mathcal{O}(N^2)$. Rather than for the molecular integrals in the AO basis, decomposition techniques are more useful for reducing the storage size of molecular integrals in the canonical MO basis

$$(ia \mid jb) = C_{\mu i} C_{\sigma a} B_{\mu\sigma X} B_{X\nu\lambda} C_{\nu j} C_{\lambda b} = B_{iaX} B_{Xjb} \tag{2.20}$$

The AO-MO transformation step is also drastically sped up, but remains a $\mathcal{O}(N^4)$ effort. Rank sparsity has therefore little impact on the overall scaling, but rather reduces the scaling *prefactor*. Over the years, different methods have been proposed to compute **C**, such as density fitting, Cholesky decompostion, pseudo-spectral methods, or tensor hypercontraction. Density matrices at different levels of theory (Hartree Fock, MP2, CC ...) also exhibit rank sparsity. Decomposition of such matrices play an important role in local molecular orbital schemes and low scaling electronic structure methods, as will be shown in later sections.

2.2 Density Fitting

The method of choice in this thesis for the decomposition of two-electron molecular integrals is *density fitting* (DF), also knwon as *resoulation of the identity* (RI). For a brief exploration of other popular methods, the reader is referred to (ANNEX).

2.2.1 Basics of Density Fitting

The two-electron integrals can be expressed in terms of the charge product densities $\rho_{\mu\nu} = \chi_\mu \chi_\nu$ as

$$(\mu\nu \mid \sigma\lambda) = \int\int \frac{\rho_{\mu\nu}(\mathbf{r_1})\rho_{\sigma\lambda}(\mathbf{r_2})}{\mathbf{r_1} - \mathbf{r_2}} d\mathbf{r_1} d\mathbf{r_2} \tag{2.21}$$

The charge densities ρ can be approximated by fitting them to a set of atom-centred auxiliary functions χ_P

$$\rho_{\mu\nu}(\mathbf{r}) = C_{P\mu\nu}\chi_P(\mathbf{r}) + \Delta\rho_{\mu\nu} \tag{2.22}$$

Or in the chemist's notation:

$$|\mu\nu) = C_{P\mu\nu}\,|P) + |\epsilon_{\mu\nu}) = |\tilde{\mu\nu}) + |\epsilon_{\mu\nu}) \tag{2.23}$$

where $C_{P\mu\nu}$ are the fitting coefficients, and $\Delta\rho_{\mu\nu}$ or $|\epsilon_{\mu\nu})$ is the error introduced by the fitting procedure. Eq. ... is known as the density fitting approximation (Whi73,Bae1973). The two-electron integrals then take the form

$$\begin{aligned}(\mu\nu \mid \sigma\lambda) &= (\widetilde{\mu\nu}|\widetilde{\sigma\lambda}) + \underbrace{(\widetilde{\mu\nu}|\epsilon_{\sigma\lambda}) + (\epsilon_{\mu\nu}|\widetilde{\sigma\lambda})}_{\text{first order}} + \underbrace{(\epsilon_{\mu\nu} \mid \epsilon_{\sigma\lambda})}_{\text{second order}} \\ &= (\widetilde{\mu\nu}|\widetilde{\sigma\lambda}) + \epsilon_J^{(1)} + \epsilon_J^{(2)}\end{aligned} \tag{2.24}$$

Here, $\epsilon_J^(1)$ and $\epsilon_J^(2)$ represent the first order (linear) and second order (quadratic) error. The fitting coefficients are then generally found by minimizing $\epsilon_J^{(2)}$. Substituting $(\epsilon_{\mu\nu}| = (\mu\nu - \widetilde{\mu\nu}|$ gives

$$\frac{\partial}{\partial C^P_{\mu\nu}}\left(\mu\nu - \widetilde{\mu\nu} \mid \sigma\lambda - \widetilde{\sigma\lambda}\right) = 0 \tag{2.25}$$

which then yields a set of linear equations

$$(\mu\nu \mid P) - \sum_Q C^Q_{\mu\nu}\,(Q \mid P) = 0 \tag{2.26}$$

Finding the fitting coefficients by minimizing $\epsilon_J^{(2)}$ has the important feature that $eps_J^{(1)} = 0$, which can be shown by substituting Eq. ... back into Eq. ... . The total electron integral error is therefore *quadratic* in the fitting error. Fitting procedures where the coefficients $C^P_{\mu\nu}$ satisfy Eq. ... are termed *robust* (Dun2000). Any restrictions posed on $C^P_{\mu\nu}$ makes ϵ_1 diffent from zero and the error scales linearly. Eq. ... needs the evaluation of the three-centre-two-electron (3c2e) and two-centre-two-electron (2c2e) integrals in the auxiliary basis set $\{P\}$

$$(\mu\nu \mid P) = \int\int \chi_\mu(\mathbf{r_1})\chi_\mu(\mathbf{r_1})\frac{1}{\mathbf{r_1} - \mathbf{r_2}}\chi_P(\mathbf{r_2})d\mathbf{r_1}d\mathbf{r_2} \tag{2.27}$$

$$(P \mid Q) = \int\int \chi_P(\mathbf{r_1})\frac{1}{\mathbf{r_1} - \mathbf{r_2}}\chi_Q(\mathbf{r_2})d\mathbf{r_1}d\mathbf{r_2} \tag{2.28}$$

The fitting coefficients are generally computed by inverting $(P \mid Q)$, which leads to the following approximation for the four-centre-two-electron integrals

$$(\mu\nu \mid \sigma\lambda) \approx (\mu\nu \mid P)\,(P \mid Q)^{-1}\,(Q \mid \sigma\lambda) \tag{2.29}$$

Matrix inversion is a $\mathcal{O}(N^3)$ computational effort. For more details on precision and best practices involving matrix inversion, see ... .

2.2.2 Scaling of the 3c2e Integrals

Using the diagrammatic representation introduced earlier, the 3c2e integral tensor reduces to

$$\mu \overset{\mathrm{S}}{\longleftrightarrow} \nu \qquad P$$

The number of non-zero elements elements therefore scales as $\mathcal{O}(N^2)$, just like for the 4c2e integrals. Similarly, the Schwarz inequality can be used to screen out small integrals

$$|(\mu\nu \mid P)| \leq |(\mu\nu \mid \mu\nu)|^{1/2} \, |(P \mid P)|^{1/2} \tag{2.30}$$

As mentioned above, Schwarz screening does not take into account increasing bra-ket distance. The long-range decay is too slow to be of any advantage in the case of the 4c2e integrals. However, it was found (Hol2015) that for an auxiliary density $\chi_P(\mathbf{r}$ with angular momentum l_P, the 3c2e integrals actually decay as $1/R^{-1-l_P}$ with increasing bra-ket distance, establishing a weak sparsity relationship between $(\mu\nu|$ and $|P)$

$$\mu \overset{\mathrm{S}}{\longleftrightarrow} \nu \qquad \underset{1/R^{-1-l_P}}{P}$$

In principle, the 3c2e integrals can be evaluated with linear effort. Hollmann et al (Hol2015) have introduced a tight upperbound, known as the SVQl estimator, to exploit this faster decay. Due to the dependence on l_P, the screening is most effective with larger basis sets with high angular momentum functions.

The fitting coefficients evaluated as $C^P_{\mu\nu} = (\mu\nu \mid Q)\,(Q \mid P)^{-1}$ formerly scale with $\mathcal{O}(N^3)$

$$\mu \overset{\mathrm{S}}{\longleftrightarrow} \nu \qquad Q \qquad P$$

due to the inverse of $(P \mid Q)$ not being sparse.

2.2.3 Local Density Fitting: Principles

The long-range behaviour introduced by Eq. ... is often deemed "unphysical" (Tew2019). *Local density fitting* (LDF) methods circumvent this problem by forcing a more rapid decay of long-range contributions, either (a) by using a different metric in the fitting procedure Eq. ... or (b) by constructing domains $[\mu\nu]$ that exclude distant fitting functions P a priori. In both cases, Eq. ... is no longer fulfilled and the error in the electron integrals ... increases linearly with the fitting error, and the density fitting procedure is no longer robust. LDF methods therefore use a different expression for the electron integrals which includes the first order terms to remove the linear error

$$(\mu\nu \mid \sigma\lambda) \approx (\widetilde{\mu\nu}|\sigma\lambda) + (\mu\nu|\widetilde{\sigma\lambda}) - (\widetilde{\mu\nu}|\widetilde{\sigma\lambda}) \tag{2.31}$$

which is known as Dunlap's robust density fitting formula.

Type	Ref.	$g(r_{12})$
Overlap	[A]	1
Coulomb-Attenuated	[B]	$\frac{erfc(\omega r_{12})}{r_{12}}$
Yukawa	[C]	$\frac{e^{-\omega r_{12}}}{r_{12}}$
Gaussian-Damped	[D]	$\frac{e^{-\omega r_{12}^2}}{r_{12}}$

Table 2.1: Expressions for the operator g in different local metrics.

2.2.4 LDF (I): Short-Range Metrics

The first type of LDF methods replaces the fitting procedure in the Coulomb metric in Eq. ... by a more general expression

$$B^P_{\mu\nu} - C^Q_{\mu\nu} M_{QP} = 0 \tag{2.32}$$

where $B^P_{\mu\nu}$ and M_{PQ} are the 3-centre- and 2-centre-2-electron integrals given by

$$B^P_{\mu\nu} = \int\int \chi_\mu(\mathbf{r_1})\chi_\nu(\mathbf{r_1}) g(\mathbf{r_1},\mathbf{r_2})\chi_P(\mathbf{r_2}) d\mathbf{r_1} d\mathbf{r_2} \tag{2.33}$$

$$M_{PQ} = \int\int \chi_P(\mathbf{r_1}) g(\mathbf{r_1},\mathbf{r_2})\chi_Q(\mathbf{r_2}) d\mathbf{r_1} d\mathbf{r_2} \tag{2.34}$$

with g being the local metric. A list of known local metrics is given in Table ... . Earliest forms of density fitting actually first used an overlap metric to directly minimizing the norm of the residual $R_{\mu\nu} = (\mu\nu| - (\widetilde{\mu\nu}|$ using linear least squares (Baer), and the fitting coefficients are computed as

$$C^P_{\mu\nu} = S^{-1}_{PQ}(\mu\nu Q) \tag{2.35}$$

where S_{PQ} is the overlap in the auxiliary basis, and $(\mu\nu Q)$ are the 3-centre-**1-electron** overlap integrals. While the overlap metric has the most rapid decay and the quantities in Eq. can be evaluated in $\mathcal{O}(N)$ time, it has the worst accuracy of all metrics. One solution to this problem is to introduce a metric which is intermediate between overlap and coulomb fitting. Examples include the Yukawa, Coulomb- and Gaussian-attenuated metrics (Table ...). These intermediate metrics introduce a damping factor ω to control the sparsity and accuracy of the density fit. In the limit where $\omega \to 0$, and $\omega \to \infty$, one recovers the coulomb and overlap metric, respectively. Figure ... shows the decay behaviour of a local metric, using the Coulomb-attenuated metric as an example, for ω = 0.01, 0.1 and 1.0, compared to the overlap and the Coulomb metric. ...

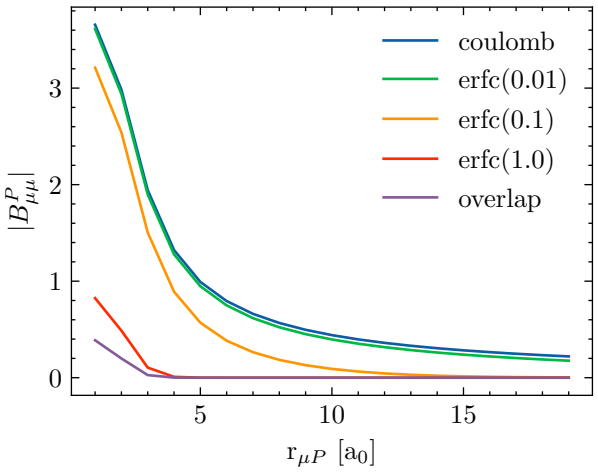


Figure 2.4: Some caption

2.2.5 LDF (II): Local Domains

The second method to force locality in density fitting consists in constructing local domains for each atom, pair of atoms or molecular orbital, and excluding any auxiliary functions that lie outside, which can drastically reduce the dimension of the fitting procedure.

Atomic Resolution of the Identity

The simplest example of a domain is one that includes a single atom. The atomic resolution of the identity (ARI) [] uses a fitting procedure where the sum over auxiliary function Q only includes those which are centred on the same atom A as the atomic orbital μ

$$|\widetilde{\mu\nu}) = \sum_{Q \cup A_\mu} (P \mid Q)^{-1}_{A_\mu} (\mu\nu \mid Q) \tag{2.36}$$

Each atom X has its own metric matrix inverse $(P \mid Q)^{-1}_X$ which takes the form

$$(P \mid Q)^{-1}_X = B_X \left((P \mid Q)_D + B_X (P \mid Q)_{OD} B_X \right)^{-1} B_X \tag{2.37}$$

where $(P \mid Q)_D$ and $(P \mid Q)_{OD}$ are the diagonal and off-diagonal part of $(P \mid Q)$ respectively. B_X is a so-called *bump matrix* which imposes a fast, but smooth decay between functions P and Q in order to avoid using all functions P for the fitting in Eq. ... . For further details, the reader is referred to the orginal publication. The bump matrix uses multiple distance criteria which make the ARI less of a black-box method.

Pair-Atomic Resolution of the Identity

A more popular and simple variant of atomic density fitting is the pair-atomic resolution of the identity (PARI) method (Mer2013). As the name implies, the domains include atom *pairs* rather than a single atom. Again expressing it in terms of the fitting procedure

$$(P \mid \mu\nu) = \sum_{Q \in A \cup B} (P \mid Q)\, C^Q_{\mu\nu} \quad \forall P \in A \cup B \tag{2.38}$$

The number of linear equations is equal to the number of non-zero pairs $\mu\nu$, which scales linearly. However, the PARI approach poses heavy constraints on the fitting coefficients, which leads to large integral errors. Merlot et al. proposed to increase the atomic pair domain with any atoms which lie between A and B. Alternatively, larger and more diffuse basis sets can be used. In both cases, performance is sacrificed for increased accuracy. The abesence of any distance dependent parameters or thresholds still make it an attractive method both for Hartree Fock and Post-Hartree Fock methods (refs).

LDF using Local Molecular Orbitals

Finally, domains can also be formed using local molecular orbitals instead of AOs. LMOs are larger than AOs, but are still generally centred on only a few atoms. The exact atomic sites can be determined for example by using a Mulliken population analysis. Consider the density fitting procedure as proposed by Polly et al. for their LDF-Hartree Fock method (pol2004)

$$(\mu i \mid P) = \sum_{Q \in [i]_{fit}} (P \mid Q)\, C^Q_{\mu i} \tag{2.39}$$

The fitting coefficients are determined individually for each AO-LMO pair $|\mu i)$, and include only those auxiliary functions centred on atoms in the fitting domain $[i]_{fit}$ for which the Mulliken charges are above a given threshold. Although the fitting coefficients need to be recomputed for each update of the MO coefficients, the number of $|\mu i)$ pairs scales linearly with system size. This type of local density fitting and variations thereof are predominantly used in pair-orbital specific local correlation methods, and will be explained in more detail further below.

2.2.6 LDF (III): Quasi-Robust Density Fitting

Local density fitting inmposes constraints on the fitting procedure, and the integral error consequently scales linearly with the fitting error. Using Dunlap's robust formula is deemed necessary in most cases to achieve acceptable accuracy, but reintroduces the slowly decaying 3c2e integrals. Furthermore, replacing the 4c2e integrals by Eq. ... greatly increases the complexity of expressions in electronic structure theory, which is still manageable for ground state methods, but quickly becomes cumbersome for excited states.

Quasi-robust density fitting (QRDF) aims to combine the exponential decay behaviour of LDF with accuracy comparable to standard density fitting, without the use of Dunlap's formula. Again, consider the fitting procedure

$$\sum_Q (P \mid Q)\, C^Q_{\mu\nu} = (P \mid \mu\nu) \tag{2.40}$$

The sets of auxiliary functions $\{P\}$ and $\{Q\}$ have different roles. The functions Q fit the charge density $|\mu\nu)$, while the P functions act as *test functions* where the electron integrals should be accurate, i.e. where $(X \mid \widetilde{\mu\nu}) \approx (X \mid \mu\nu)$. For two functions μ and ν not located on the same atom, their charge density $|\mu\nu)$ lies in the vacuum between them, and the atom-centred auxiliary functions may be ill-suited to fit $|\mu\nu)$. For this reason, the fitting procedure draws from all fitting functions $\{P\}$ spanning the whole molecule

to cancel out the linear error, which in consequence introduces long-range contributions in $C^P_{\mu\nu}$ in the coulomb metric, even if $|P)$ is not close to $|\mu\nu)$. The basic idea of QRDF is to only chose fitting functions $\{P\}$ close to $|\mu\nu)$ via overlap criteria, but still perform the fitting procedure in the coulomb metric.

The QRDF Fitting Procedure

For a set of given μ, ν, select a set of *fitting function* $\{P_{\mu\nu}\} \in \{P\}$ close to $|\mu\nu)$ according to the criteria

$$\left| \sum_R S^{-1}_{PR}(R\mu\nu) \right| > T \tag{2.41}$$

where S is the auxiliary overlap matrix, and $(R\mu\nu)$ are the 1-centre-3-electron overlap integrals. Next, choose a set of test functions $\{Q_{\mu\nu}\} \in \{P\}$ using

$$f(Q_{\mu\nu}, P_{\mu\nu}) < R \tag{2.42}$$

with

$$f(A, B) = \frac{\alpha\beta}{\alpha + \beta} |\mathbf{A} - \mathbf{B}|^2 \tag{2.43}$$

where for two auxiliary functions A and B, the values α, β are their smallest primitive exponents and $\mathbf{A}$, $\mathbf{B}$ are their respective positions. The fitting coefficients are then determined via

$$\sum_P (Q_{\mu\nu} \mid P_{\mu\nu})\, C^P_{\mu\nu} = (Q_{\mu\nu} \mid \mu\nu) \tag{2.44}$$

where the fitting coefficients are accurate within the set of test functions $\{Q_{\mu\nu}\}$. The linear equations in Eq. ... can be solved via QR decomposition of the rectangular matrix $(Q_{\mu\nu} \mid P_{\mu\nu})$. The QRDF scheme depends on two parameters, T and R. In the limit where $T \to 0$ and $R \to \infty$, the standard fitting procedure in the coulomb metric is recovered.

The fitting functions $\{P_{\mu\nu}\}$ are selected via overlap criteria and therefore scale linearly with the number of pairs $|\mu\nu)$, and consequently the same holds true for the number of test functions $\{Q_{\mu\nu}\}$ close to $\{P_{\mu\nu}\}$ chosen by Eq. ... . In the limit of large molecules, the size of the rectangular matrix in Eq. ... becomes constant and the fitting procedure can be evaluated with $\mathcal{O}(N)$ effort. However, a QR decomposition needs to be computed for each set of $|\mu\nu)$, leading to relatively high prefactor which makes the method unuseable for dense 3D structures like water clusters, as will be discussed in the results section. The QRDF method has been shown to deliver accuracies comparable to standard density fitting, without the use of Dunlap's formula, making it a very attractive alternative to other LDF schemes, especially if one wishes to reduce the complexity of expressions involving LDF.

2.2.7 Auxiliary Basis Sets

The density fitting approximation does not make any assumptions about the size or shape of the auxiliary basis set used. In principle, the fit is exact for the basis set containing all N^2_{AO} gaussian products $\chi_P = \chi_\mu \chi_\nu$. In practice, the product space is over-complete and can be represented by much smaller basis sets. Accurate results can be obtained for

auxiliary basis sets which are about 4 times larger than the principal basis set they are used with.

Auxiliary basis sets generally need more higher angular momentum functions than standard basis sets. Consider an isolated, unperturbed atom, with electrons occupying atomic orbitals with highest angular momentum l_{occ}. A minimal basis set for this atom contains functions of angular momenta 0 to l_{occ}. However, a minimal auxiliary basis set for fitting the product space $\chi_\mu^{0...l_{occ}}\chi_\nu^{0...l_{occ}}$ needs functions with maximum angular momentum $2l_{occ}$. For example, 2nd row elements ($l_{occ} = 1$) need an auxiliary basis set containing d-functions, and first row transition metals ($l_{occ} = 2$) even need g-functions. Similarly to standard basis sets, to describe atoms in molecules where the orbitals are subject to polarization effects, even higher angular momentum functions are needed to fit polarization functions. In practice, a principal basis set with maximum angular momentum l_{bas} is paired with an auxiliary basis set with highest angular momentum $l_{bas} + l_{occ}$.

Auxiliary basis sets have the drawback of being method-specific. There are two categories: auxiliary basis sets for density fitted Hartree Fock (DF-HF) and for density fitted correlated methods (e.g. DF-MP2, DF-CCSD, DF-ADC(2)). Auxiliary basis sets for DF-HF not only need to reproduce Hartree Fock energies, but also need to minimize theor impact on post-Hartree methods. An ill-suited auxiliary basis set leads to a deterioration of the virtual orbital space, and hence an increased error for correlated methods.

Optimization procedures often try to minimize the energy differences between the standard method and its density fitting approximation in a series of atomic calculations. For example, the jkfit family of basis sets (cc-pVXZ-JKFIT [Wei2002], def-XVP-JKFIT [Wei2007]) minimize the error

$$\Delta E_{HF} = E_{HF} - E_{DF-HF} \tag{2.45}$$

The RI basis set family (cc-pVXZ-RIFIT (Wei1998), def2-XVP-RIFIT (Ber1998)) minimize the same energy difference but for MP2 or Coupled Cluster.

Another disadvantage of auxiliary basis sets is that the accuracy of the fitting procedure cannot be easily controlled as a function of its composition (number of functions, angular momenta...), but rather extensive benchmarks are needed for each basis set that is introduced. An alternative approach was proposed by Aquilante et al. (Aqu2007) where the fitting basis sets are generated automatically by cholesky decomposition of the atomic 2-electron integrals

$$(\mu\nu \mid \sigma\lambda) = L_{\mu\nu}^X L_{\sigma\lambda}^X \tag{2.46}$$

The choleksy vectors $\mathbf{L}_{\mu\nu}$ indicate which product densities should be taken to construct the auxiliary basis. This type of atomic Cholesky decompositon (aCD) basis sets has the adavantage that the accuracy can be rigorously controlled by the decompostion threshold θ. To remove linear dependicies in the aCD basis set, another Cholesky decomposition can be performed to yield the atomic compact Cholesky decomposition (acCD) auxiliary basis set (Aqu2009).

2.3 Multipole Expansion of the Electron Integrals

The slow $1/R$ between the product densities $\Omega_{\mu\nu}$ and $\Omega_{\lambda\sigma}$ in the coulomb integrals is a major obstacle for achieving linear scaling in cases where no other sparsity relationship can be stablished between indices beclonging to separate charge densities, e.g. in the evaluation of the coulomb matrix **J** versus the exchange matrix **K**. Luckily, there are approximate methods for integral evaluation that can be computed with $\mathcal{O}(N)$ effort, known as *multipole methods*.

2.3.1 Classical and Non-Classical Electron Integrals

First, one needs to introduce the concept of classical and non-classical interactions. Two electron integrals are said to be *non-classical* if the two charge densities $\Omega_{\mu\nu}$ and $\Omega_{\sigma\lambda}$ overlap, and *classical* if the charge densties are well separated. In the latter case, the electron integrals represent classical interactions between disjoint point charges [ref], and can be approximated using multipole methods, whereas the non-classical contributions must be evaluated using the more expensive standard integral codes such as McMurchie-Davidson or Obara-Saika.

Two gaussian distributions Ω_P and Ω_Q are considered *well-separated* up to a target accuracy 10^{-k}, if their centre-to-centre distance R_{PQ} is larger than the sum of their extents ext_P and ext_Q:

$$R_{PQ} > ext_P + ext_Q \tag{2.47}$$

with the extent of a gaussian product P defined as

$$r_P = \frac{1}{\sqrt{p}} erfc^{-1}(10^{-k}) \tag{2.48}$$

where p is the reduced exponent as given in Equation 0. Another important thing to note is that the number of significant non-classical and classical integrals scale as $\mathcal{O}(N)$ and $\mathcal{O}(N^2)$ respectively (ref), which has important consequences as will be shown further below.

2.3.2 Multipole Expansion

For two well-separated charge distributions P and Q, the inverse interelectronic distance can be exapanded in terms of Legendre polynomials $\mathcal{P}$ as

$$\frac{1}{r_{12}} = \sum_{l=0}^{\infty} \frac{\Delta r_{12}^l}{R_{PQ}^l} \mathcal{P}_l cos\theta \tag{2.49}$$

with

$$cos\theta = \frac{\Delta \mathbf{r}_{12}^l \mathcal{R}_{QP}}{\Delta r_{12} R_{QP}} \tag{2.50}$$

$$\Delta \mathbf{r}_{12} = \mathbf{r}_{1P} \mathbf{r}_{2Q} \tag{2.51}$$

where $\mathbf{r}_{1P}$ and $\mathbf{r}_{2Q}$ are the distance between electron 1,2 and the centres P,Q. Equation ... is also known as the partial-wave expansion of the coulomb operator (Arfk1970).

Plugging Equation ... into Equation gives the bipolar multipole expansion of the two-electron integrals

$$g_{abcd} = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} \sum_{j=0}^{\infty} \sum_{k=-j}^{j} M_{ab}^{lm}(P) T_{lm,jk} M_{cd}^{jk} \tag{2.52}$$

where $\mathbf{M}_{ab}^{lm}(P)$ is the multipole moment of the charge distribution P with total moment $l+k$, and $\mathbf{T}$ is the so-called interaction matrix. As such, the complicated 6-dimensional evaluation of g can be simply substituted by two 3-dimensional integrations of the multipole moments $\mathbf{M}$ at a much lower cost. For the lowest order expansion, where $l = m = 0$, and $j = k = 0$, the multipole moments and the interaction matrix become

$$M_{ab}^{00} = S_{ab} \tag{2.53}$$

$$M_{cd}^{00} = S_{cd} \tag{2.54}$$

$$T_{00,00} = 1/R_{PQ} \tag{2.55}$$

The zero order term of the multipole expansion therefore takes the form

$$g_{abcd} \approx \frac{S_{ab} S_{cd}}{R_{PQ}} \tag{2.56}$$

2.3.3 Fast Multipole Method

While the number of individual non-zero integrals still scales with $\mathcal{O}(N^2)$, the total contribution of all pair-wise interactions to the total energy (Hartree Fock, MP2 ...), can actually be evaluated in $\mathcal{O}(N)$.

For the sake of simplicity, consider system with point-charge particles with charge Z, in a 2-dimensional plane. The total interaction energy is given by

$$U = \sum_{i>j} \frac{Z_i Z_j}{r_{ij}} \tag{2.57}$$

Evaluating Equation ... as is takes a quadratic effort. In a first approximation, one can divide the plane into a grid of blocks of equal size, where each block contains a certain number of particles (Figure ...). Consider the interaction of a single particle i in its source block C with the other particles in the system. The interaction has two contributions: near-field (NF) contributions U_{NF} from the other particles in the source block, and the blocks immediately surrounding it, and far-field (FF) contributions U_{FF} from boxes that are well-separated from C. The NF interactions are evaluated directly by summing over all particles j in the near-field

$$U_i^{NF} = \sum_{j \in NF} \frac{Z_i Z_j}{R_{ij}} \tag{2.58}$$

while FF interactions are computed using multipole expansions $\mathbf{q}_{iC}$ and $\mathbf{q}_A$ of the FF boxes and the particle i

$$U_i^{FF} = \sum_{A \in FF} \mathbf{q}_{iC} \mathbf{T}_{CA} \mathbf{q}_A \tag{2.59}$$

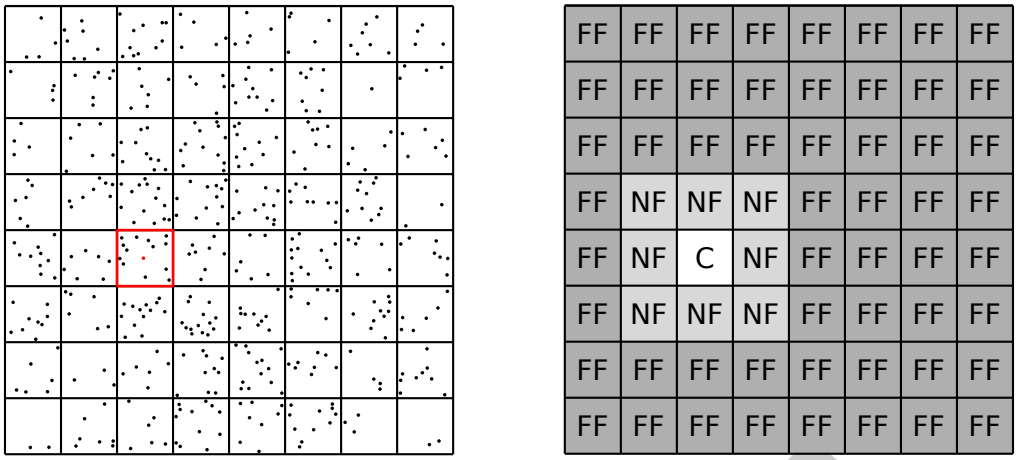


Figure 2.5: A nice caption

While evaluating the interaction energy at block-level rather than particle-level can considerably reduce the prefactor, the cost of this *single-level multipole level* is still quadratic, since for each particle i, there is a system-dependent number of FF boxes. The granularity of the blocks is the same, independent of how far away the blocks are. To achieve linear scaling, the crucial point to realize is that, the further one gets from the source block C, the smaller the single-particle interaction U_i becomes, and the less accurately it actually needs to be evaluated. This means that the farther one moves away from C, the larger the FF boxes can be. For this reason, *multi-level multipole methods* introduce a hierarchy of boxes (Figure ...), where at level 0, the whole system is in a single box, and for each subsequent level, the field is divided into fourths. FF boxes that are closest to C are evaluated at the highest level/granularity S. The region of FF boxes sourrounding the closest FF boxes are then treated at a lower level $S-1$, and so on, until all interactions have been computed. Because the multipole expansion is evaluated for increasing box size, it can be shown that the total number of boxes is constant for a single particle i. This is the basic idea on which the *Fast Multipole Method* (FMM) operates [refs], and it has quickly become one of the most important algorithms in scientific computing, as the problem of particle-particle interaction is not limited to the field of quantum chemistry. FMM can evaluate the total interaction energy U with linear computational complexity.

2.3.4 Continuous Fast Multipole Method

The fast multipole method does not work for continuous charge distrbutions like Gaussian functions, as their extents can be quite different from one another, making the separation into NF and FF contributions more difficult. Nonetheless, FMM has been generalized to the continous case, known as the continuous fast multipole method (CFMM) (ref). The principle is the same as in multi-level multipole methods, only special care needs to be taken to only include classical contributions into the FMM treatment. For further details, the reader is referred to the original publication.

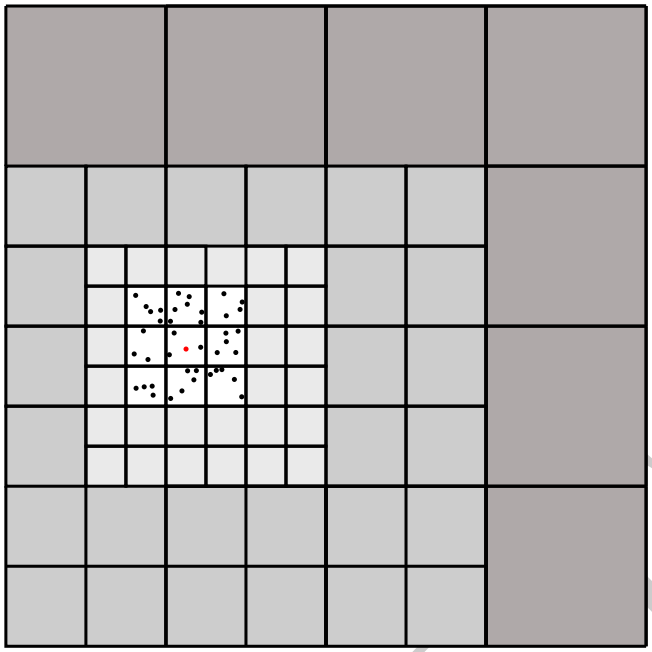


Figure 2.6: Another caption

2.4 The ABCs of LMOs: Orbital Representations

A small intro

Occupied and virtual molcular orbitals can generally be represented in two ways: canonical molecular orbitals (CMOs) and local molecular orbitals (LMOs). CMOs are the eigenvectors of the Fock matrix obtained by solving the eigenvalue problem

$$\mathbf{FC} = \mathbf{SC}\epsilon \tag{2.60}$$

where the eigevalues ϵ are known as the molecular orbital energies of the associated CMOs. However, CMOs are not unique in the sense that there are multiple molecular representations possible which yield the same electron density $\mathbf{P}$. Observables such as the electron density, or the total energy, are said to be invariant under unitary transformations (Fock V (1930) Z Phys 61:26–148). The CMOs $\mathbf{C}$ relate to other representations $\mathbf{L}$ as

$$L_{\mu\underline{i}} = U_{\underline{i}i}C_{\mu i} \tag{2.61}$$

where $\mathbf{U}$ is a unitary transformation matrix with $\mathbf{U}^\dagger\mathbf{U} = \mathbb{1}$. Typically, $\mathbf{U}$ is chosen to generate a set of molecular orbitals which are localized on as few atoms as possible, hence local molecular orbitals. While CMOs and LMOs agree on observables, they show differences for non-obervables, such as molecular orbital energy or orbital shape.

There are several reasons for choosing an LMO representation. First, as mentioned above, LMOs are used in local correlation methods, because CMOs are too delocalized, and electron correlation between LMO centres decay more rapidly. Secondly, they offer a more intuitive picture for chemists and help to interpret chemical phenomena (cite), e.g. invloving lone pairs or π bonds. Different representations can be used to interpret different phenomena, e.g. Boys LMOs vs NTOs.

Over the years, a myriad of different schemes has been proposed on how to find appropriate tranformation matrices **U**. We will now go over some examples.

2.4.1 Local Molecular Orbitals

Some words about it

2.4.2 LMOs by Reducing a Functional

One of the most popular methods for finding LMOs consists in maximizing a localization function $\eta(\phi)$ by successive rotation of the orbital space. The most prominent examples are Foster-Boys (FB)(1), Edmiston-Ruedenberg (ER) (2) and Pipek-Mezey (PM) (3). Their functionals can be written as

$$\zeta_{FB}(\chi) = \sum_i \langle \chi_i | \, \mathbf{r} \, | \chi i \rangle^2 \tag{2.62}$$

$$\zeta_{ER}(\chi) = \sum_i (\chi_i \chi_i \mid \chi_i \chi_i) \tag{2.63}$$

$$\zeta_{FB}(\chi) = \sum_i \sum_A \langle \chi_i | \, \mathbf{P}_A \, | \chi i \rangle^2 \tag{2.64}$$

The problem is generally solved using an iterative procedure consisting in consecutive pair-wise rotations, known as Jacobi sweeps (ALGO). These sweeps are rotated until convergence is reached, which may be slow. The methods differ within the procedure by how the rotational angle is computed, and scale diiferently with system size, with $\mathcal{O}(N^3)$ for FB, $\mathcal{O}(N^5)$ for ER and $\mathcal{O}(N^4)$ for PM. A faster alternative to Jacobi sweeps does also exist (4).

Over the years, PM has been the more popular choice of the three: like ER and unlike FB, it conserves σ-π separation (0), but it scales more favorably than ER.

Functional localization methods are most often used for rotating occupied MOs. Virtual MOs are often plagued by convergence issues and have a steep computational cost simply due to being much more numerous than occupied MOs (5). It is crucial that molecular localization should not take longer than the methods they are used for, and hence VMOs are often localized using separate methods.

EXAMPLES!! Ethylene

2.4.3 Projected Atomic Orbitals

A set of highly localized molecular orbitals can be obtained by projecting the CMOs onto the atomic orbital basis, known as projected atomic orbitals (PAO) (0). For a set of orthonormal occupied and virtual molecular orbitals $\{\Psi_i\}$ and $\{\Psi_a\}$, the projection operators $\hat{P}$ and $\hat{Q}$ are defined as (1)

$$\hat{P} = |\Psi_i\rangle \langle \Psi_i| = |\chi_\mu\rangle \, C_{\mu i} C_{\nu i} \, \langle \chi_\nu| \tag{2.65}$$

$$\hat{Q} = |\Psi_a\rangle \langle \Psi_a| = |\chi_\mu\rangle \, C_{\mu a} C_{\nu a} \, \langle \chi_\nu| \tag{2.66}$$

which are then applied to the atomic orbitals χ

$$\hat{P}\,|\chi_{\mu'}\rangle \;= \textstyle\sum_{\mu} P_{\mu\nu} S_{\nu\mu'}\,|\chi_{\mu'}\rangle \;= \sum_{\mu} \overline{P}_{\mu\mu'}\,|\chi_{\mu'}\rangle = \left|\chi_{\underline{\mu}}\right\rangle \tag{2.67}$$

$$\hat{Q}\,|\chi_{\mu'}\rangle \;= \textstyle\sum_{\mu} Q_{\mu\nu} S_{\nu\mu'}\,|\chi_{\mu'}\rangle \;= \overline{Q}_{\mu\mu'}\,|\chi_{\mu'}\rangle = |\chi_{\overline{\mu}}\rangle \tag{2.68}$$

The projection operators $\hat{P}$, $\hat{Q}$ and the non-symmetric PAO coefficient matrices $\overline{\mathbf{P}}$, $\overline{\mathbf{Q}}$ are *idempotent*

$$\overline{\mathbf{PP}} = \overline{\mathbf{P}} \qquad \overline{\mathbf{QQ}} = \overline{\mathbf{Q}} \tag{2.69}$$

and *mutually orthogonal*

$$\overline{\mathbf{PQ}} = \mathbf{0} \qquad \overline{\mathbf{P}} + \overline{\mathbf{Q}} = \mathbf{1} \tag{2.70}$$

But not orthogonal within themselves

$$\left\langle \chi_{\underline{\mu}} \middle| \chi_{\underline{\nu}} \right\rangle = S^{PAO}_{\underline{\mu\nu}} \qquad \langle \chi_{\overline{\mu}} | \chi_{\overline{\nu}} \rangle = S^{PAO}_{\overline{\mu\nu}} \tag{2.71}$$

The number of PAOs (occupied or virtual) is equal to the number of AOs, and are therefore linearly dependent (redundant). CMOs are transformed to PAOs by using

$$\left|\chi_{\underline{\mu}}\right\rangle \;= (\mathbf{SC})_{\mu i}\,|\Psi_i\rangle \;= \overline{C}_{\mu i}\,|Psi_i\rangle \tag{2.72}$$

$$|\chi_{\overline{\mu}}\rangle \;= (\mathbf{SC})_{\mu a}\,|\Psi_a\rangle \;= \overline{C}_{\mu a}\,|\Psi_a\rangle \tag{2.73}$$

The back-transformation is defined as

$$|\Psi_i\rangle \;= C_{\mu i}\left|\underline{\mu}\right\rangle \tag{2.74}$$

$$|\Psi_a\rangle \;= C_{\mu a}\,|\overline{\nu}\rangle \tag{2.75}$$

PAOs are centred on the atom on which their corresponding AO is localized, but can still be delocalized over multiple atoms, depending on the sparsity of the density matrix. Methods which are entirely formulated in PAOs are rare but possible (1). The projection method is most often used on the virtual orbital space, where standrad localiztaion procedures fail. A set of virtual PAOs is generated from LMOs by using the orthogonality relationship from Eqaution ...

$$\overline{\mathbf{Q}} = \left(\mathbf{1} - \mathbf{L}\mathbf{L}^{\dagger}\mathbf{S}\right)\mathbf{C} \tag{2.76}$$

2.4.4 Subspace Projected Atomic Orbitals

Some applications need localized molecular orbitals that only span a certain region of a molecule, e.g. density matrix embedding theory (DMET) (refs) or local ADC (ref). The molecule is split into two subunits, and atoms are grouped into an active region A and an inactive region B according to specific selection criteria. Region A contains the molecular subunit of interest.

Most implementations use the Mulliken gross charges to find ... ? Not used for virtuals? Put it into AO-ADC Part?

2.4.5 Cholesky Molecular Orbitals

Sparsity of the atomic density matrix is crucial for achieving low-scaling electronic structure methods. Aquilante et al. proposed (0) to define a set of occupied molecular orbitals by Cholesky decomposition of the density matrix. Analysis of the resulting Cholesky molecular orbitals (CholMOs) showed their localized character inherited from the sparsity of the density matrix.

$$\mathbf{P} = \mathbf{L}\mathbf{L}^{\mathbf{T}} \tag{2.77}$$

Figure ... shows the sparsity of the occupied density matrix and the occupied cholesky molecular coefficient matrix of the linear alkane $H_{322}C_{160}$. The number of CholMOs is equal to the rank of the density matrix, which is equal to the number of occupied orbitals. The CholMOs are computed by an incomplete Cholesky decomposition with full row and column pivoting (ALGO). The unitary transformation matrix is given by

$$U_{i\underline{i}} = C_{\mu i} S_{\mu\nu} L_{\nu\underline{i}} \tag{2.78}$$

The decomposition algorithm scales with $\mathcal{O}(N^3)$ but can be made linearly scaling by using sparse matrix algebra. CholMOs have several advantages: the Cholesky decomposition is fast and non-iterative, and an initial guess for molecular orbitals is not needed.

The scheme can be extended to virtual orbitals as well, by CD of the virtual atomic density matrix $\mathbf{Q}$. The rank of $\mathbf{Q}$ is equal to the number of virtual orbitals $n_v ir$, therefore the prefactor of the incomplete CD increases with basis set size. Especially in the presence of diffuse functions, the rank reduction might not offer much of an advantage compared to simpler localization methods such as PAOs.

Moreover, orbitals obtained by CD are less localized than FB or ER LMOs, especially for small molecules. Low scaling is still possible using CholMOs in the context of LMO correlation methods, albeit with a larger prefactor.

CD is also used in the context of AO-MP2 to reduce the prefactor of inetgral transformation by using the rank sparsity of the pseudo-density matrices, as will be shown further below.

CholMOs can also used as an initial guess for iterative localization schemes to achive faster convergence.

2.4.6 Natural Orbitals

While the schemes described above try to generate a set of occupied and/or virtual molecular orbitals localized in space, natural orbital (NOs) methods try to generate a set of "compact" orbitals, i.e. a minimal set of orbitals that can describe the problem at hand. The concept of natural orbitals was first introduced by Löwdin (A). The natural orbitals Θ_i of a wave function Ψ are defined as the eigenfunctions of the one-particle density operator $\hat{n}$

$$\hat{n} \left|\Theta_i\right\rangle = n_i \left|\Theta_i\right\rangle \tag{2.79}$$

where n_i are the occupation numbers of the associated orbital Θ_i. One can then choose a reduced orbital space $\{\tilde{\Psi}_i\}$ by only taking into account those orbitals with an occupation number above a certain threshold τ. The orbitals are "natural" in the sense that they

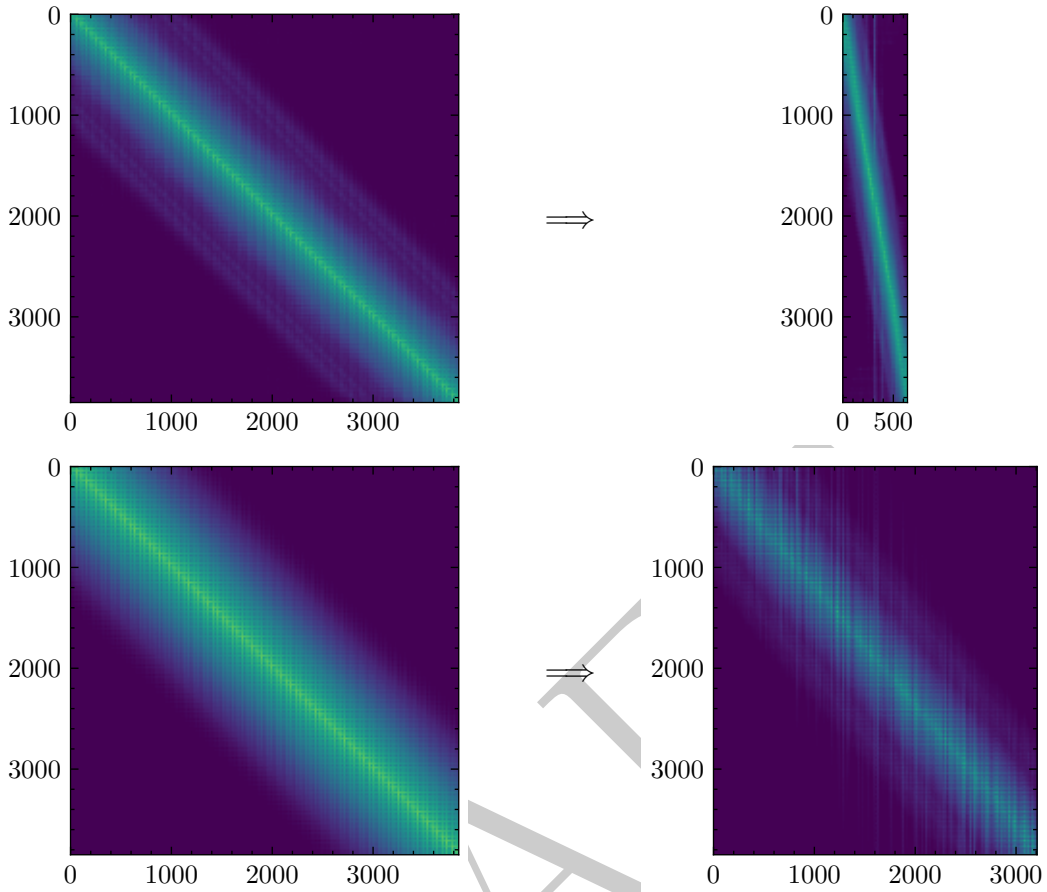


Figure 2.7: Something

are determined purely using Ψ, and are intrinsic to the system. NOs are computed by diagonalizing the one-particle density matrix at the desired level of theory (Hartree-Fock, MP, CIS, CC).

NOs are state-specific (Pok2019), meaning that NOs computed from the ground state densities may not be well suited to describe excited states, and NOs of different excited states might also greatly differ. As such, as will be later shown for local flavours of ADC, NOs need to be recomputed for each state.

Natural Orbitals in Hartree Fock Theory

In Hartree Fock theory, natural orbitals are mostly reserved for qualitative population and bond order analysis.

Natural atomic orbitals (NAOs) are computed by diagonalizing the blocks $P_{\mu_A \nu_A}$ of the atomic density matrix, where μ_A, ν_A are basis functions centred on atom A. NAOs are optimal for describing the electron density around individual atom centres (IUPAC). NAOs are also useful for obtaining a set of guess orbitals from densitiy matrices formed from the superposition of atomic densities (SAD) guess.

NHOs obtained from NAOs + off-diag NAOs

NBOs obtained from NHOs

Frozen Natural Orbitals

For large basis sets, CMOs are much more compact than the virtual orbital span, and the number of occupied NOs is not significantly lower than that of occupied CMOs. It is therefore sufficient to only compute the eigenfunctions of the virtual-virtual block of the one-particle density matrix, which are known as frozen natural orbitals (FNOs) (Bar1970). FNOs need information of the correlated wave function, and are therefore typically computed at a lower level of theory. For example, the easiest way to obtain a set of FNOs for CCSD or CCSD(T) computation is to diagonalize the virtual-virtual block of the MP2 density matrix (Sos1989, Tau2005, Tau2008)

$$D_{ab} = \frac{1}{2} sum_{cij} \frac{K_{ij}^{cb} K_{ij}^{ca}}{\epsilon_{ij}^{ab} \epsilon_{ij}^{ca}} \tag{2.80}$$

with

$$K_{ij}^{ab} = 2\,(ia \mid jb) - (ib \mid ja) \tag{2.81}$$

$$\epsilon_{ij}^{ab} = \epsilon_i + \epsilon_j - \epsilon_a - \epsilon_b \tag{2.82}$$

The FNOs are then canonicalized (see ...). The combined set of occupied CMOs and virtual FNOs forms a very compact representation suitable for CC ground state and excited state calculations.

Natural Transition Orbitals

Consider the CIS eigenvalue problem for finding the excitation energies ω_n and their associated transition density matrices R_n

$$\mathbf{A_{CIS}} R_n = \omega_n R_n \tag{2.83}$$

The matrices $\mathbf{R}_n$ contain $n_{occ} n_{vir}$ expansion coefficients c_{ia} which show how much an orbital-virtual MO pair ia contributes to the excitation n. The numer of non-negligebale coefficients can be far from zero, making interpretations of the computed results difficult for some systems.

Natural transition orbitals (NTOs) were introduced to facilitate the qualititative description of an excited state and finding connections to experimental spectra (Luz1976, Mar2003, Mar2008). NTOs are typically obtained by computing the singular value decomposition (SVD) of the state densities $\mathbf{R}_n$

$$\mathbf{R} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\dagger \tag{2.84}$$

where $\mathbf{U}$ and $\mathbf{V}$ are unitary matrices with dimension $n_{occ} n_{NTO}$ and $n_{vir} n_{NTO}$, and Σ is a n_{NTO} by n_{NTO} matrix containg the singular values s on its diagonal. The CMOs $\{\Psi_i^{occ}, \Psi_a^{vir}\}$ are transformed to the NTO basis $\{\overline{\Psi}_k^{occ}, \overline{\Psi}_k^{vir}\}$ using

$$\left|\overline{\Psi}_k^{occ}\right\rangle = U_{ki} \left|\Psi_i^{occ}\right\rangle \tag{2.85}$$

$$\left|\overline{\Psi}_k^{vir}\right\rangle = V_{ka} \left|\Psi_a^{vir}\right\rangle \tag{2.86}$$

The singular value s_k show the contribution of an NTO pair k to the excited state. In most cases, the number of significant NTO pairs is significantly lower than $n_{occ}n_{vir}$ and at most equal to n_{occ}. NTOs are not limited to CIS, but can also be obtained by SVD decomposition of the singles-singles block of excited state densities from higher order methods such as ADC or CCLR.

Natural transition orbitals have also found use in local excited state correlation methods (Bau2017,Hof2017), where CIS NTOs are combined with MP2 NOs to obtain a compact orbital representation for ground and excited state coupled cluster calculations.

EXAMPLE!!! phenylalanine

2.4.7 Specific Virtual Orbitals

In most cases, using LMOs instead of CMOs does not offer any a priori advantage in terms of the computational complexity asscociated with correlated methods, and additional approximations are necessary. In local correlation methods, this is often done by truncating the VMO space. TRuncation of the VMOs has been an active field of research for "a long time", and several schemes have emerged over the years. A naive approach to truncate the virtual space would be to eliminate VMOs with orbital energies above a certain threshold; however, this proved to be unusable in most contexts (ref). More successful methods for VMO truncation use the concept of what we will refer to as *specific virtual orbitals* (SVOs). SVOs are specific in the sense that each individual occupied LMO i or each pair of LMOs ij has their own set of SVOs a_i (orbital specific virtual orbitals) or a_{ij} (pair specific virtual orbitals) associated to it. The concept of SVOs naturally arises in the context of correlated methods such as the coupled electron pair approximation (CEPA) where the total energy is computed is computed as the sum of electron pair energies e

$$E_{CEPA} = \sum_{ij} e_{ij} \tag{2.87}$$

The electron pair energy decays rapidly as a function of the distance r between MO centres in an LMO basis. Distant virtual orbitals contribute less to the elctron pair energy as virtual orbitals close to ij. It has been shown early on that instead of using the whole virtual orbital span, one can correlate only a subset or reduced set of virtual orbitals with each electron pair (0,1,2,3) and still recover most of the correlation energy. In the limit of large molecules, the number of signiciant virtual orbitals for an electron pair becomes independent of system size (4). There are different ways to choose how to define the VMO subsets.

Domain Specific Virtual Orbitals

We call domain specific virtual orbitals (DSVOs) any type of virtual orbitals where the subsets are formed *a priori* by distance or partial charge criteria. Examples include the local MP2 and local CCSD implementations by Schütz et al.(AA,AB,AC)

First, occupied CMOs are localized by one of the methods described above. The method of choice in Ref [AA-AC] was the Foster-Boys scheme. Virtual CMOs are recast into the PAO basis. Each individual occuied LMO $|\Psi_i\rangle$ is then assigned a subset $[i]$ of PAOs, chosen by a Boughton-Pulay (BP) criterium (AD) or by population analysis (AE).

For a given electron pair ij, the pair domain is then formed by taking the union $[ij] = [i] \cup [j]$. The set of all virtual pair domains $[ij]$ forms the DSVOs.

Alongside AOs, DSVOs were among the first orbital representations in which linear scaling correlated methods were formulated. Their dependency on distance criteria for selecting the pair domains makes them less rigorous than other methods.

Pair Natural Orbitals

First introduced under the guise of "pseudo-natural orbitals" (Edmiston), then rediscovered by Neese (CA,CB,CC), projected natural orbitals (PNOs) have risen in popularity in the recent years (refs). Similarly to DSVMOs, each electron pair has a set of PNOs associated to it. PNOs are formed by diagonalizing the MP2 pair density matrix for each LMO pair ij (hence "pair-natural")

$$\mathbf{D}^{ij} = \frac{1}{1+\delta_{ij}} \left(\tilde{\mathbf{t}}^{ij} \mathbf{t}^{ij} + tilde \mathbf{t}^{ij} \mathbf{t}^{ij\dagger} \right) \tag{2.88}$$

with

$$\tilde{\mathbf{t}}^{ij}_{ab} = 2\mathbf{t}^{ij}_{ab} - \mathbf{t}^{ji}_{ab} \tag{2.89}$$

The eigevalue decomposition of $\mathbf{D}$ then gives

$$\mathbf{D}^{ij} \mathbf{Q}^{ij} = n^{ij} \mathbf{Q}^{ij} \tag{2.90}$$

where $\mathbf{Q^{ij}}$ are the pair specific transformation matrices, and n^{ij} their occupation numbers. The Fock matrix in the LMO representation is not diagonal, and the MP2 amplitudes are approximated by

$$t^{ab}_{ij} = \frac{(ia \mid jb)}{\epsilon_a + \epsilon_b - f_{ii} - f_{jj}} \tag{2.91}$$

where f_{ii} are the diagonal entries of the Fock matrix in the LMO basis. The pair domains $[ij]$ are chosen by keeping the PNOs with an occupation number larger than a threshold τ_{PAO}. Therefore, accuracy is controlled by a single, distance-independent parameter, which is an advantage over other methods like DSVOs.

However, computing the PNOs requires a full MP2 calculation, and with density fitting scales with $\mathcal{O}(N^5)$. Moreover, even if the PNO basis is compact, the fact that each LMO pair has its own virtual orbital basis may lead to a prohibitively large number of PNOs for large molecules

Local PNOs:

Orbital Specific Virtuals

Closely related to PNOs are the orbital specific virtual orbitals (OSVs) (Yan2011). The OSVs for an LMO $|\Psi_i\rangle$ are obatined by taking the diagonal PNOs for the domain $[ii]$. The MP2 density matrix reduces to

$$\mathbf{D}^{ii} = 4\mathbf{t}^{ii}\mathbf{t_{ii}} \tag{2.92}$$

Instead of reducing the density matrix, one can just diagonalize $\mathbf{t}^{ij}$ instead.

$$\mathbf{t}^{ii}\mathbf{Q}^{ii} = t^{ii}\mathbf{Q}^{ii} \tag{2.93}$$

where t^{ii} are the eigenvalues, which are used to compute the compute the occupation numbers $n^{ii} = (t^{ii})^2$. OSVs for which $n^{ii} > \tau_{OSV}$ are included into the orbital specific domain $[i]$. Pair domains $[ij]$ are then formed as the union of $[i]$ and $[j]$ simalar to DSVOs.

OSVs have the advantage that the can be constructed with $\mathcal{O}(N^3)$ scaling provided that density fitting is used. However, OSVs are less compact than PNOs.

OSVs can be used to lower the computational complexity to construct PNOs. Several hybrid OSV-PNO schemes have been proposed whith a computaional complexity of $\mathcal{O}(N^4)$ (Kra2012, Hat2012), $\mathcal{O}(N^3)$ (Sch2013) and finally $\mathcal{O}(N)$ (Rip2013).

Local Pair Natural Orbitals

2.5 Electron Pairs

Nesbet theorem, number pairs etc...

Chapter 3

Local Correlation: Ground State

Some choice words here.....

3.1 Low-Scaling Self-Consistent Field Methods

Hartree Fock and density field theory, also grouped under the umbrella term of self-consistent field (SCF) methods, are the working horses in quantum chemsitry, and the underlying equations, the Roothan and Kohn-Sham equations, are well known and studied. Conventional formulations of HF and DFT, without inclusion of sparsity, scale with $\mathcal{O}(N^3)$ to $\mathcal{O}(N^4)$ which hampers extension to very large systems. There are three major bottle-necks: (1) computation of the coulomb matrix, (2) computation of the exchange matrix and (3) diagonalization of the Fock matrix. Over the last couple of decades, multiple different approaches have been proposed on how to lower the scaling of constructing the Fock matrix and circumvent matrix diagonalization. To this day, the field of low scaling SCF methods remains an active area of research in theoretical chemistry. The next sections will address the time-determining steps in detail.

3.1.1 The Coulomb Matrix

Consider again the expression for the coulomb matrix $\mathbf{J}$

$$J_{\mu\nu} = (\mu\nu \mid \lambda\sigma)\, P_{\sigma\lambda} \tag{3.1}$$

Equation ... gives the following sparsity diagram:

$$\mu \overset{\mathrm{S}}{\longleftrightarrow} \nu \qquad \sigma \overset{\mathrm{P/S}}{\longleftrightarrow} \lambda$$

The counstruction of $\mathbf{J}$ has an inherent compuational complexity of $\mathcal{O}(N^2)$, eventhough the number of non-zero elements scales linearly due to the overlap relationship between μ and ν. Quadratic scaling algorithms are straight-forward to implement. The first method to construct $\mathbf{J}$ with $\mathcal{O}(N)$ effort was the continous fast multipole method (CFMM, cf Section .). For each element $\mu\nu$ the contributions $\sigma\lambda$ are split into near-field and far-field contributions. NF interactions are computed using standard integration techniques, while FF interactions are computed using multi-level multipole expansion. The linear

scaling consists even if the density matrix is not sparse. Other tree-like algorithms have also been proposed (Str1996,Cha1996). In all cases, computing the NF interactions are by far the most time-consuming step.

? J-engine ?

To speed up the evaluation of the non-classical contributions, one may introduce the density fitting approximation (Eq. ...). The coulomb matrix is then evaluated in several steps via an intermediate **d** as

$$J_{\mu\nu} = (\mu\nu \mid X)\, \tilde{d}_X \tag{3.2}$$

$$d_X = (Y \mid \mu\nu)\, P_{\nu\mu}; \quad \tilde{d}_X = (X \mid Y)^{-1}\, d_X \tag{3.3}$$

The computational effort remains unchanged, with $\mathcal{O}(N^2)$, but with a much lower prefactor, especially for larger, more diffuse basis sets (ref Weigend). The inversion of the metric matrix $(X \mid Y)$ scales cubically, and dominates the cost of the DF approximation for large molecules.

For long-range interactions, one could also consider using local density fitting, such as the atomic resolution of the idenstity or pair-atomic reslution of the identity. Unfortunately, LDF also only reduces the prefactor, rather than scaling. Furthermore, it is necessary to use the robust density fitting of the electron integrals (Eqaution ...) to recover quadratic scaling in the fitting error, due to the constraints imposed on the density fitting procedure. The coulomb matrix is then expressed as

$$J_{\mu\nu} = (\mu\nu \mid X)\, b_X + c^Y_{\mu\nu}\, [g_Y - \tilde{g}_Y] \tag{3.4}$$

$$g_X = C^X_{\mu\nu} P_{\mu\nu}; \quad \tilde{g}_X = (X \mid Y)\, b_Y; \quad b_X = C^X_{\mu\nu} P_{\mu\nu} \tag{3.5}$$

The robust LDF-J approximation is evaluated at an effort similar to standard DF-J. The only advantages to LDF in this case pertain to the fitting procedure itself. It is no longer necessary to invert the 2c2e integrals $(X \mid Y)$, and the fitting coefficients $C^X_{\mu\nu}$ can be evaluated in linear scaling fashion. However, it has been demonstrated that using Dunlap's robust formula in combination with local metrics leads to "attractive electron" states where the SCF energy may converge to very high positive values (Mer2013,Hol2014). The reason is that the the two-electron integral tensor in the robust LDF approximation is no longer postive semidefinite, but *indefinite*, which can lead to severe convergence problems. One way to circumvent this problem is to loosen the constraints on the density fitting procedure, or use a larger auxiliary basis set. In both cases, performance is compromised.

As shall be shown in the results section/chapter?, quasi-robust density offers a better alternative to robust LDF-J methods, with high accuracy, and without convergence problems.

MEMORY : ON-THE FLY

POISSON?

3.1.2 The Exchange Matrix

Exact Exchange

The expression for the exchange matrix is given by

$$K_{\mu\nu} = (\mu\sigma \mid \nu\lambda)\, P_{\lambda\sigma} \tag{3.6}$$

In the section where sparsity diagrams were introduced, it was demonstrotated that the indices of the exchange expression can be fully linked:

$$\mu \overset{\mathrm{S}}{\longleftrightarrow} \sigma \overset{\mathrm{P}}{\longleftrightarrow} \lambda \overset{\mathrm{S}}{\longleftrightarrow} \nu$$

The non-zero elements in $\mathbf{K}$ scale linearly and can also be evaluated with $\mathcal{O}(N)$. This property has been realized quite early on (Sch1996). However, a straight-forward implementation where the 4c2e integrals are directly contracted with the density matrix $\mathbf{P}$ using sparse matrix algebra does not give the desired results, when applying the standard $\mathcal{O}(N^2)$ Schwarz-screening to $(\mu\sigma \mid \nu\lambda)$. To lower the scaling of electron integral evaluation, it is important to design a screening algorithm which imposes the P junction between σ and λ which in turn leads to only a linear increase in the number of bra-ket pairs.

The ONX method by Schwegler (Sch1997) was the first $\mathcal{O}(N)$ scheme for constructing the exchange matrix, but did not exploit permutational symmetry, which lead to a four fold increase in the prefactor. More competitive methods were proposed later, such as Linear Exchange (LinK) by Ochsenfled et al (Och1997), or symmetrized ONX (SONX) (Sch2000) that could also be applied to small systems without major overhead.

For all approaches, an important step is the screening of the bra-ket pairs using a density-weighted integral estimate

$$|P_{\lambda\sigma}||\,(\mu\sigma \mid \mu\sigma)\,|^{1/2}|\,(\nu\lambda \mid \nu\lambda)\,|^{1/2} \leq \tau \tag{3.7}$$

which scales linearly for increasing size of systems whith large HOMO-LUMO gaps. Furthermore, shell ordering is very important to avoid the $\mathcal{O}(N^2)$ complexity of thresholding and enable early exit out of the shell loops during contruction of the exchange matrix. Similarly to the J kernel, the electron integrals need not to be held in memory, but can be recomputed on the fly.

Density Fitting

The downside of exact linear exchange algorithms is the steep $\mathcal{O}(N^4)$ scaling with increasing basis set size which delays the onset of the low-scaling regime. For this reason, considerable effort has been invested in recent years to also exploit rank sparsity by density fitting. The DF expression for the exchange matrix in the AO basis reads

$$C^X_{\mu\nu} = (X \mid Y)^{-1}\,(Y \mid \mu\nu) \tag{3.8}$$

$$K_{\mu\nu} = C^X_{\nu\lambda}\,(X \mid \mu\sigma)\,P_{\sigma\lambda} \tag{3.9}$$

In a straight-forward implementation using sparse matrix algebra, Equation ... and Equation .. are evaluated with $\mathcal{O}(N^3)$ and $\mathcal{O}(N^2)$ effort respectively. The non-zero elements of both the fitting coefficients C and the 3c2e integrals increase quadratically. Their storage can quickly become problematic for large basis sets if both are held in-core. In principle, both tensors can be recomputed batchwise on-the-fly to reduce memory-footprint, but in contrast to the DF-J kernel where only the 3c2e integrals need to be generated at each iteration, recomputing the fitting coefficients each time introduces a prefactor that is too large for an out-of-core DF-K kernel to be of any practical use.

For an efficient, direct evaluation of the exchange matrix using density fitting, an MO based approach is much more favourable (Wei2002). The MO-DF-K kernel is evaluated as

$$B^X_{\mu i} = C_{\nu i}\,(X \mid \mu\nu) \tag{3.10}$$

$$D^X_{\mu i} = (X \mid Y)^{-1/2}\, B^X_{\mu i} \tag{3.11}$$

$$K_{\mu\nu} = B^X_{\mu i} B^X_{\nu i} \tag{3.12}$$

with cubic computational complexity. The matrix elements of the exchange matrix are evaluated batch-wise over occupied blocks I. By contracting the 3c2e electron integrals with the coeffcient matrix $C_{\mu i}$ to form the half-transformed integrals $B^X_{\mu i}$, storage can be reduced from $N_{aux}N^2_{AO}$ to $N_{aux}N_{AO}N_{occ}/N_I$. The 3c2e integrals need to be recomputed for each block I, but in practice the number of blocks can be held quite small. The DF-MO-K method is especially well suited for small to medium sized molecules with large diffuse basis sets for post-HF calculations.

Local Density Fitting

For standard density fitting, one looses the linear scaling character of the exchange matrix, which can however be recovered by using LDF. Again, the Dunlap's robust density fitting needs to applied to get accurate results. The robust DF-K kernel can take the form

$$E^X_{\mu\nu} = C^X_{\mu\sigma} P_{\lambda\nu} \tag{3.13}$$

$$L_{\mu\nu} = E^X_{\mu\sigma}\,(X \mid \nu\sigma) - \tfrac{1}{2} E^X_{\mu\sigma}\,(X \mid Y)\, C^Y_{\nu\sigma} \tag{3.14}$$

$$K_{\mu\nu} = L_{\mu\nu} + L_{\nu\mu} \tag{3.15}$$

All steps can be evaluated in $\mathcal{O}(N)$ time, if one presumes linear scaling of the fitting coeffcients:

$$E: \quad X \overset{\text{LDF}}{\longleftrightarrow} \mu \overset{\text{S}}{\longleftrightarrow} \sigma \overset{\text{P}}{\longleftrightarrow} \nu$$

$$L: \quad X \overset{\text{LDF}}{\longleftrightarrow} \mu \overset{\text{P}}{\longleftrightarrow} \sigma \overset{\text{S}}{\longleftrightarrow} \nu \quad + \quad X \overset{\text{LDF}}{\longleftrightarrow} \mu \overset{\text{P}}{\longleftrightarrow} \sigma \overset{\text{S}}{\longleftrightarrow} \nu \overset{\text{LDF}}{\longleftrightarrow} Y$$

$$K: \quad \mu \longleftrightarrow \nu \quad + \quad \mu \longleftrightarrow \nu$$

Alternatively, an LDF-K scheme based on LMOs is also possible (Pol2004,Mej2014). Over the years, many different LDF-K kernels have been proposed that approximate $C^X_{\mu\nu}$ based on LMO domains (Pol2004,Mej2014), the atomic resolution of the identity (ARI) (Sod2007), the pair-atomic resolution of the identity (PARI) (Mer2013) or the concentric atomic density fitting (CADF) (Hol2017). Although the electron integrals are no longer positive semidefinite, LDF-K is not plagued by the same convergence problems as LDF-J, and it has been shown that LDF-K can be combined with standard DF-J to circumvent convergence problems (Man2015).

Other

3.1.3 The SCF Procdure

In the standard SCF procudure, the construction of the Fock matrix is followed by a cubic scaling diagonalization to obtain the MO coefficient matrix and the MO energies. As was discussed in detail in previous sections, the eigenvectors of the Fock matrix, i.e. the canonical MOs, are delocalized, and therefore using a sparse eigenvalue solver is unfortunately not an option. The solution is to entirely avoid any MO quantities and replace the Fock diagonalization step. For a functional, fully AO-based method, the same constraints on the density matrix need to be fulfilled as in the standard SCF procedure:

$$\mathbf{P} = \mathbf{P}^{\dagger} \qquad \text{Hermiticity} \tag{3.16}$$

$$Tr(\mathbf{PS}) = N_{ele} \qquad \text{N-representability} \tag{3.17}$$

$$\mathbf{PSP} = \mathbf{P} \qquad \text{Idempotency} \tag{3.18}$$

$$\mathbf{FPS} - \mathbf{SPF} = \mathbf{0} \qquad \text{Commutator} \tag{3.19}$$

There are two main approaches for replacing Fock matrix diagonalization: Purification (or spectral projection) and density matrix minimzation (Jor2005).

In spectral projection methods, an inital guess for the density matrix $\mathbf{P}$ of its Fock matrix $\mathbf{F}$ is obtained as

$$\mathbf{P} = \Theta\left(\mu\mathbf{I} - \mathbf{F}\right) \tag{3.20}$$

where μ is the chemical potential, and Θ is the Heavyside step-function. The guess density then has orbital occupation numbers spread between 0 and 1, and has the same eigenvectors as the Fock matrix. The idempotent density is then determined by *density purification*. Density purification is a iterative procudre where a purification transformation is repeatedly applied to the density matrix which converges the orbital occupation numbers either towards 0 or 1. One of the earliest purification transformation by McWeeny (Wee1959) takes the form

$$\mathbf{P}_{n+1} = 3\mathbf{P}_n\mathbf{SP}_n - 2\mathbf{P}_n\mathbf{SP}_n\mathbf{SP}_n \tag{3.21}$$

Equation ... is also known as the grand-canonical purification scheme. Other transformations have been proposed over the years, such as canonical purification (Pal1998) or trace-resetting purification (Nik2003), which do not need the chemical potential μ. The only operations in purification schemes are matrix multiplications, which can be made linearly scaling using sparse matrix algebra. Compared to Fock diagonalization, density purification has a larger overhead, but can be easily integrated without needing large modifications of existing SCF code.

The second method, density matrix minimization, starts from an existing idempotent density matrix guess from a previous SCF cycle and minimizes the energy functional (Li1993,Daw1993,Nun1994)

$$E = Tr\left[\left(3\mathbf{PSP} - 2\mathbf{PSPSP}\right)\left(\mathbf{K} - \mu\mathbf{I}\right)\right] \tag{3.22}$$

where $\mathbf{K}$ is the effective one-electron Hamiltonian matrix. The energy minimum is found either by gradient descent or the curvy-step approach (Hel2000,Sha2003).

3.2 Local Ground State Correlation Methods: MP2

Second-Order Møller Plesset is one of the simplest post-Hartree Fock methods available, but still scales as $\mathcal{O}(N^5)$. Since the seminal work of Saebo and Pulay (Pul1983,Sae1985), several different methods have been proposed which drastically reduce the computational complexity. Attempts can generally be grouped into two categories: AO-MP2 and LMO-MP2. While both schemes do have their differences, they share some of the problems assciated with computing the MP2 energy in a local basis.

First, the energy denominator in the MP2-amplitudes t make it difficult to reformulate the MP2 energy expressions in a different basis. AO-MP2 and LMO-MP2 take different approaches: AO-MP2 solves the problem using the Laplace quadrature, while LMO-MP2 methods usually use an orbital-invariant formulation of MP2 using the Hylleraas functional.

Second, steps involving the transformation of the AO 2-electron integrals to the Pseudo-AO or LMO basis still remain a major bottle-neck, even with sparsity involved. Both AO- and LMO-MP2 use screening criteria, additional domain restrictions, density fitting or similar methods to lower the cost of integral transformation. These additional procedures are crucial if one wishes to achieve a truly linear scaling MP2 method with a reduced overhead.

We will now address each point in detail in the next sections.

3.2.1 Atomic Orbital MP2

MP2 was first formulated in the AO basis in 1993 by Häser , and a linear scaling algorithm was presented by Scuseria and Ayala in 1999 (ref). AO-MP2 has since then been extended to DF-MP2 (ref) and SOS-MP2 (ref).

ALSO: CC2 CAN ALSO BE USED SIMILARLY

The Laplace Transform

In 1991, Almlöf showed (Alm1991) that the energy denominator in the MP2 amplitudes can be removed using an integral transform called the *Laplace Transform*

$$\frac{1}{\epsilon_a + \epsilon_b - \epsilon_i - \epsilon_j} = \int_0^\infty e^{-(\epsilon_a+\epsilon_b-\epsilon i-\epsilon_j)t} dt \tag{3.23}$$

The t-integration can be replaced (Has1993) by a finite summation using a functional approximation:

$$\frac{1}{\epsilon_a + \epsilon_b - \epsilon_i - \epsilon_j} \approx \sum_\alpha^n w^{(\alpha)} e^{-(\epsilon_a+\epsilon_b-\epsilon i-\epsilon_j)t^{(\alpha)}} \tag{3.24}$$

where $w^{(\alpha)}$ and $t^{(\alpha)}$ are the Laplace weights and exponents at the Laplace points α. Accuracy can be controlled by the number of Laplace points n. An efficient AO-MP2 implementation heavily relies on an accurate quadrature scheme to achive the desired accuracy using as few Laplace points as possible to reduce overhead caused by the repeated AO transformation at each step. In general, 5-8 Laplace points are needed

to achieve milli-Hartree accuracy, and 10 to 15 points for μHartree accuracy. For more details, the reader is referred to section ... .

AO MP2 Equations

Using the Laplace transform, the energy expression for restricted canonical MP2 can be expressed as

$$\begin{aligned} E_{MP2} &= -\sum_{iajb} \frac{(ia \mid jb)\left[2\,(ia \mid ib) - (ib \mid ja)\right]}{\epsilon_a + \epsilon_b - \epsilon_i - \epsilon_j} \\ &\approx -\sum_{\alpha}^{n} \sum_{iajb} (ia \mid jb)\left[2\,(ia \mid ib) - (ib \mid ja)\right] w^{(\alpha)} e^{-(\epsilon_a+\epsilon_b-\epsilon i-\epsilon_j)t^{(\alpha)}} \end{aligned} \tag{3.25}$$

We can then proceed to factor out the coefficient matrices

$$\begin{aligned} &-\sum_{\alpha}^{n} \sum_{iajb} (ia \mid jb)\left[2\,(ia \mid ib) - (ib \mid ja)\right] w^{(\alpha)} e^{-(\epsilon_a+\epsilon_b-\epsilon i-\epsilon_j)t^{(\alpha)}} \\ &= -\sum_{\alpha}^{n} \sum_{iajb} \sum_{\substack{\mu\nu\lambda\sigma \\ \mu'\nu'\lambda'\sigma'}} w^{(\alpha)} e^{-(\epsilon_a+\epsilon_b-\epsilon i-\epsilon_j)t^{(\alpha)}} C_{\mu' i} C_{\sigma' a}\,(\mu'\sigma' \mid \nu'\lambda')\, C_{\nu' j} C_{\lambda' b} \\ &\qquad \times \left\{C_{\mu i} C_{\sigma a}\left[2\,(\mu\sigma \mid \nu\lambda) - (\mu\lambda \mid \nu\sigma)\right] C_{\nu j} C_{\lambda b}\right\} \\ &= -\sum_{\alpha}^{n} \sum_{\substack{\mu\nu\lambda\sigma \\ \mu'\nu'\lambda'\sigma'}} \underline{P}^{(\alpha)}_{\mu\mu'} \overline{P}^{(\alpha)}_{\sigma\sigma'}\,(\mu'\sigma' \mid \nu'\lambda')\, \underline{P}^{(\alpha)}_{\nu\nu'} \overline{P}^{(\alpha)}_{\lambda\lambda'} \left[2\,(\mu\sigma \mid \nu\lambda) - (\mu\lambda \mid \nu\sigma)\right] \end{aligned} \tag{3.26}$$

with the occupied and virtual *pseudo* or *Laplace* density matrices

$$\begin{aligned} \underline{P}^{(\alpha)}_{\mu\mu'} &= \sum_{i} C_{\mu i} e^{0.25 ln(w^{(\alpha)}) + \epsilon_i t^{(\alpha)}} C_{\mu' i} \\ \overline{P}^{(\alpha)}_{\mu\mu'} &= \sum_{i} C_{\sigma a} e^{0.25 ln(w^{(\alpha)}) - \epsilon_a t^{(\alpha)}} C_{\sigma' i} \end{aligned} \tag{3.27}$$

Introducing the *pseudo-AO* transformed electron integrals

$$\left(\underline{\mu}\overline{\sigma} \mid \underline{\nu}\overline{\lambda}\right)^{(\alpha)} = \underline{P}^{(\alpha)}_{\mu\mu'} \overline{P}^{(\alpha)}_{\sigma\sigma'}\,(\mu'\sigma' \mid \nu'\lambda')\, \underline{P}^{(\alpha)}_{\nu\nu'} \overline{P}^{(\alpha)}_{\lambda\lambda'} \tag{3.28}$$

the energy expression for AO-MP2 reads

$$E_{AO-MP2} = -\sum_{\alpha}^{n} \sum_{\mu\nu\lambda\sigma} \left(\underline{\mu}\overline{\sigma} \mid \underline{\nu}\overline{\lambda}\right)^{(\alpha)} \left[2\,(\mu\sigma \mid \nu\lambda) - (\mu\lambda \mid \nu\sigma)\right] \tag{3.29}$$

For t = 0, $\underline{\mathbf{P}}^{(\alpha)}$ and $\overline{\mathbf{P}}^{(\alpha)}$ are equal to the Hartree Fock density matrices. The Laplace matrices also fulfill similar relationships

$$\underline{\mathbf{P}}^{(\alpha)} \mathbf{S} \overline{\mathbf{P}}^{(\alpha)} = \mathbf{0} \tag{3.30}$$

$$\underline{\mathbf{P}}^{(\alpha)}\mathbf{S} + \overline{\mathbf{P}}^{(\alpha)}\mathbf{S} = \mathbf{I}_{exp} \tag{3.31}$$

where $\mathbf{I}_{exp}$ is a diagonal matrix with trace

$$Tr[\mathbf{I}_{exp}] = \sum_i e^{0.25ln(w^{(\alpha)})+\epsilon_i t^{(\alpha)}} + \sum_a e^{0.25ln(w^{(\alpha)})-\epsilon_a t^{(\alpha)}} \tag{3.32}$$

The entries of the pseudo-density also decay exponentially as function of the distance between pseudo-AO centres.

STRICTLY SPEAKING, NOT ENTIRELY IN AN AO BASIS, BUT MIXED AO-pseudo-PAO-basis.

Quadratic Scaling AO-MP2

Using the linked index rule, we can easily find the computational complexity of the AO-MP2 method. From the energy expression in Eq. we find that we compute the dot product between two different tensors, the AO-ERIs $(\mu\sigma \mid \nu\lambda)$ and the pseudo-AO-ERIs $\left(\underline{\mu\overline{\sigma}} \mid \underline{\nu\overline{\lambda}}\right)^{(\alpha)}$. The scaling is thus determined by the sparsity of those two tensors. We know from a previous discussion that the ERIs can be computed with $\mathcal{O}(N^2)$ effort. The pseudo-AO ERIs are computed by transforming the ERIs with the pseudo-density matrices, whose indices μ, ν are connected by a P junction. The diagrammatic expression for the pseudo-AO ERIs in Eq. ... is given by

$$\begin{aligned} \mu &\overset{P}{\longleftrightarrow} \mu' \overset{S}{\longleftrightarrow} \sigma' \overset{P}{\longleftrightarrow} \sigma \\ \nu &\overset{P}{\longleftrightarrow} \nu' \overset{S}{\longleftrightarrow} \lambda' \overset{P}{\longleftrightarrow} \lambda \end{aligned} \tag{3.33}$$

Two vertices indicate an $\mathcal{O}(N^2)$ effort for evaluating Eq. ... . Therefore, the inherent asymptotic scaling of AO-MP2, without any other further approximations, is $\mathcal{O}(N^2)$ as well. Similarly to the AO ERIs, a quadratic scaling evaluation of the pseudo-AO ERIs can be achived using a Schwarz-like screening, as first advocated by Almlöf. Defining the screening matrices

$$\begin{aligned} Q_{\mu\nu} &= \mid (\mu\nu \mid \mu\nu) \mid^{1/2} \\ X_{\mu\nu} &= \mid \left(\underline{\mu\nu} \mid \underline{\mu\nu}\right) \mid^{1/2} \\ Y_{\mu\nu} &= \mid (\mu\overline{\nu} \mid \mu\overline{\nu}) \mid^{1/2} \\ Z_{\mu\nu} &= min\left(\sum_\sigma A_{\mu\sigma} \mid \overline{P}_{\sigma\nu} \mid; \sum_\sigma B_{\mu\sigma} \mid \underline{P}_{\sigma\nu} \mid\right) \end{aligned} \tag{3.34}$$

gives an upper-bound for each transformation step in Eq. ..., for example

$$(\mu'\sigma' \mid \nu'\lambda') \leq Q_{\mu'\sigma'}Q_{\nu'\lambda'} \tag{3.35}$$

$$\left(\underline{\mu\sigma'} \mid \nu'\lambda'\right) \leq X_{\mu'\sigma'}Q_{\nu'\lambda'} \tag{3.36}$$

$$\left(\underline{\mu\overline{\sigma}} \mid \underline{\nu\overline{\lambda}}\right) \leq Z_{\mu\sigma}Z_{\nu\lambda} \tag{3.37}$$

and an eficient screening protocol can be devised (Has1993) to get quadratic scaling AO-MP2.

Linear Scaling AO-MP2

For the two-electron repulsion integrals, the $1/R$ decay between the charge densities $(\mu\sigma|$ and $|\nu\lambda)$ is too slow to be of any use even for large systems. However, it has been shown (Aya1999) that *bra* and *ket* in the Laplace integral tensor $e^{(\alpha)}$ decays much faster with $1/R^3$. Here, we follow the discussion in Ref Lam2005a.

For two non-overlapping charge densities $(\mu\sigma|$ and $|nu\lambda)$ the following inequality holds

$$(\mu\sigma \mid \nu\lambda) = (\mu\sigma| \frac{1}{\mathbf{r}_{12}} |\nu\lambda) \leq \frac{1}{R} \left| \sum_{n=0}^{\infty} \frac{(\mu\sigma| (\mathbf{r}_1 - \mathbf{r}_2)^n |\nu\lambda)}{R^n} \right| \tag{3.38}$$

We then introduce the following abbreviation for the nth order 1-centre multipole integrals

$$M_{\mu\sigma}^{(n)} = \int \chi_\mu(r_1) \mathbf{r}_1^n \chi_\sigma(r_1) dr \tag{3.39}$$

where M^0 are the overlap integrals, M^1 are the dipole integrals etc. We can then rewrite equation (...) as a multipole expansion

$$\begin{aligned}
(\mu\sigma \mid \nu\lambda) \leq\ & R^{-1} \left| M_{\mu\sigma}^{(0)} M_{\nu\lambda}^{(0)} \right| + R^{-2} \left| M_{\mu\sigma}^{(1)} M_{\nu\lambda}^{(0)} - M_{\mu\sigma}^{(0)} M_{\nu\lambda}^{(1)} \right| \\
& + R^{-3} \left| M_{\mu\sigma}^{(2)} M_{\nu\lambda}^{(0)} - 2 M_{\mu\sigma}^{(1)} M_{\nu\lambda}^{(1)} + M_{\mu\sigma}^{(0)} M_{\nu\lambda}^{(2)} \right| \\
& + R^{-4} \left| M_{\mu\sigma}^{(3)} M_{\nu\lambda}^{(0)} - 3 M_{\mu\sigma}^{(2)} M_{\nu\lambda}^{(1)} + 3 M_{\mu\sigma}^{(1)} M_{\nu\lambda}^{(2)} - M_{\mu\sigma}^{(0)} M_{\nu\lambda}^{(3)} \right| \\
& + \mathcal{O}(R^{-5})
\end{aligned} \tag{3.40}$$

From equation ..., we know that $M_{\underline{\mu}\overline{\sigma}}^{(0)} = S_{\underline{\mu}\overline{\sigma}} = 0$. The multipole expansion for the pseudo-AO ERIs $e^{(\alpha)}$ is therefore reduced to

$$\begin{aligned}
\left(\underline{\mu}\overline{\sigma} \mid \underline{\nu}\overline{\lambda}\right) \leq\ & R^{-3} \left| -2 M_{\underline{\mu}\overline{\sigma}}^{(1)} M_{\underline{\nu}\overline{\lambda}}^{(1)} \right| \\
& + R^{-4} \left| -3 M_{\underline{\mu}\overline{\sigma}}^{(2)} M_{\underline{\nu}\overline{\lambda}}^{(1)} + 3 M_{\underline{\mu}\overline{\sigma}}^{(1)} M_{\underline{\nu}\overline{\lambda}}^{(2)} \right| + \mathcal{O} \\
& + \mathcal{O}(R^{-5})
\end{aligned} \tag{3.41}$$

which shows the $1/R^3$ dependence of the tensor $\left(\underline{\mu}\overline{\sigma} \mid \underline{\nu}\overline{\lambda}\right)$. Combined with the $1/R$ decay of the AO ERIs, this leads to an overall $1/R^4$ behaviour for the AO-MP2 energy. This long-range decay can be exploited to introduce a sparsity relationship between the bra and ket quantities, and reduce the scaling of AO-MP2 from $\mathcal{O}(N^2)$ to $\mathcal{O}(N^1)$. In the original paper by Ayala and Scuseria, this decay was accounted for by introducing an interaction domain centred on each atomic orbital μ in the form of a sphere. For the integrals $\left(\underline{\mu}\overline{\mu} \mid \underline{\nu}\overline{\nu}\right)$, the domain $\mathcal{D}(\mu)$, comprises all charge distributions $\sigma\lambda$ for which

$$\left(\underline{P}_{\mu\sigma} S_{\sigma\lambda} \overline{P}_{\lambda\mu}\right) \geq \epsilon \tag{3.42}$$

The radius R_μ of the interaction sphere is defined by the maximum distance between μ and the charge density $\sigma\lambda$ in its domain. One can then screen long-range behaviour for the interaction sphere μ and ν by the distance criterium

$$r_{\mu\nu} - R_\mu - R_\nu \geq r_0 \tag{3.43}$$

The biggest drawback of the scheme above is that the thresholding parameters r_0 and ϵ are system-dependent. A more rigorous screening method has been proposed by Lambrecht et al. known as multipole based integral estimates (MBIE) (Lam2005,Lam2005a). MBIEs offer a tight upper bound for the AO and pseudo-AO electron integrals by using the multipole expansion in Eq. ... and replacing the higher order terms $\mathcal{O}(R^{-5})$ by lower-order ones. ALternative: QQR screening

Cholesky Decomposition of Pseudo-Densities

As with any method formulated entirely in an AO basis, AO-MP2 suffers from $\mathcal{O}(N^4)$ scaling with increasing basis set N. The cost associated with larger basis sets can be mitigated by Cholesky decomposition of the pseudo-density matrices (CDD) (Zie2009). Similar to the orbital localization technique described in Section ..., where the (incomplete) CD of the occupied and virtual Hartree Fock density matrices yields a set of occupied and virtual Cholesky molecular orbitals, the CD of the pseudo-density matrices $\underline{P}^{(\alpha)}$ and $\overline{P}^{(\alpha)}$ yields a set of Cholesky pseudo-molecular orbitals:

$$\underline{\mathbf{P}}^{(\alpha)} = \underline{\mathbf{L}}^{(\alpha)}\underline{\mathbf{L}}^{(\alpha)T} \tag{3.44}$$

$$\overline{\mathbf{P}}^{(\alpha)} = \overline{\mathbf{L}}^{(\alpha)}\overline{\mathbf{L}}^{(\alpha)T} \tag{3.45}$$

The pseudo-molecular orbitals show a local behaviour inherited from the sparsity of the pseudo-density matrices. It has been observed however (Lue2017), that the pseudo-MOs L are not always very well localized. A more localized set of MOs can be obtained by using the orthogonalized pseudo-density matrices, for example in the case of $\underline{\mathbf{P}}^{(\alpha)}$:

$$\underline{\mathbf{P}}^{(\alpha)}_{orth} = \mathbf{S}^{1/2}\underline{\mathbf{P}}^{(\alpha)}\mathbf{S}^{1/2} \tag{3.46}$$

The pseudo-MO coefficients are then obtained as

$$\underline{L}^{(\alpha)} = \mathbf{S}^{-1/2}\underline{\mathbf{L}}^{(\alpha)}_{orth} \tag{3.47}$$

The square root and inverse square root of the overlap matrix $\mathbf{S}$ are most reliably found by cholesky decomposition. The number of occupied and virtual pseudo-MOs is given by the rank of the occupied/virtual pseudo-density matrices, which in turn is equal or less than the number of occupied/virtual CMOs.

One can then formulate the CDD-AO-MP2 energy expression

$$E_{CDD-AO-MP2} = -\sum_{\alpha}^{n}\sum_{\underline{i}\overline{a}\underline{j}\overline{b}} \left(\underline{i}\overline{a} \mid \underline{j}\overline{b}\right)^{(\alpha)} \left[2\left(\underline{i}\overline{a} \mid \underline{j}\overline{b}\right)^{(\alpha)} - \left(\underline{i}\overline{b} \mid \underline{j}\overline{a}\right)^{(\alpha)}\right] \tag{3.48}$$

whith the pseudo-MO integrals

$$\left(\underline{i}\overline{a} \mid \underline{j}\overline{b}\right)^{(\alpha)} = \underline{L}^{(\alpha)}_{\mu\underline{i}}\overline{L}^{(\alpha)}_{\sigma\overline{a}}\left(\mu\sigma \mid \nu\lambda\right)\underline{L}^{(\alpha)}_{\nu\underline{j}}\overline{L}^{(\alpha)}_{\lambda\overline{b}} \tag{3.49}$$

CDD-AO-MP2 therefore reduces the sizes of the tensors from N^4_{AO} to $N^2_{occ}N^2_{vir}$, while still being sparse. Similar to AO-MP2, Schwarz screening and interaction domain can be introduced to obtain quadratic and linear scaling CDD-AO-MP2.

Density Fitting in AO-MP2

To reduce the prefactor associated with integral transformation, either from AOs to pseudo-AOs, or from AOs to pseudo-MOs, on can furthermore introduce density fitting (Zie2009,Mau2014).The transformed 3c2e integrals are given at each Laplace point α by

$$\left(X \mid \underline{\mu\overline{\nu}}\right)^{(\alpha)} = \left(X \mid \mu'\nu'\right) \underline{P}^{(\alpha)}_{\mu\mu'} \overline{P}^{(\alpha)}_{\nu\nu'} \tag{3.50}$$

which are evaluated with $\mathcal{O}(N^2)$ cost. Using local density fitting approxaimations, this step can be reduced to linear.

SOS-AO-DF-MP2

From section ..., we know that SOS-MP2 is a cost-efficient variant of MP2 with excellent accuracy. Starting from equation ..., we omit the same-spin contributions and also apply the density fitting approximation to arrive at the energy expression for the AO-DF-SOS-MP2 (Mau2014,Gla2020)

$$\begin{aligned} E_{AO-DF-SOS-MP2} = -c_{os} \sum_{\alpha=1}^{nlap} \sum_{\mu\nu\sigma\lambda} & \left(\underline{\nu\overline{\sigma}} \mid X\right)^{(\alpha)} \left(X \mid Y\right)^{-1} \left(Y \mid \underline{\nu\overline{\lambda}}\right)^{(\alpha)} \\ & \left(\mu\sigma \mid X'\right) \left(X' \mid Y'\right) \left(Y' \mid \nu\lambda\right) \end{aligned} \tag{3.51}$$

Introducing the intermediates

$$Z^{(\alpha)}_{XY} = \left(X \mid \underline{\mu\overline{\sigma}}\right)^{(\alpha)} \left(\mu\sigma \mid Y\right) \tag{3.52}$$

$$\tilde{Z}^{(\alpha)}_{XY} = \left(X \mid R\right)^{-1} Z^{(\alpha)}_{RX} \tag{3.53}$$

we arrive at a very compact expression

$$E_{AO-DF-SOS-MP2} = -c_{os} \sum_{\alpha=1}^{nlap} \sum_{XY} \tilde{Z}^{(\alpha)}_{XY} \tilde{Z}^{(\alpha)}_{YX} \tag{3.54}$$

Without local density fitting, the time determining step is the computation of $\mathbf{Z}^{(\alpha)}$ The sparse map of the intermediate is given by

$$\begin{array}{ccc} X & \mu' \overset{\mathrm{S}}{\leftrightarrow} \nu' & \\ & \mathrm{P}\updownarrow \qquad \updownarrow\mathrm{P} & \\ & \mu \underset{\mathrm{S}}{\longleftrightarrow} \nu & Y \end{array}$$

which suggests that the AO-DF-SOS-MP2 has an overall asymptotic scaling of $\mathcal{O}(N^3)$. With local density fitting, the graph can however become fully connected

$$\begin{array}{ccc} & \mathrm{LDF} & \\ X & \mu' \overset{\mathrm{S}}{\leftrightarrow} \nu' & \\ & \mathrm{P}\updownarrow \qquad \updownarrow\mathrm{P} & \\ & \mu \underset{\mathrm{S}}{\longleftrightarrow} \nu & Y \\ & \mathrm{LDF} & \end{array}$$

where "LDF" is the sparsity relationship introduced between the auxiliary density X and the product density $(\mu\nu|$, which is metric-specific. In the case of quasi-robust density fitting, LDF = S, and the intermediates $\mathbf{Z}^{(\alpha)}$ can be constructed with linear effort. For weaker decay behaviour, such as the error function coulomb-attenuated metric, the scaling is intermediate between linear and quadratic (Gla2020).

CAN BE APPLIED TO CC2

3.2.2 Local Molecular Orbital MP2

Problems with PAO based methods:

While linear scaling MP2 was first achived using an atomic orbital formulation, the first low-scaling MP2 implementations were actually formulated in a local molecular orbital basis with domain-specific virtual orbitals (Pul1983-Sae1987). SEPA, electron pairs etc, NESBETS theorem ....

Laplace LMP2

In the local molecular orbital basis, the Fock matrix is no longer diagonal, and the amplitudes t_{ia}^{jb} can no longer be easily expressed in a local basis, due to the energy denominator. AO-MP2 tackle this problem by virtue of the Laplace transform. Similarly, one can obtain an energy expression in the LMO basis. The Laplace decomposed MP2 energy is given by

$$E_{MP2} = \sum_{\alpha}^{n} \sum_{iajb} |w^{(\alpha)}| e^{(\epsilon_i+\epsilon_j-\epsilon_a-\epsilon_j)t^{(\alpha)}} \left[2\,(ia \mid jb) - (ib \mid ja)\right](ia \mid jb) \tag{3.55}$$

Introducing the unitary occupied and virtual LMO-MO transformation matrix $\mathbf{U}$

$$|i\rangle = U_{i\underline{i}}\,|\underline{i}\rangle \tag{3.56}$$

$$|a\rangle = U_{i\overline{a}}\,|\overline{a}\rangle \tag{3.57}$$

which is factorized out, Equation ... becomes

$$\begin{aligned} E_{MP2} &= \sum_{\alpha}^{n} \sum_{iajb} \sum_{\underline{i}\overline{a}\overline{b}} \sum_{\underline{k}\overline{c}\underline{l}\overline{d}} |w^{(\alpha)}| e^{(\epsilon_i+\epsilon_j-\epsilon_a-\epsilon_j)t^{(\alpha)}} U_{i\underline{i}} U_{a\overline{a}} \left[2\left(\underline{i}\overline{a} \mid \underline{j}\overline{b}\right) - \left(\underline{j}\overline{b} \mid \underline{j}\overline{b}\right)\right] U_{j\underline{j}} U_{b\overline{b}} \\ &\qquad U_{i\underline{k}} U_{a\overline{c}} \left(\underline{k}\overline{c} \mid \underline{l}\overline{d}\right) U_{j\underline{l}} U_{b\overline{d}} \\ &= \sum_{\alpha}^{n} \sum_{\underline{i}\overline{a}\overline{b}} \left[2\left(\underline{i}\overline{a} \mid \underline{j}\overline{b}\right) - \left(\underline{j}\overline{b} \mid \underline{j}\overline{b}\right)\right] \sum_{\underline{k}\overline{c}\underline{l}\overline{d}} X_{\underline{i}\underline{k}}^{(\alpha)} Y_{\overline{a}\overline{c}}^{(\alpha)} \left(\underline{k}\overline{c} \mid \underline{l}\overline{d}\right) X_{\underline{j}\underline{l}}^{(\alpha)} Y_{\overline{b}\overline{d}}^{(\alpha)} \\ &= \sum_{\underline{i}\overline{a}\overline{b}} \left[2\left(\underline{i}\overline{a} \mid \underline{j}\overline{b}\right) - \left(\underline{j}\overline{b} \mid \underline{j}\overline{b}\right)\right] \mathcal{T}_{\underline{i}\overline{a}\underline{j}\overline{b}} \end{aligned} \tag{3.58}$$

with the Laplace amplitudes $\mathcal{T}$ and the Laplace matrices

$$X_{\underline{i}\underline{k}}^{(\alpha)} = \sum_{i} U_{i\underline{i}} |w^{(\alpha)}|^{1/4} e^{\epsilon_i t^{(\alpha)}} U_{k\underline{k}} \tag{3.59}$$

$$Y_{\overline{ac}}^{(\alpha)} = \sum_{a} U_{a\overline{a}} |w^{(\alpha)}|^{1/4} e^{-\epsilon_a t^{(\alpha)}} U_{c\overline{c}} \tag{3.60}$$

Equation ... is the general expression for the MP2 energy in a local molecular orbital basis, where both the occupied and virtual orbitals are *orthogonal*. The situation changes slightly when using non-orthogonal PAOs, as PAOs and CMOs are no longer related by a unitary transformation:

$$|I\rangle = P_{Ii} |i\rangle \tag{3.61}$$

$$|i\rangle = P_{Ii} S_{IJ}^{-1} |J\rangle \tag{3.62}$$

with **S** being the overlap matrix in the PAO basis. Due to the non-orthogonality of the PAOs, entries in the overlap matrix can become very small which might lead to numerical instability when computing its inverse. For this reason, the inverse $\mathbf{S}^{-1}$ is substituted by a more general *pseudo-inverse* **V** with the property (see ...)

$$\mathbf{SVS} = V \tag{3.63}$$

The Laplace matrices will then take the following form instead (Kat2008)

$$\mathbf{X}^{(\alpha)} = \mathbf{V}\mathbf{A}^{(\alpha)}\mathbf{V}^{\dagger} \tag{3.64}$$

$$\mathbf{Y}^{(\alpha)} = \mathbf{V}\mathbf{B}^{(\alpha)}\mathbf{V}^{\dagger} \tag{3.65}$$

$$A_{IK}^{(\alpha)} = \sum_{i} P_{Ii} |w^{(\alpha)}|^{1/4} e^{\epsilon_i t^{(\alpha)}} P_{kK} \tag{3.66}$$

$$B_{AB}^{(\alpha)} = \sum_{a} Q_{Aa} |w^{(\alpha)}|^{1/4} e^{-\epsilon_a t^{(\alpha)}} Q_{Aa} \tag{3.67}$$

In practice, only the virtual orbital space is transformed to PAOs, while the occupied space is kept in an orthogonal local orbital representation.

NESBETS THEOREM

Orbital Invariant MP2

Alternatively, the local MP2 amplitudes can be determined iteratively via an *orbital-invariant* formulation of the MP2 energy expression. Orbital-invariant MP2 predates LT-MP2 by a decade and is based on the *Hylleraas functional* (Hyl,Pul1986). The Hylleraas functional form of the energy is given by minimizing

$$E^{(2)} = min \left[2 \left\langle \Psi^{(1)} \right| \mathbf{H} - E_0 \left| Psi^{(0)} \right\rangle - \left\langle \Psi^{(1)} \right| \mathbf{H}_0 - E_0 \left| \Psi^{(1)} \right\rangle \right] \tag{3.68}$$

In the case of MP2, the qunatities in Equation ... take the form

$$\left\langle \Psi^{(1)} \right| \mathbf{H} - E_0 \left| \Psi^{(0)} \right\rangle = \frac{1}{4} \sum_{ijab} t_{ijab} \langle ij|ab\rangle \tag{3.69}$$

$$\left\langle \Psi^{(1)} \right| \mathbf{H}_0 - E_0 \left| \Psi^{(1)} \right\rangle = \frac{1}{8} \sum_{ijabc} t_{iajb} f_{cb} t_{iajc} - \frac{1}{8} \sum_{ijkab} t_{iajb} f_{jk} t_{iakb} \tag{3.70}$$

Minimization of the MP2 Hylleraas functional, with respect to the amplitudes $\mathbf{t}$ yields a set of linear equations given by

$$R_{iajb} = \langle ij| \, |ab\rangle + \sum_c \left(t_{ijab} f_{cb} + f_{ac} t_{iacb}\right) - \sum_k \left(t_{iakb} f_{kj} + f_{ik} t_{kajb}\right) = 0 \tag{3.71}$$

where $\mathbf{R}$ is the residual. The amplitudes $\mathbf{t}$ are then no longer computed directly by a closed expresion, but iteratively by solving the system of equations, in a similar vein to coupled cluster. This method is said to be orbital invariant, because any molecular orbital representation can be used. For a set of orthogonal MOs $\underline{i}$ and $\overline{a}$, the quantities in Equation ... are simply replaced by their local equivalent. As was the case in LT-LMP2, if PAOs are to be used for the virtual orbital space, the non-orthogonality needs to be taken into consideration. For a mixed LMO-PAO basis, Equation ... reads

$$\begin{aligned} R_{\underline{i}A\underline{j}B} = \langle \underline{ij}| \, |AB\rangle + \sum_C \Big(f_{AC} t_{\underline{i}C\underline{j}L} S_{LB} + S_{AC} t_{\underline{i}C\underline{j}D} f_{DB}\Big) \\ - \sum_{\underline{k}} \Big(f_{\underline{ik}} S_{AC} t_{\underline{k}C\underline{j}D} S_{DB} + f_{\underline{kj}} S_{AC} t_{\underline{i}C\underline{j}D} S_{DB}\Big) = 0 \end{aligned} \tag{3.72}$$

For specific virtual orbitals, the equations are solved individually for each electron pair ij to obtain their amplitude $\mathbf{t}_{ij}$ and then compute the pair correlation energy.

why not AO?

Quadratic Scaling LMP2

Similar to CEPA, the MP2 energy can be computed as a sum of electron pair energies

$$E_{MP2} = \sum_i j e_{ij} \tag{3.73}$$

$$e_{ij} = \left(2t_{ij}^{ab} - t_{ij}^{ba}\right) (ia \mid jb) \tag{3.74}$$

where the amplitudes t_{ijab} are computed according to Eq. ... . The occupied molecular orbitals ij are localized using e.g. Foster-Boys, Pipiek-Mezey or a Cholesky decomposition of the density matrix. Virtual orbitals are generally localized by projection onto the atomic orbital space (PAOs) and subsequently assigning them to pair domains $[ij]$ (domain specific virtuals), or by diagonalizing the MP2 density matrix for each electron pair (pair natural orbitals). In all cases, the number of SVOs scales as $\mathbf{O}(1)$ for each electron pair ij, in the limit of large molecules (see ...).

For well localized orbitals, the electron pair correlation e_{ij} decays with $1/r_{ij}^6$ whith the distance between orbital centres. Electron pairs are generally divided into four groups: strong pairs ($r_{ij} < 1a_0$), weak pairs ($1 < r_{ij} \leq 8a_0$), distant pairs ($8 < r_{ij} \leq 15a_0$), and very distant pairs ($15 < r_{ij}$) (Sch1999). Other than by distance criteria, electron pairs can also be grouped by their pair energy (Nee2009). Figure ... shoes the number of significant electron pairs in each category for glycine chains. The number of strong, weak and distant pairs scale as $\mathcal{O}(N)$, while the number of very distant pairs scales quadratically.

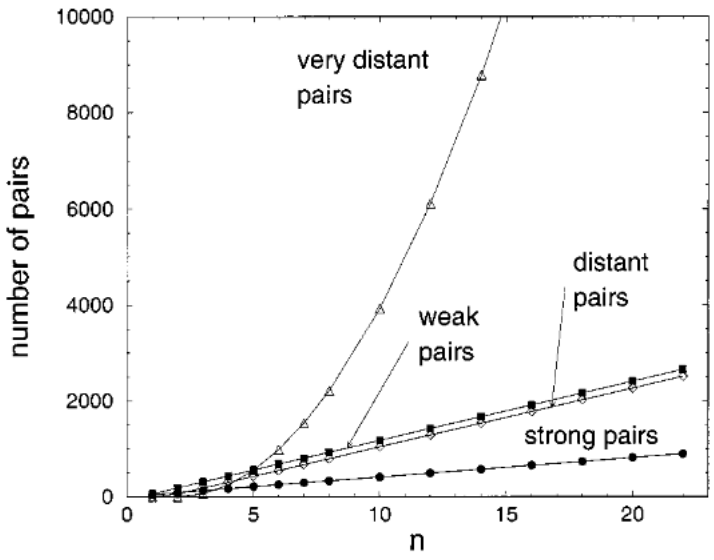


Figure 3.1: Taken from Sch1999

The major bottle-neck in LMP2 is, as usual, the transformation of the 2 electron integrals from the AO basis into the local basis

$$\left(\underline{i}\overline{a} \mid \underline{j}\overline{b}\right) = L_{\mu\underline{i}} L_{\sigma\overline{a}} \left(\mu\sigma \mid \nu\lambda\right) L_{\nu\underline{j}} L_{\lambda\overline{b}} \tag{3.75}$$

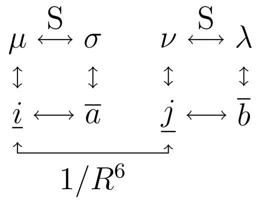
The expression above translates into the sparsity diagram

$$\begin{array}{ccccc} \mu & \overset{\mathrm{S}}{\longleftrightarrow} & \sigma & \quad\quad & \nu & \overset{\mathrm{S}}{\longleftrightarrow} & \lambda \\ \updownarrow & & \updownarrow & & \updownarrow & & \updownarrow \\ \underline{i} & \longleftrightarrow & \overline{a} & & \underline{j} & \longleftrightarrow & \overline{b} \end{array}$$

which indicates that the MO integrals can be evaluated with $\mathcal{O}(N^2)$ effort without further approximations. One thing to note is that the quadratic scaling is also obatined, even if the sparsity relationships $i \leftrightarrow a$ and $j \leftrightarrow b$ did not exist, i.e. where virtual orbitals are localized, but not grouped into (apir) domains. The major disadavantage of such non-pair specific methods is that the virtual orbital space is less compact, which leads to a high overhead for integral transformation involving virtual orbitals, which could be the reason that there are no examples in literature using such a scheme. Establishing an a priori sparsity relationship between occupied and virtual space allows to more easily reach the low-scaling regime.

Linear Scaling LMP2

It has been found [Sae1987] early on that the quadratic scaling very distant electron pairs can safely ignored without major impact on the total correlation energy. (Distance or Connectivity ciriteria) Distant pairs can also be approximated either by a multipole expansion (Het1998) or empirically (Rau1995), which further lowers the prefactor of the method. As a consequence, this establishes a sparsity relationship between i and j, and the sparsity diagram for the MO integrals becomes fully connected



and linear scaling LMP2 therefore becomes possible (Sch1999).

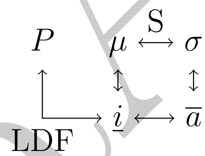
Instead of using distance criteria, can use screening:

Density Fitting for LMP2

While specific virtual orbitals form a very compact representation of the virtual space, the fact that each electron pair has their own orthogonal vitual orbital basis means that the total number of virtuals can become exceedingly large, and consequently increases the cost associated with the AO-MO transformation step. The most expensive step then becomes

$$(\underline{i}\overline{a} \mid P) = L_{\mu\underline{i}} \, (\mu\nu \mid P) \, L_{\nu\overline{a}} \tag{3.76}$$

Transformation of the 3c2e integrals scales with $\mathcal{O}(N^2)$. Linear scaling can be achieved by instroducing an orbital-specific fitting domain $[i]_f it$, e.g. by assigning all auxiliary functions P on atoms with a Mulliken charge above a given threshold for the local orbital i (Pin2015), or by using a Boughton-Pulay like scheme (Wer2015). This yields the sparsity diagramm



As opposed to SOS-AO-MP2, where density fitting can give a more favourable factorization of the energy expression, the MO integrals need to fully assembled for LMP2 in order to solve the linear equations (...). The assembly is done in two steps

$$B^X_{\underline{i}\overline{a}} = \sum_{Y \in [i]_{fit} \cup [j]_{fit}} (X \mid Y)^{-1/2} \, (Y \mid \underline{i}\overline{a}) \tag{3.77}$$

$$\left(\underline{i}\overline{a} \mid \underline{j}\overline{b}\right) = \sum_{X \in [i]_{fit} \cup [j]_{fit}} B^X_{\underline{i}\overline{a}} B^X_{\underline{j}\overline{b}} \tag{3.78}$$

The two steps are repeated for each electron pair ij, and the sum runs over all auxiliary functions P in the unified fitting domain $[i]_{fit} \cup [j]_{fit}$, which enforces linear scaling for these steps as well.

3.2.3 Atomic Orbital CC2

The atomic orbital approach for low scaling MP2 can be extended to CC2 ... currently under work

3.2.4 Atomic Orbital CCSD: PAO or AO?

!!!!!!!!!!!! REWRITE !!!!!!!!!!!!!!!! As opposed to MP2 and CC2, where the energy denominator in the closed form expression for the doubles amplitudes t_{iajb} makes it difficult to find an orbital-invariant representation of the energy expressions, it is much easier to do so for CCSD. The restricted CCSD correlation energy expression reads

$$E_{corr} = \sum_{ijab} v_{iajb} t_{iajb}^{CCSD} \qquad v_{iajb} = 2\,(ia \mid jb) - (ib \mid ja) \tag{3.79}$$

In their 1999 paper [ref], Scuseria and Ayala have shown that there are two possible "AO" representations of the CCSD correlation energy. The first and most straight-forward approach is to factor out the MO coefficient matrices from the MO electron repulsion integrals v, similar to AO-MP2

$$\begin{aligned} E_{corr} &= \sum_{ijab} \sum_{\mu\nu\sigma\lambda} C_{\mu i} C_{\sigma a} v_{\mu\sigma\nu\lambda} C_{\nu j} C_{\lambda b} t_{iajb} \\ &= \sum_{\mu\nu\sigma\lambda} v_{\mu\sigma\nu\lambda} t_{\underline{\mu}\overline{\sigma}\underline{\nu}\overline{\lambda}} \end{aligned} \tag{3.80}$$

whith the amplitudes $\mathbf{t}$ recast in a PAO-like basis:

$$t_{\underline{\mu}\overline{\sigma}\underline{\nu}\overline{\lambda}} = C_{\mu i} C_{\sigma a} t_{iajb} C_{\nu j} C_{\lambda b} \tag{3.81}$$

The MO amplitudes can be recovered with

$$t_{iajb} = \overline{C}_{\mu i} \overline{C}_{\sigma a} t_{\underline{\mu}\overline{\sigma}\underline{\nu}\overline{\lambda}} \overline{C}_{\nu j} \overline{C}_{\lambda b} \tag{3.82}$$

It should be noted that $\underline{\mu}$, $\overline{\sigma}$, $\underline{\nu}$ and $\overline{\lambda}$ do not correspond to the definition of PAOs given in Section .... As opposed to standard PAOs, they lack the overlap matrix $\mathbf{S}$ in the MO-PAO transformation and vice-versa (compare Equation ... with Eqaution ...). Here, the occupied and virtual PAO-like spaces are not mutually orthogonal, but related through the overlap matrix, similar to the pseudo-PAOs used in AO-MP2

$$\mathbf{PSQ} = \mathbf{0} \qquad \mathbf{P} + \mathbf{Q} = \mathbf{S}^{-1} \tag{3.83}$$

This alternative PAO-like formulation will be referred to as *mutually non-orthogonal projected atomic orbitals* (or mnoPAOs) from here on out.
Instead of using this mnoPAO approach, Scuseria and Ayala proposed an alternative formulation where the coefficient matrices are factored out from the t-amplitudes instead, using the PAO backtransformation 2.73:

$$\begin{aligned} E_{corr} &= \sum_{ijab} \sum_{\mu\nu\sigma\lambda} v_{iajb} C_{\mu i} C_{\sigma a} \theta_{\underline{\mu}\overline{\sigma}\underline{\nu}\overline{\lambda}} C_{\nu j} C_{\lambda b} \\ &= \sum_{\mu\nu\sigma\lambda} \Pi_{\underline{\mu}\overline{\sigma}\underline{\nu}\overline{\lambda}} \theta_{\underline{\mu}\overline{\sigma}\underline{\nu}\overline{\lambda}} \end{aligned} \tag{3.84}$$

The PAO-amplitudes θ are related to the MO quantities by

$$\theta_{\underline{\mu}\overline{\sigma}\underline{\nu}\overline{\lambda}} = \overline{C}_{\mu i} \overline{C}_{\sigma a} t_{iajb} \overline{C}_{\nu j} \overline{C}_{\lambda b} \tag{3.85}$$

$$t_{iajb} = C_{\mu i} C_{\sigma a} \theta_{\underline{\mu}\overline{\sigma}\underline{\nu}\overline{\lambda}} C_{\nu j} C_{\lambda b} \tag{3.86}$$

The mnoPAO and PAO t-amplitudes are related by

$$\theta_{\underline{\mu}\overline{\sigma}\underline{\nu}\overline{\lambda}} = S_{\mu\mu'} S_{\overline{\sigma}\overline{\sigma}'} t_{\underline{\mu}'\overline{\sigma}'\underline{\nu}'\overline{\lambda}'} S_{\underline{\nu}\underline{\nu}'} S_{\overline{\lambda}\overline{\lambda}'} \tag{3.87}$$

$$\theta_{\underline{\mu}\overline{\sigma}\underline{\nu}\overline{\lambda}} = P_{\mu\mu'} P_{\overline{\sigma}\overline{\sigma}'} \theta_{\underline{\mu}\overline{\sigma}\underline{\nu}\overline{\lambda}} P_{\underline{\nu}\underline{\nu}'} P_{\overline{\lambda}\overline{\lambda}'} \tag{3.88}$$

!!!!! REWRITE !!!!!!!!!!

3.2.5 Local Coupled Cluster

3.2.6 FNO Coupled Cluster ??

3.3 Critical Stance on AO vs LMO

Distance criteria, or Mulliken/Löwdin pouplation AO only for closed expressions (not for CCSD++) LOcal: ij cirteria, ij-¿ab criteria

Chapter 4

Local Correlation: Excited State

The previous chapter demonstrated how it is possible to achieve low to linear computational complexity for Hartree Fock as well as Møller-Plesset perturbation theory and Coupled Cluster. Highly optimized code has been developped for all of them, exploiting distributed memory parallelism and/or accelerators such as GPUs. The local treatment of electron correlation can naturally be extended to excited states as well. The development of low-scaling excited states has been steadily progressing since the early 2000s, and many competing flavors have emerged over the years both for the Algebraic Diagrammtic Construction method, Coupled Cluster linear response, and Equation-of-Motion Coupled Cluster. Local excited state methods need to take into considerations both local electron correlation, as well as the local character of the excited state itself.

4.1 Low Scaling ADC, CCLR and EOM-CC using Molcular Orbitals

Virtually all existing low-scaling implementations of ADC, CCLR and EOM-CCSD use some form of local or compact molecular orbital representation, to varying degrees of success. The major problem that these methods face is the non-locality of certain excited states such as charge transfer states (Figure ...), which can involve occupied and virtually orbitals which are localized on entirely different parts in the system. Clearly, truncating virtual orbitals spatially is no longer a valid option, and makes a straight-forward extension of LMO-methods difficult, becuase they cut out far-away contributions. Similarly, the excited state is often not properly described by the electronic ground state (pair-)densities and their associated (pair) natural representation. Over the years, various strategies have been proposed to adapt exisiting LMO, NO and PNO schemes to excited states as well.

4.1.1 Orbital Invariant ADC/CCLR/EOM-CCSD

Similarly to ground state MP and CC methods, the working equations can be reformulated to work with any orbital representation. Consider for example the general ADC(2)

working excitations for computing the singles and doubles vectors

$$\begin{aligned} r_{ia} &= A_{ia,jb}u_{jb} + A_{ia,jbkc}u_{jbkc} \\ r_{iajb} &= A_{iajb,kc}u_{kc} + A_{iajb,kcdl}u_{kcdl} \end{aligned} \tag{4.1}$$

Transforming to the local basis is straight-forward: the trial vectors u_{ia}, as well as the molecular electron integrals $(ia \mid jb)$ and MP2 amplitudes t_{iajb} in the Jacobian $\mathbf{A}$ (see Section ...) are simply replaced by their local counterparts $u_{\underline{i}\overline{a}}$, $\left(\underline{i}\overline{a} \mid \underline{j}\overline{b}\right)$ and $t_{\underline{i}\overline{a}\underline{j}\overline{b}}$. The local MP2 amplitudes are computed using either the Hylleraas functional (Equation ...) or the Laplace transform (Equation ...). Similar things apply for an orbital-invariant formulation of CCLR and EOM-CCSD.

ADC(2) and CC2 variants also allow an on-the-fly computation of the doubles part (see ...). Here, the orbital invariant formaulation becomes less straight-forward becuase the doubles-doubles block of the ADC and CC2 Jacobian matrix is no longer diagonal, similar to the Fock matrix. Fortunately, the Laplace transform can be applied to circumvent this problem, in this case shown for ADC(2)

$$\begin{aligned} r_{ia}(\omega) &= A_{ia,jb}u_{jb} + A_{ia,jbkc}\frac{A_{iajb,kc}u_{kc}}{\omega - \epsilon_a - \epsilon_b + \epsilon_i + \epsilon_j} \\ &= A_{ia,jb}u_{jb} - \sum_{\alpha}^{n} |w^{(\alpha)}| e^{(\omega-\epsilon_a-\epsilon_b+\epsilon_i+\epsilon_j)tpa} A_{ia,jbkc}A_{,kc}u_{kc} \end{aligned} \tag{4.2}$$

Using a similar apporach to Equation ..., the local ADC(2) equations are then given by

$$r_{\underline{i}\overline{a}}(\omega) = A_{\underline{i}\overline{a},\underline{j}\overline{b}}u_{\underline{j}\overline{b}} - A_{\underline{i}\overline{a},\underline{j}\overline{b}\underline{k}\overline{c}} \sum_{\alpha}^{n} e^{\omega t^{(\alpha)}} X^{(\alpha)}_{\underline{j}\underline{j}'} Y^{(\alpha)}_{\overline{b}\overline{b}'} A_{\underline{j}'\overline{b}'\underline{k}'\overline{c}',\underline{l}\overline{d}} u_{\underline{l}\overline{d}} X^{(\alpha)}_{\underline{k}\underline{k}'} Y^{(\alpha)}_{\overline{c}\overline{c}'} \tag{4.3}$$

where the Laplace matrices $\mathbf{X}$ and $\mathbf{Y}$ read

$$X^{(\alpha)}_{\underline{i}\underline{k}} = \sum_{i} U_{i\underline{i}} |w'^{(\alpha)}|^{1/4} e^{\epsilon_i t'^{(\alpha)}} U_{k\underline{k}} \tag{4.4}$$

$$Y^{(\alpha)}_{\overline{a}\overline{c}} = \sum_{a} U_{a\overline{a}} |w'^{(\alpha)}|^{1/4} e^{-\epsilon_a t'^{(\alpha)}} U_{c\overline{c}} \tag{4.5}$$

Note that the Laplace parameters $w'^{(\alpha)}$ and $t'^{(\alpha)}$ are different from the ones used in the MP2 amplitudes (—), due to the presence of the eigenvalue *omega* in the denominator. Each time the eigenvalue changes, the Laplace parameters need to be recomputed to obtain an accurate approximation.

An orbital invariant reformulation is not needed by every method. NOs, PNOs and NTOs can be *canonicalized* by diagonalizing the occupied-occupied and virtual-virtual block of the Fock matrix in the truncated NO/PNO/NTO basis to get a smaller set of canonical molecular orbitals and orbital energies. Because these types of representations generally do not depend on distance criteria, they are unaffected by the delocalized nature of the CMOS, as they seek compactness rather than locality.

4.1.2 State Specificity for Local Molecular Orbitals

The most challenging part in extending domain-specific virtual orbital methods to excited states lies in determing a suitable excitation domain in which to expand the virtual space. The first implementations of local excited state EOM-CCSD (Kor2003,Cra2002) and CC2-LR (Kat2003) constructed the domains using a Mulliken-charge like analysis of the CIS coeffcients $r_i a$. The CIS coefficients are first transformed to the LMO-PAO basis

$$r_{\underline{i}A} = U_{\underline{i}i} r_{ia} Q_{Aa} \tag{4.6}$$

To determine the importance w of each LMO/PAO, the squares of the norms of the coefficients are summed up row- and column-wise

$$\begin{aligned} w_{\underline{i}} &= \sum_A |r_{\underline{i}A}|^2 \\ w_A &= \sum_{\underline{i}} |r_{\underline{i}A}|^2 \end{aligned} \tag{4.7}$$

The LMOs/PAOs are then ordered by decreasing weight. Their weights are then summed up until a certain threshold T_{LMO}/T_{PAO} is reached (typically around 0.995 to 0.9999). The excited state orbital domains $[\underline{i}]_{ES}$ containg the relevant virtual orbitals are then contructed by applying the BP algorithm to a set of "excited natural orbitals" (Kor2003)

$$\phi^*_{\underline{i}} = \sum_{\overline{a}} r_{\underline{i}A} \phi_A \tag{4.8}$$

The full orbital domain of i is then given as the union of its ground-state and excited state domain $[\underline{i}] = [i]_{GS} \cup [i]_{ES}$. The virtual orbital weights w_A can be used to impose further restrictions on the virtual orbital space. Finally, the pair domains ij are formed as the union $[\underline{i}] \cup [\underline{j}]$. In general, only the computation of the doubles part, which is time-determining, is subject to domain-restrictions, while the singles part is computed without domain lists.

The method however has the major flaw that the orbital domains are highly senitive to the CIS transition density, which does not describe the excited state very accurately. Some orbitals can be dropped in the domain construction which might become important for doubles contributions. LMO methods face an interesting chicken-or-egg problem where they need information from the excited state wave function, to accurately compute properties of said function. There are several ways to mitigate this problem. In their local CC2-LR implementation, Kats and Schütz (Kat2009) use the Laplace transform (Equation ...) to recompute the doubles amplitudes on the fly, which allows to adapt the excited state domains dynamically during the optimization procedure. Starting from the CIS transition density, the domains are recomputed at each step by analyzing the state vector $r_{\underline{i}A}(\omega)$ as described above. This greatly increased accuracy compared to canonical calculations with energy differences well below 0.1 eV.

Mester et al. proposed a more pragmatic approach, where they first analyse the CIS state vector to extract the important LMOs and PAOs. They then augment the domains $[i]_{EX}$ by adding all remaining molecular orbitals that has a significant Mulliken charge on an atom that is also significant for i. This is based on the assumption that, although CIS might not be a good approximations, the important orbitals should still be close by.

Nonetheless, the LMO method is again plagued by spurious distance dependent thresholds and Mulliken charge thresholds. Nowadays, local excited state methods are mostly dominated by PNOs, NOs, or NTOs.

4.1.3 State Specificity for Natural Orbitals

NO methods achieve performance by dropping virtual natural orbitals with low occupation numbers. The first implementations of EOM-CC and CCLR in the NO representation used natural orbitals obtained from the diagonalization of the ground state MP2 density matrix (Lan2010,Kum2017). The excited state character was not taken into account, but still a reasonable speed-up could be observed. However, it was shown (Kum2017) that properties like the polarizability are much more sensitive to the truncation of the virtual orbitals than the ground state correlation energy, with the error increasing linearly as a function of the number of dropped virtual natural orbitals. While VNOs with low occupation numbers, i.e. diffuse character, can be safely ignored for the ground state correlation energy, diffuse VNOs play a much more important role for response properties, and hence fewer VNOs can be omitted. Better results could be obtained by simply truncating the virtual CMOs instead, which invalidates the use of VNOs.

In their NO-CC2 and NO-ADC(2) implementations, Mester et al. (Mes2017, Mes2018, Mes2019) proposed to compute a set of occupied and virtual NOs by diagonalizing occupied and virtual state-averaged densities

$$\mathbf{D}_{ij} = \frac{1}{2}\left(\mathbf{D}_{ij}^{MP2} + \mathbf{D}_{ij}^{CIS(D)}\right) \tag{4.9}$$

$$\mathbf{D}_{ab} = \frac{1}{2}\left(\mathbf{D}_{ab}^{MP2} + \mathbf{D}_{ab}^{CIS(D)}\right) \tag{4.10}$$

where $\mathbf{D}^{MP2}$ is the MP2 ground state density and $\mathbf{D}^{CIS}$ is the state-specific CIS(D) excited state density. Their restricted expressions read

$$D_{ij}^{MP2} = \sum_{kab}\left(2t_{ik}^{ab}t_{jk}^{ab} - t_{ik}^{ab}t_{jk}^{ab}\right) \tag{4.11}$$

$$D_{ab}^{MP2} = \sum_{ijc}\left(2t_{ij}^{ca}t_{ij}^{cb} - t_{ij}^{ca}t_{ij}^{bc}\right) \tag{4.12}$$

$$D_{ij}^{CIS(D)} = \sum_{a} c_i^a c_j^a + \sum_{kab}\left(2t_{ik}^{ab}t_{jk}^{ab} - t_{ik}^{ab}t_{jk}^{ab}\right) \tag{4.13}$$

$$D_{ab}^{CIS(D)} = \sum_{i} c_i^a c_i^b + \sum_{ijc}\left(2c_{ij}^{ca}c_{ij}^{cb} - c_{ij}^{ca}c_{ij}^{bc}\right) \tag{4.14}$$

where c_i^a are the CIS coefficients and the c_{ij}^{ab} are the CIS(D) doubles coefficients

$$c_{ij}^{ab} = \frac{\sum_c \left[(ac \mid bj)\, c_i^c + (ac \mid bi)\, c_j^c\right] - \sum_k \left[(kj \mid ai)\, c_k^b + (kj \mid bj)\, c_k^a\right]}{D_{ij}^{ab} + \omega_{CIS}} \tag{4.15}$$

The state-averaged density needs to be recomputed and diagonalized for each state because $\mathbf{D}^{CIS(D)}$ depends on the excitation energy ω. While the CIS(D) density is much

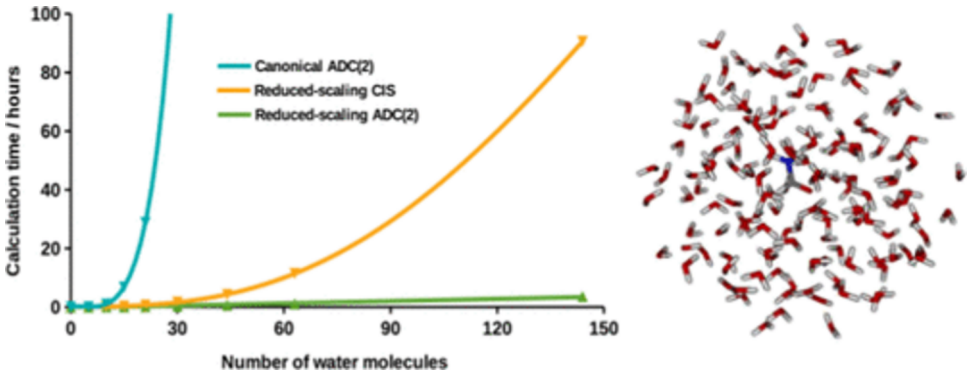


Figure 4.1: This is actually really cool

easier to compute than the ADC(2) or CC2-LR state density, it still scales with $\mathcal{O}(N^5)$. To mitigate the computational complexity, the density is constructed in a truncated orbital space: first, a set of occupied and virtual LMOs are chosen according to the CIS weighting criteria w described in the previous section. The basis is augmented by spatially close orbitals, and then canonicalized to yield a highly compact orbital molecular space which lowers the cost of constructing the CIS(D) densities.

In combination with natural auxiliary functions, this hybrid NO-LMO scheme can reduce the timings for CC2 and ADC(2) to such a drastic extent that the CIS pre-iterations become the time-determining step (Figure ...), with an additional error of only 2-4 meV. The reduced scaling however comes at a high prefactor for computing multiple different excitation energies.

4.1.4 State Specificity for Pair Natural Orbitals

Pair natural orbital methods face the same problems as NOs, where PNOs with low occupation numbers are considerably more important for response properties than for ground state properties. In a similar vein, excited state PNOs can be generated by considering lower level excited state electron pair densities (Hel2011). PNO methods have been succuessfully extended to ADC(2), CC2-LR (Hel2013), ADC(2)-x (Hel2014) and CCSD-LR (ref) by using CIS(D) or CIS(D)-like densities

$$D_{ij}^{ab} = \sum_{c} \left(2b_{ij}^{ab} - b_{ij}^{ba}\right) b_{ij}^{ab} + \left(2b_{ij}^{ab} - b_{ij}^{ba}\right) b_{ij}^{ba} \tag{4.16}$$

where $\mathbf{b}_{ij}$ are state-specific modified pair amplitudes which are not uniquely defined. Again, these methods come at the cost of a higher prefactor due to the relatively high cost of constructing PNOs. Nonetheless, it was shown that the computational complexity can be lowered to $\mathcal{O}(N^3)$ for PNO-CCSD-LR.

Efforts have also been made to develop PNO response methods which are more economical for computing larger excitation manifolds by removing the state-specificity. Instead of taking individual excited state densities, Peng et al. (Pen2018) proposed to generate a set of *state-averaged* PNOs obtained by diagonalization of the average excited

state density over an N-state manifold

$$\mathbf{D}_{ij} = \frac{1}{N} \sum_{k}^{N} \mathbf{D}_{ij}^{(k)} \tag{4.17}$$

A production-qaulity implementation has not yet been shown which uses this approach.

In their perturbed pair-natural orbital (PNO++) approach for CCLR, Cunha and Crawford (Cun2021) incorporate the external perturbation into the electron pair density

$$D_{ij}^{ab} = \sum_{c} \left(2x_{ij}^{ab} - x_{ij}^{ba}\right) x_{ij}^{ab} + \left(2x_{ij}^{ab} - x_{ij}^{ba}\right) x_{ij}^{ba} \tag{4.18}$$

where $\mathbf{x}$ are perturbed amplitudes given by

$$x_{ij}^{ab} = \frac{\overline{B}}{\overline{H}_{aa} + \overline{H}_{bb} - \overline{H}_{ii} - \overline{H}_{jj} + \omega} \tag{4.19}$$

with an external perturbation $\overline{B}$ and the similarity transformed Hamiltonian $\overline{H}$.

Finally, there are also the *back-transformed* PNOs, or bt-PNOs, where the ground state PNO-quantities like the amplitudes are transformed back to the canonical basis and used in the canonical working equations (Dut2016).

In the end, most local excited state methods using natural orbitals differ by how they redefine the amplitudes $\mathbf{t}$ for the individual excited states or the whole perturbed molecular system. It is still very much an active field of research.

4.1.5 State Specificity for Natural Transition Orbitals

The last method to obtain a compact representation of excited states is via natural transition orbitals. NTOs are the equivalent of NOs for excited states, and represent a compact representations of their dominant contribution (Figure ...). Baudin and Kristensen have developped two different CC2LR schemes based on NTOs called LoFEX (local framework for calculating excitation energies) (Bau2016) and CornFLEX (correlated natural transition orbital framework for calculating excitation energies) (Bau2017).

Again, one needs information avout the excited state to effciently compute its properties. The LoFEX method starts with a time-dependent Hartree Fock caculation and generates a set of NTOs by decomposition of the TDHF transition vectors $\mathbf{r}$ by diagonalization

$$\mathbf{r}\mathbf{r}^{\dagger}\mathbf{U} = \lambda_o \mathbf{U} \tag{4.20}$$

$$\mathbf{r}^{\dagger}\mathbf{r}\mathbf{V} = \lambda_v \mathbf{V} \tag{4.21}$$

Equations ... are alternative ways to compute the occupied and virtual NTO transformation matrices $\mathbf{U}$ and $\mathbf{V}$, rather than by singular value decomposition. A set of dominant NTO pairs is then chosen for which their occupation numbers are above a given treshold τ_{LoFEX}. The non-dominant NTOs are not discarded, but rather localized. The idea is to construct a sourrounding excitation orbital space (XOS) containing LMOs that are important for correlation effects of the NTOs. A first guess to the XOS is chosen based on distance criteria and Löwdin charges. The CC2LR are then solved in that basis, and new NTOs

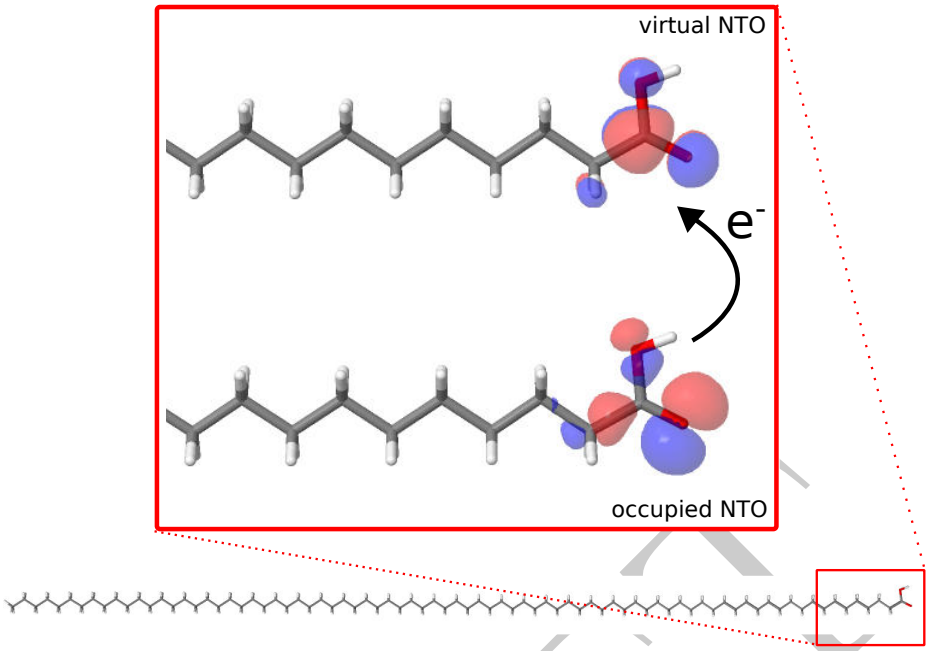


Figure 4.2: Dominant natural transition orbital pair for the lowest excitation of the carboxylic acid $C_{79}H_{159}COOH$ ($\pi \rightarrow \pi^*$ transition). The span of the NTOs is very small compared to the rest of the molecule, and the compactness can be used to drastically speed up excited state caculations.

are computed from the CC2 transition vector and added to the XOS. This procedure is repeated until the excitation energy ω for that state has converged. While the guess XOS is first formed using distance cirtieria, the subsequent optimization procudure makes the method much more robust and black-box. Even for relatively small molecules, LoFEX can obtain considerable speed-ups. The main disadvantage is that LoFEX does not give any leverage for very delocalized excitations.

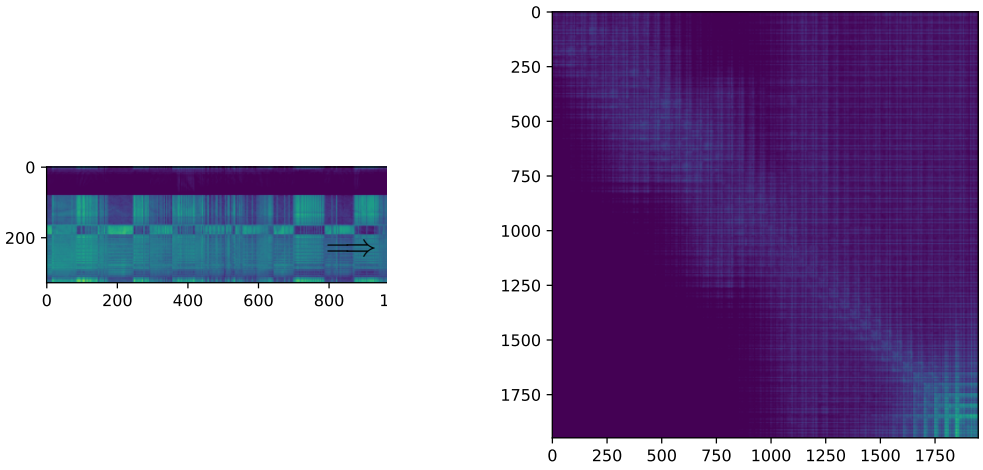
The CornFLEX method constructs a set of CIS(D)-like NTOs (CIS(D')-NTOs) which is obtained from diagonalizing a CIS(D)-like density matrix in the CIS-NTO basis. As opposed to CIS-NTOs, the CIS(D') NTOs also include correlation effects and are a more robust representation that the simple ad-hoc extension of CIS-NTOs using LMOs. Speed-ups can be observed in CornFLEX even for delocalized excitations.

4.2 Atomic Orbital Configuration Interaction Singles

-¿ AO TDSCF (Kussmann) -¿ TDHF and RPA are equivalent in linear response regime.

The methods presented in the previous section all work similarly. They first start by approximating the targeted excited states with a lower level of theory using CIS or CIS(D). They then solve higher order equations in the basis obtained from that approximation and may also dynamically augment the correlation domain while optimizing the excitation energies. The methods work on the principle of orbital *compactness* rather than sparsity

At the moment of writing, CIS is the only excited state method which is traditionally evaluated in the AO basis. While CIS does not give qualitatively good results, it is still



(a) (b)

Figure 4.3: Logarithm of the absolute values of the matrix elements in the transition densities in the MO (left) and AO basis (right) for the lowest excited state for the carboxylic acid $C_{79}H_{159}COOH$. The excitation domain is entirely localized on the carboxylic group. Using sparse matrix algebra, significant speed-ups can be obtained for CIS in the AO basis

a very important stepping stone for higher order methods, as was demonstrated in the previous section. Omitting the zero-order contributions, the CIS working equations are given by

$$u_{ia} = [2\,(ia \mid jb) - (ib \mid ja)]\,u_{jb} \tag{4.22}$$

Factoring out the MO coefficient matrices:

$$\begin{aligned} u_{ia} &= C_{\mu i} C_{\sigma a}\,[2\,(\mu\sigma \mid \nu\lambda) - (\mu\lambda \mid \nu\sigma)]\,C_{\nu j} C_{\lambda b} u_{jb} \\ &= C_{\mu i} C_{\sigma a}\,[2\,(\mu\sigma \mid \nu\lambda) - (\mu\lambda \mid \nu\sigma)]\,P_{\nu\lambda} \end{aligned} \tag{4.23}$$

where $\mathbf{P}$ is the non-symmetric transition density in the AO basis. The CIS working equations can be reduced to the construction of a "pseudo"-Fock matrix which has a coulomb and an exchange part. The Fock matrix is then transformed to the MO basis:

$$F_{\mu\nu} = J_{\mu\nu} + K_{\mu\nu} \tag{4.24}$$

$$u_{ia} = C_{\mu i} F_{\mu\nu} C_{\mu a} \tag{4.25}$$

For localized excitations, the AO transition density is sparse (Figure ...), and similar approximation can be used as in Hartree Fock, e.g. LinK, CFMM, or LDF. CIS can therefore be evaluated with $\mathcal{O}(N)$ computational effort.

Chapter 5

The Spin-Opposite Scaled Algebraic Diagrammatic Construction Method in the Atomic Orbital Basis

The Algebraic Diagrammatic Construction method can be considered as Møller Plesset for excited states. It is therefore not surprising that local correlation method for MP can also be applied to MP. In chapter ..., it has been shown that local approximations for the ground state can be grouped into 3 categories: atomic orbitals, local orbitals and natural orbitals. Only the latter two have been used in the context of ADC as discussed in chapter .... An atomic orbital representation of ADC has not yet been considered in literature, and will be the subject of this chapter. First, the restricted doubles-folded ADC working equations are derived. Then the SOS approximation is applied. Finally, the restricted SOS-ADC working equations are derived in the AO basis, with and without density fitting.

5.1 Working Equations for Restricted ADC with Doubles-Folding

The eigenvalue problem in the algebraic diagrammatic construction method truncated at doubles excitations takes the form

$$\begin{bmatrix} \mathbf{A}_{SS} & \mathbf{A}_{SD} \\ \mathbf{A}_{DS} & \mathbf{A}_{DD} \end{bmatrix} = \begin{bmatrix} \mathbf{v}_S \\ \mathbf{v}_D \end{bmatrix} \boldsymbol{\Omega} \tag{5.1}$$

where $\mathbf{A}$ is the symmetric Jacobian ADC matrix with the singles-singles (SS), doubles-singles (DS), singles-doubles (SD) and doubles-doubles (DD) sub-blocks, with the eigenvectors $\mathbf{v}$ and the diagonal eigenvalue matrix Ω. The eigenvalue problem in Equation ... is generally solved using the Davidson diagonalization procedure (A2) to extract the first few lowest eigenvalues. Rather than constructing the entire Jacobian matrix which scales as $\mathcal{O}(N^8)$ using Equations ... to ..., the Davidson method computes the matrix-vector products $\mathbf{r} = \mathbf{A}\mathbf{u}$ with the current trial vectors $\mathbf{u}$ using a closed-form expression, which reduces the computational complexity to $\mathcal{O}(N^5)$. The MVPs can be expressed in block-

form as

$$\begin{aligned} \mathbf{r}_S &= \mathbf{A}_{SS}\mathbf{u}_S + \mathbf{A}_{SD}\mathbf{u}_D \\ \mathbf{r}_D &= \mathbf{A}_{DS}\mathbf{u}_S + \mathbf{A}_{DD}\mathbf{u}_D \end{aligned} \tag{5.2}$$

The trial vector space in the Davidson diagonalization scales with fourth order as o^2v^2, and can quickly become a memory bottle-neck for large molecules. As shown in the previous chapter, one can recompute the doubles-part of the MVP on-the-fly using *doubles-folding*

$$\mathbf{r}_S(\omega) = \mathbf{A}_{SS}\mathbf{u}_S + \frac{\mathbf{A}_{SD}}{\omega - \mathbf{DD}}\mathbf{u}_S \tag{5.3}$$

This trick is only possible for ADC(2)-s where the doubles-doubles block of the ADC matrix is diagonal. While the memory footprint for the diagonalization is reduced from o^2v^2 to ov, the MVP becomes dependent on the eigenvalue ω and a modified Davidson procedure needs to be used to solve this *pseudo*-eigenvalue value (A...).

The above expression is valid in the case of unrestriced ADC(2) where molecular *spin*-orbitals are assumed, rather than *spatial* orbitals. The working equations for standard ADC(2) and doubles-folded ADC(2) are given in Table ... . Implementations such as adcman in Q-Chem can use these formulae directly by delegating any considerations of spin-symmetry to a special tensor library called libtensor, which programmatically keeps track of the non-vanishing spin block components and reduces the expressions to the restricted ADC(2) equations for closed-shell molecules (Figure ...).

If no special block tensor library is used, it is numerically advantageous to split the ADC(2) matrix-vector products into their spin-components, and compute each block individually. Using a double-bar notation to indicate MOs with opposite spin $\sigma(i) \neq \sigma(\bar{i})$, the matrix-vector product can be written as

$$\begin{aligned} r_{ia}(\omega) =& (\epsilon_a - \epsilon_i)u_{ia} - \sum_{jb} \left[(ij \mid ab) - (ia \mid jb)\right] u_{jb} + \sum_{\bar{j}\bar{b}} \left(ia \mid \bar{j}\bar{b}\right) u_{\bar{j}\bar{b}} \\ &+ \sum_b I_{ab}u_{ib} + \sum_j I_{ij}u_{ja} - \frac{1}{2}\left[t_{ia\bar{j}\bar{b}} I^{(1)}_{\bar{j}\bar{b}} + \left(ia \mid \bar{j}\bar{b}\right) I^{(2)}_{\bar{j}\bar{b}}\right] \\ &- \frac{1}{2}\left[(t_{iajb} - t_{jaib})\, I^{(}1)_{jb} + ((ia \mid jb) - (ja \mid ib))\, I^{(2)}_{jb}\right] \\ &+ \sum_{kcl} u_{kalc}(\omega)\,(ik \mid cl) + \sum_{k\bar{c}\bar{l}} u_{ka\bar{l}\bar{c}}(\omega)\left(ik \mid \bar{c}\bar{l}\right) \\ &- \sum_{ckd} (ac \mid kd)\, u_{ikcd}(\omega) - \sum_{c\bar{k}\bar{d}} \left(ac \mid \bar{k}\bar{d}\right) u_{ic\bar{k}\bar{d}}(\omega) \end{aligned} \tag{5.4}$$

with the pre-iteration intermediates (computed only once)

$$\begin{aligned} I_{ab} =& \frac{1}{2}\sum_{kcl} \left[t_{kalc}\,(kb \mid lc) - t_{kalc}\,(kc \mid lb) + (ak \mid cl)\, t_{kblc} - (al \mid ck)\, t_{kb}lc\right] \\ &+ \frac{1}{2}\sum_{k\bar{c}\bar{l}} \left[t_{ka\bar{l}\bar{c}}\left(kb \mid \bar{l}\bar{c}\right) + \left(ak \mid \bar{c}\bar{l}\right) t_{kb}\bar{l}\bar{c}\right] \end{aligned} \tag{5.5}$$

$$\begin{aligned} I_{ij} = &\frac{1}{2}\sum_{ckd}\left[t_{ickd}\,(jc \mid kd) - t_{ickd}\,(jd \mid kc) + (ci \mid dk)\,t_{jckd} - (ck \mid di)\,t_{jckd}\right] \\ &+ \frac{1}{2}\sum_{\overline{\overline{ckd}}}\left[t_{i\overline{\overline{ckd}}}\left(jc \mid \overline{\overline{kd}}\right) + \left(ci \mid \overline{\overline{dk}}\right)t_{j\overline{\overline{ckd}}}\right] \end{aligned} \tag{5.6}$$

and the iteration intermediates which depend on the trial vectors $\mathbf{u}$ (computed at each Davidson iteration)

$$I^{(1)}_{ia} = \sum_{jb}\left[(jb \mid ia) - (ja \mid ib)\right]u_{jb} + \sum_{\overline{\overline{jb}}}\left(\overline{\overline{jb}} \mid ia\right)u_{jb} \tag{5.7}$$

$$I^{(2)}_{ia} = \sum_{jb}\left[t_{iajb} - t_{jaib}\right]u_{jb} + \sum_{\overline{\overline{jb}}} t_{ia\overline{\overline{jb}}}u_{\overline{\overline{jb}}} \tag{5.8}$$

The doubles components are computed on-the-fly and read

$$\begin{aligned} u_{iajb}(\omega) = \frac{1}{\omega - \epsilon_a - \epsilon_b + \epsilon_i + \epsilon_j}\Bigg\{&\sum_k\left[u_{ka}\,(ki \mid bj) - u_{ka}\,(kj \mid bi) - u_{kb}\,(ki \mid aj)\right. \\ &\left. +\; u_{kb}\,(kj \mid ai)\right] - \sum_c\left[u_{ic}\,(ac \mid bj) - u_{ic}\,(aj \mid bc) - u_{jc}\,(ac \mid bi) + u_{jc}\,(bc \mid ai)\right]\Bigg\} \end{aligned} \tag{5.9}$$

$$\begin{aligned} u_{ia\overline{\overline{jb}}}(\omega) = \frac{1}{\omega - \epsilon_a - \epsilon_b + \epsilon_i + \epsilon_j}\Bigg\{&\sum_k u_{ka}\left(ki \mid \overline{\overline{bj}}\right) + \sum_{\overline{\overline{k}}} u_{\overline{\overline{kb}}}\left(\overline{\overline{kj}} \mid ai\right) \\ &- \sum_c u_ic\left(ac \mid \overline{\overline{bj}}\right) - \sum_{\overline{\overline{c}}} u_{\overline{\overline{jc}}}\left(\overline{\overline{bc}} \mid ai\right)\Bigg\} \end{aligned} \tag{5.10}$$

The expression for the MVP for beta electrons ($r_{\overline{\overline{ia}}}$) is obtained by replacing alpha orbitals (i) by beta orbitals ($\overline{\overline{i}}$) and vice-versa in the expressions above. The off-diagonal blocks $r_{i\overline{\overline{a}}}$, i.e. the spins-flip states will not be considered here and are set to zero. For closed-shell molecules, the complexity of the formulas can be drastically reduced by introducing the following spin-symmetry relationships:

$$\begin{aligned} t_{iajb} = t_{ia\overline{\overline{jb}}} = t_{\overline{\overline{ia}}jb} = t_{\overline{\overline{ia}}\overline{\overline{jb}}} \\ (ia \mid jb) = \left(ia \mid \overline{\overline{jb}}\right) = \left(\overline{\overline{ia}} \mid jb\right) = \left(\overline{\overline{ia}} \mid \overline{\overline{jb}}\right) \end{aligned} \tag{5.11}$$

$$\begin{aligned} u_{ia} &= u_{\overline{\overline{ia}}} \qquad \text{if singlet} \\ u_{ia} &= -u_{\overline{\overline{ia}}} \qquad \text{if triplet} \end{aligned} \tag{5.12}$$

One then obtains two separate expressions for restricted ADC(2), depending on whether

singlet or triplet states are addressed

$$\begin{aligned} r^S_{ia}(\omega) = (\epsilon_a - \epsilon_i)u_{ia} - \sum_{jb} \left[2\,(ia \mid jb) - (ij \mid ab)\right] u^S_{jb} + \sum_b I_{ab}u_{ib} + \sum_j u^S_{ja} I_{ij} \\ - \frac{1}{2}\sum_{jb} \left[2t_{iajb} - t_{ibja}\right] I^{(1)S}_{jb} - \frac{1}{2} sum_{jb} \left[2\,(ia \mid jb) - (ib \mid ja)\right] I^{(2)S}_{jb} \\ + \sum_{kcl} (ik \mid lc) \left(2u^S_{kalc}(\omega) - u^S_{lakc}(\omega)\right) + \sum_{ckd} \left(2u^S_{ickd}(\omega) - u^S_{kcid}(\omega)\right) (kd \mid ac) \end{aligned} \tag{5.13}$$

$$\begin{aligned} r^T_{ia}(\omega) = (\epsilon_a - \epsilon_i)u^T_{ia} - \sum_{jb} (ij \mid ab)\, u^T_{jb} + \sum_b I_{ab}u^T_{ib} + \sum_j I_{ij}u^T_{ja} \\ + \frac{1}{2}\sum_{jb} t_{ibja} I^{(1)T}_{jb} + \frac{1}{2}\sum_{jb} (ib \mid ja)\, I^{(2)T}_{jb} \\ + \sum_{kcl} (ik \mid lc)\, u^T_{kalc}(\omega) + \sum_{ckd} \left[2u^T_{ickd}(\omega) - u^T_{idkc}(\omega) - u^T_{kcid}(\omega)\right] \end{aligned} \tag{5.14}$$

with the singlet and triplet doubles intermediates

$$\begin{aligned} u^{S,SOS}_{iajb}(\omega) = \frac{c_{osc}}{\omega - \epsilon_a + \epsilon_i - \epsilon_b + \epsilon_j} \Bigg\{ \sum_k \left[u^S_{ka}\,(ki \mid jb) + u^S_{kb}\,(kj \mid ai)\right] \\ - \sum_c \left[u^S_{ic}\,(jb \mid ac) - u^S_{jc}\,(ib \mid ac)\right] \Bigg\} \end{aligned} \tag{5.15}$$

$$u^T_{iajb}(\omega) = \frac{1}{\omega - \epsilon_a + \epsilon_i - \epsilon_b + \epsilon_j} \left\{ \sum_k u^T_{ka}\,(ki \mid bj) - \sum_c u^T_{ic}\,(ac \mid jb) \right\} \tag{5.16}$$

The pre-iteration intermediates are given by

$$\begin{aligned} I_{ab} &= \frac{1}{2}\sum_{kcl} \left[(2t_{kalc} - t_{kcla})\,(kb \mid lc) + (2t_{kblc} - t_{kclb})\,(ka \mid lc)\right] \\ &= \frac{1}{2}\left[\sum_{kcl} (2t_{kalc} - t_{kcla})\,(kb \mid lc)\right]_{a \leftrightarrow b} \end{aligned} \tag{5.17}$$

$$\begin{aligned} I_{ij} &= \frac{1}{2}\sum_{ckd} \left[(2t_{ickd} - t_{idkc})\,(jc \mid kd) + (2t_{jckd} - t_{jdkc})\,(ic \mid kd)\right] \\ &= \frac{1}{2}\left[\sum_{ckd} (2t_{ickd} - t_{idkc})\,(jc \mid kd)\right]_{i \leftrightarrow j} \end{aligned} \tag{5.18}$$

and are the same for both singlet and triplet expressions. The iterative intermediates however are split:

$$I^{(1)S}_{ia} = \sum_{jb} \left(2\,(ia \mid jb) - (ib \mid ja)\right) u^S_{jb} \tag{5.19}$$

$$I^{(2)S}_{ia} = \sum_{jb} \left(2t_{iajb} - t_{ibja}\right) u^S_{jb} \tag{5.20}$$

$$I_{ia}^{(1)T} = -\sum_{jb} (ib \mid ja)\, u_{jb}^{T} \tag{5.21}$$

$$I_{ia}^{(2)T} = -\sum_{jb} t_{ibja} u_{jb}^{T} \tag{5.22}$$

For the restricted equations ... to ..., the indices $ijkl...$ and $abcd...$ represent the occupied and virtual spin-integrated *spatial* molecular orbitals.

5.2 Working Equations for Restricted SOS-ADC(2) with Doubles-Folding

In the restricted ADC expressions for the matrix-vector product, the 4-index intermediates $u_{iajb}(\omega)$ need to be evaluated and temporarily stored, even if the density fitting approximation is used. This memory-intensive step can be avoided by using spin-opposite scaling (Section ...). Consider again the approximations introduced by the SOS method for the unrestricted ADC(2) matrix equations

- In the spin-amplitudes $\hat{t}_{iajb}$ the same-spin contributions are nulled and the opposite-spin contributions are scales by $c_o s$

$$\hat{t}_{SOS} = c_{os} \hat{t}_{iajb} \left(1 - \delta_{\sigma(i)\sigma(j)}\right) \tag{5.23}$$

- In the doubles-singles and singles-doubles block of the ADC matrix, some same-spin contributions are also removed, and the whole block is scaled by a different constant c_{osc}

$$C_{ia,kcld} = c_{osc} \left[\langle kl| \, |id\rangle \, \delta_{ac} - \langle kl| \, |ic\rangle \, \delta_{ac} - \langle al| \, |cd\rangle \, \delta_{ik} + \langle ak| \, |cd\rangle \, \delta_{il}\right] \left(1 - \delta_{\sigma(k)\sigma(l)}\right) \tag{5.24}$$

$$C_{iajb,kc} = c_{osc} \left[\langle kb| \, |ij\rangle \, \delta_{ac} - \langle ka| \, |ij\rangle \, \delta_{bc} - \langle ab| \, |cj\rangle \, \delta_{ik} + \langle ab| \, |ci\rangle \, \delta_{jk}\right] \left(1 - \delta_{\sigma(i)\sigma(j)}\right) \tag{5.25}$$

  Here, the function $\sigma(x)$ returns the spin of orbital x.

Applying the above constraints to Equation ... gives the spin components of the SOS-ADC(2) matrix-vector product with doubles-folding

$$\begin{aligned} r_{ia}^{SOS}(\omega) =& (\epsilon_a - \epsilon_i) u_{ia} - \sum_{jb} \left[(ij \mid ab) - (ia \mid jb)\right] u_{jb} + \sum_{\overline{\overline{jb}}} \left(ia \mid \overline{\overline{jb}}\right) u_{\overline{\overline{jb}}} \\ &+ \sum_{b} I_{ab}^{SOS} u_{ib} + \sum_{j} I_{ij}^{SOS} u_{ja} - \frac{1}{2} \left[t_{ia\overline{\overline{jb}}} I_{\overline{\overline{jb}}}^{(1)SOS} + \left(ia \mid \overline{\overline{jb}}\right) I_{\overline{\overline{jb}}}^{(2)SOS}\right] \\ &- \frac{1}{2} \left[(ia \mid jb) - (ja \mid ib)\right] I_{jb}^{(2)SOS} \\ &+ c_{osc} \left\{ \sum_{k\overline{\overline{cl}}} u_{ka\overline{\overline{lc}}}(\omega) \left(ik \mid \overline{\overline{cl}}\right) - \sum_{c\overline{\overline{kd}}} \left(ac \mid \overline{\overline{kd}}\right) u_{ic\overline{\overline{kd}}}(\omega) \right\} \end{aligned} \tag{5.26}$$

with the SOS pre-iteration intermediates

$$I_{ab}^{SOS} = \frac{1}{2} \sum_{\overline{\overline{kcl}}} \left[t_{ka\overline{\overline{lc}}} \left(kb \mid \overline{\overline{lc}} \right) + \left(ak \mid \overline{\overline{cl}} \right) t_{kb\overline{\overline{lc}}} \right] \tag{5.27}$$

$$I_{ij}^{SOS} = \frac{1}{2} \sum_{\overline{\overline{ckd}}} \left[t_{ic\overline{\overline{kd}}} \left(jc \mid \overline{\overline{kd}} \right) + \left(ci \mid \overline{\overline{dk}} \right) t_{jc\overline{\overline{kd}}} \right] \tag{5.28}$$

and the SOS iteration intermediates

$$I_{ia}^{(1)SOS} = \sum_{jb} \left[(jb \mid ia) - (ja \mid ib) \right] u_{jb} + \sum_{\overline{\overline{jb}}} \left(\overline{\overline{jb}} \mid ia \right) u_{\overline{\overline{jb}}} \tag{5.29}$$

$$I_{ia}^{(2)SOS} = \sum_{\overline{\overline{jb}}} t_{ia\overline{\overline{jb}}} u_{\overline{\overline{jb}}} \tag{5.30}$$

Only the opposite-spin components of the doubles components are needed:

$$\begin{aligned} u_{ia\overline{\overline{jb}}}^{SOS}(\omega) = \frac{c_{osc}}{\omega - \epsilon_a - \epsilon_b + \epsilon_i + \epsilon_j} \Bigg\{ & \sum_k u_{ka} \left(ki \mid \overline{\overline{bj}} \right) + \sum_{\overline{\overline{k}}} u_{\overline{\overline{kb}}} \left(\overline{\overline{kj}} \mid ai \right) \\ & - \sum_c u_{ic} \left(ac \mid \overline{\overline{bj}} \right) - \sum_{\overline{\overline{c}}} u_{\overline{\overline{jc}}} \left(\overline{\overline{bc}} \mid ai \right) \Bigg\} \end{aligned} \tag{5.31}$$

Finally, for closed-shell molecules, the matrix vector products for singlet and triplet excitations for restricted SOS-ADC(2) can be obtained by inserting the spin-symmetry relationships (...) into Equation (...)

$$\begin{aligned} r_{ia}^{S,SOS}(\omega) = {} & (\epsilon_a - \epsilon_i) u_{ia}^S - \sum_{jb} \left[2\,(ia \mid jb) - (ij \mid ab) \right] u_{jb}^S + \sum_b I_{ab}^{SOS} u_{ib} + \sum_j u_{ja}^S I_{ij}^{SOS} \\ & - \frac{1}{2} \sum_{jb} t_{iajb} I_{jb}^{(1)S,SOS} - \frac{1}{2} \sum_{jb} \left[2\,(ia \mid jb) - (ib \mid ja) \right] I_{jb}^{(2)S,SOS} \\ & + c_{osc} \left\{ \sum_{kcl} (ik \mid lc)\, u_{kalc}^{S,SOS}(\omega) - \sum_{ckd} u_{ickd}^{S,SOS}(\omega)\, (kd \mid ac) \right\} \end{aligned} \tag{5.32}$$

EQUATION

$$\begin{aligned} r_{ia}^{T,SOS}(\omega) = {} & (\epsilon_a - \epsilon_i) u_{ia}^S - \sum_{jb} (ij \mid ab)\, u_{jb}^T + \sum_b I_{ab}^{SOS} u_{ib} + \sum_j u_{ja}^S I_{ij}^{SOS} \\ & - \frac{1}{2} \sum_{jb} t_{iajb} I_{jb}^{(1)T,SOS} + \frac{1}{2} \sum_{jb} (ib \mid ja)\, I_{jb}^{(2)T,SOS} \\ & + c_{osc} \left\{ \sum_{kcl} (ik \mid lc)\, u_{kalc}^{T,SOS}(\omega) - \sum_{ckd} u_{ickd}^{T,SOS}(\omega)\, (kd \mid ac) \right\} \end{aligned} \tag{5.33}$$

$$u_{iajb}^{S,SOS}(\omega) = \frac{c_{osc}}{\omega - \epsilon_a - \epsilon_b + \epsilon_i + \epsilon_j} \left\{ \sum_k u_{ka} \left(ki \mid bj\right) + \sum_k u_{kb} \left(kj \mid ai\right) - \sum_c u_i c \left(ac \mid bj\right) - \sum_c u_{jc} \left(bc \mid ai\right) \right\} \tag{5.34}$$

$$u_{iajb}^{T,SOS}(\omega) = \frac{c_{osc}}{\omega - \epsilon_a - \epsilon_b + \epsilon_i + \epsilon_j} \left\{ \sum_k u_{ka} \left(ki \mid bj\right) - \sum_k u_{kb} \left(kj \mid ai\right) - \sum_c u_i c \left(ac \mid bj\right) + \sum_c u_{jc} \left(bc \mid ai\right) \right\} \tag{5.35}$$

with the the intermediates

$$I_{ab}^{SOS} = \frac{c_o s}{2} \left[\sum_{kcl} t_{kalc} \left(kb \mid lc\right) \right]_{a \leftrightarrow b} \tag{5.36}$$

$$I_{ij}^{SOS} = \frac{c_o s}{2} \left[\sum_{ckd} t_{ickd} \left(jc \mid kd\right) \right]_{i \leftrightarrow j} \tag{5.37}$$

$$I_{ia}^{(1)S,SOS} = \sum_{jb} \left(2 \left(ia \mid jb\right) - \left(ib \mid ja\right)\right) u_{jb}^{S} \tag{5.38}$$

$$I_{ia}^{(2)S,SOS} = c_{os} \sum_{jb} t_{iajb} u_{jb}^{S} \tag{5.39}$$

$$I_{ia}^{(1)T,SOS} = -\sum_{jb} \left(ib \mid ja\right) u_{jb}^{T} \tag{5.40}$$

$$I_{ia}^{(2)T,SOS} = -c_{os} \sum_{jb} t_{iajb} u_{jb}^{T} \tag{5.41}$$

5.3 Working Equations For Restricted AO-SOS-ADC(2) with Doubles-Folding

The goal of an atomic orbital based formulation of ADC(2) is to compute the matrix-vector product in an intermediate AO basis and transform it back to the MO basis (or alternatively an LMO basis) for the Davidson procedure, similarly to how it is done for CIS:

$$r_{ia} = C_{\mu i} r_{\underline{\mu}\overline{\nu}} C_{\nu a} \tag{5.42}$$

Furthermore, it is convenient to split the MVP into six components which are evaluated individually

$$r_{ia}(\omega) = r_{ia}^{CIS} + r_{ia}^{2A} + r_{ia}^{2B} + r_{ia}^{2C} + r_{ia}^{2D} + r_{ia}^{2E}(\omega) \tag{5.43}$$

In the next sections, using Equations ... and ... as starting points, the working equations for restricted AO-SOS-ADC(2) will be derived and discussed in detail.

5.3.1 First Order

The first order part of the MVP is identical in both ADC(2) and SOS-ADC(2)

$$r_{ia}^{S,CIS} = (\epsilon_a - \epsilon_i)u_{ia}^S + \sum_{jb} \left[2\,(ia \mid jb) - (ij \mid ab)\right] u_{jb}^S \tag{5.44}$$

$$r_{ia}^{T,CIS} = (\epsilon_a - \epsilon_i)u_{ia}^T - \sum_{jb} (ij \mid ab)\, u_{jb}^T \tag{5.45}$$

An AO formulation is obtained in an indentical manner to AO-CIS by factoring out the coefficient matrices to obtain Hartree-Fock-like expressions:

$$\begin{aligned} r_{ia}^{S,CIS,AO} &= (\epsilon_a - \epsilon_i)u_{ia}^S + \sum_{ia} C_{\mu i}C_{\sigma a}\left[\left(2\,(\mu\sigma \mid \nu\lambda) - (\mu\nu \mid \sigma\lambda)\right) u_{\underline{\nu\lambda}}^S\right] \\ &= (\epsilon_a - \epsilon_i)u_{ia}^S + \sum_{ia} C_{\mu i}C_{\sigma a}\left[2\tilde{J}_{\mu\sigma}^S - \tilde{K}_{\mu\sigma}^S\right] \end{aligned} \tag{5.46}$$

$$\begin{aligned} r_{ia}^{T,CIS,AO} &= (\epsilon_a - \epsilon_i)u_{ia}^T - \sum_{ia} C_{\mu i}C_{\sigma a}\left[(\mu\nu \mid \sigma\lambda)\, u_{\underline{\nu\lambda}}^T\right] \\ &= (\epsilon_a - \epsilon_i)u_{ia}^T - \sum_{ia} C_{\mu i}C_{\sigma a}\tilde{K}_{\mu\sigma}^T \end{aligned} \tag{5.47}$$

where $\tilde{\mathbf{J}}$ and $\tilde{\mathbf{K}}$ are the coulomb and exchange kernels, and $u_{\underline{\mu\sigma}}$ is the transition density in the AO basis

$$u_{\underline{\mu\sigma}} = C_{\mu i}u_{ia}C_{\sigma a} \tag{5.48}$$

The zero order terms (i.e. the molecular orbital energy differences) do not need to be formulated in an AO basis, because the computation time is negligeable. Similarly, transforming $\tilde{\mathbf{J}}$ and $\tilde{\mathbf{K}}$ to the MO basis formerly scales as $\mathcal{O}(N^3)$ but has very low overhead and does not influence the overall scaling of AO-SOS-ADC(2). The time-determining steps are the computataion of the J-kernel, which scales as $ccpx2$ (or $\mathcal{O}(N)$ if the continuous fast multipole method is used) and the K-kernel, which scales as $\mathcal{O}(N)$ in limit of large systems. For triplet excitations, the scaling is reduced to linear due to the absence of coulomb contributions.

5.3.2 Second Order: Part 2A and 2B

The expressions for component 2A and 2B read

$$r_{ia}^{S,SOS,2A} = \sum_b I_{ab}^{SOS}u_{ib}^S + \sum_j I_{ij}^{SOS}u_{ja}^S \tag{5.49}$$

$$r_{ia}^{T,SOS,2A} = \sum_b I_{ab}^{SOS}u_{ia}^T + \sum_j I_{ij}^{SOS}u_{ja}^T \tag{5.50}$$

with the intermediates as defined in Equation .... and .... Rather than casting the whole expression into the AO basis, it is more convenient to evaluate only the non-symmetrized intermediates $I_a^{SOS,ns}b$ and $I_{ij}^{SOS,ns}$ in the AO basis. The expressions for the intermediates

involve the t-amplitudes, and to obtain an orbital-invariant formulation, it is necessary to use the Laplace transform

$$\frac{1}{\epsilon_a - \epsilon_i + \epsilon_b - \epsilon_j} = \sum_{\alpha}^{nlap} |w^{(\alpha)}| e^{-\epsilon_a t^{(\alpha)}} e^{\epsilon_i t^{(\alpha)}} e^{-\epsilon_b t^{(\alpha)}} e^{\epsilon_j t^{(\alpha)}} \tag{5.51}$$

Using a similar strategy to AO-MP2 to factor out the coefficient matrices, the intermediates can be formulated as

$$I_{ab}^{AO-SOS,ns} = \frac{c_o s}{2} \sum_{kcl} \sum_{\alpha} |w^{(\alpha)}| e^{\Delta_{kalc} t^{(\alpha)}} (ka \mid lc) (kb \mid lc) \tag{5.52}$$

$$= \frac{c_o s}{2} \sum_{b} C_{\lambda b} \sum_{\alpha} \mid w^{(\alpha)} \mid^{1/4} e^{-\epsilon_a t^{(\alpha)}} C_{\sigma a} \sum_{\kappa\gamma\tau} \left(\underline{\kappa}\sigma \mid \underline{\tau}\overline{\gamma}\right)^{(\alpha)} (\kappa\lambda \mid \tau\gamma) \tag{5.53}$$

$$= \frac{c_o s}{2} \sum_{b} C_{\lambda b} \sum_{\alpha} \mid w^{(\alpha)} \mid^{1/4} e^{-\epsilon_a t^{(\alpha)}} C_{\sigma a} A_{\sigma\lambda}^{(\alpha)} \tag{5.54}$$

with the pseudo-AO electron integrals and the occupied/virtual pseudo density matrices

$$\left(\underline{\kappa}\sigma \mid \underline{\tau}\overline{\gamma}\right)^{(\alpha)} = P_{\kappa\kappa'}^{(\alpha)} (\kappa'\sigma \mid \tau'\gamma') P_{\tau\tau'}^{(\alpha)} Q_{\gamma\gamma'}^{(\alpha)} \tag{5.55}$$

$$P_{\mu\mu'}^{(\alpha)} = \sum_{i} C_{\mu i} e^{0.25 ln \left|w^{(\alpha)}\right| + \epsilon_i t^{(\alpha)}} C_{\mu' i} \tag{5.56}$$

$$Q_{\nu\nu'}^{(\alpha)} = \sum_{a} C_{\nu a} e^{0.25 ln \left|w^{(\alpha)}\right| - \epsilon_a t^{(\alpha)}} C_{\nu' a} \tag{5.57}$$

Similarly

$$I_{ij}^{AO-SOS,ns} = \frac{c_{os}}{2} \sum_{\alpha} \sum_{ckd} \left|w^{(\alpha)}\right| e^{\Delta_{ickd} t^{(\alpha)}} (ic \mid kd) (jc \mid kd) \tag{5.58}$$

$$= \frac{c_{os}}{2} \sum_{j} C_{\nu j} \sum_{i} \left|w^{(\alpha)}\right|^{1/4} C_{\mu i} e^{\epsilon_i t^{(\alpha)}} \sum_{\gamma\kappa\delta} \left(\mu\overline{\gamma} \mid \underline{\kappa}\overline{\delta}\right) (\nu\gamma \mid \kappa\delta) \tag{5.59}$$

$$= \frac{c_{os}}{2} \sum_{j} C_{\nu j} \sum_{i} \left|w^{(\alpha)}\right|^{1/4} C_{\mu i} e^{\epsilon_i t^{(\alpha)}} B_{\mu\nu}^{(\alpha)} \tag{5.60}$$

Finally, the intermediates are symmetrized

$$I_{ab}^{AO-SOS} = I_{ab}^{AO-SOS,ns} + I_{ba}^{AO-SOS,ns} \tag{5.61}$$

$$I_{ij}^{AO-SOS} = I_{ij}^{AO-SOS,ns} + I_{ji}^{AO-SOS,ns} \tag{5.62}$$

The time-determining step for both intermediates is the computation of the Laplace intermediates $\mathbf{A}^{(\alpha)}$ and $\mathbf{B}^{(\alpha)}$. The subsequent multiplication with the coefficient matrices is again negligeable. Consider now the sparsity diagram for the Laplace intermediate $\mathbf{A}^{(\alpha)}$:

$$\lambda \overset{\mathrm{S}}{\longleftrightarrow} \kappa \overset{\mathrm{P}}{\longleftrightarrow} \kappa' \overset{\mathrm{S}}{\longleftrightarrow} \sigma \qquad \tau \overset{\mathrm{P}}{\longleftrightarrow} \tau' \overset{\mathrm{S}}{\longleftrightarrow} \gamma' \overset{\mathrm{P}}{\longleftrightarrow} \gamma$$

The diagram has two edges, and hence $\mathbf{A}$ is evaluated with $\mathcal{O}(N^2)$ computational complexity. The same can be shown for $\mathbf{B}$. The total memory footprint is also quadratic in N.

5.3.3 Second Order: Part 2C

Component 2C is computed as

$$r_{ia}^{S,SOS,2C} = -\frac{c_{os}}{2}\sum_{jb} t_{iajb} I_{jb}^{(1)S,SOS} \tag{5.63}$$

$$r_{ia}^{T,SOS,2C} = -\frac{c_{os}}{2}\sum_{jb} t_{iajb} I_{jb}^{(1)T,SOS} \tag{5.64}$$

Applying the Laplace transform, this then gives

$$\begin{aligned}
r_{ia}^{S,AO-SOS,2C} &= -\frac{c_{os}}{2}\sum_{jb}\sum_{alpha}\left|w^{(\alpha)}\right| e^{\Delta_{iajb}t^{(\alpha)}}\,(ia\mid jb)\left[\sum_{kc}\left(2\,(jb\mid kc)-(jc\mid kb)\right)u_{kc}^{S}\right] \\
&= -\frac{c_{os}}{2}\sum_{alpha}\sum_{ia}\left|w^{(\alpha)}\right|^{1/2} C_{\mu i}e^{\epsilon_i t^{(\alpha)}} C_{\sigma a}e^{-\epsilon_a t^{(\alpha)}}\left\{\sum_{\nu\lambda}\left(\mu\alpha\mid\underline{\nu\lambda}\right)^{(\alpha)}\left[\sum_{\kappa\gamma}\left(2\,(\nu\lambda\mid\kappa\gamma)-(\nu\gamma\mid\kappa\lambda\right.\right.\right. \\
&= -\frac{c_{os}}{2}\sum_{alpha}\sum_{ia}\left|w^{(\alpha)}\right|^{1/2} C_{\mu i}e^{\epsilon_i t^{(\alpha)}} C_{\sigma a}e^{-\epsilon_a t^{(\alpha)}}\left\{\sum_{\nu\lambda}\left(\mu\alpha\mid\underline{\nu\lambda}\right)^{(\alpha)}\left[2\tilde{J}_{\lambda\nu}-\tilde{K}_{\lambda\nu}\right]\right\} \\
&= -\frac{c_{os}}{2}\sum_{alpha}\sum_{ia}\left|w^{(\alpha)}\right|^{1/2} C_{\mu i}e^{\epsilon_i t^{(\alpha)}} C_{\sigma a}e^{-\epsilon_a t^{(\alpha)}} I_{\mu\sigma}^{(1)(\alpha)S,AO-SOS}
\end{aligned} \tag{5.65}$$

Similarly, triplet contributions are given by

$$\begin{aligned}
r_{ia}^{T,AO-SOS,2C} &= \frac{c_{os}}{2}\sum_{alpha}\sum_{ia}\left|w^{(\alpha)}\right|^{1/2} C_{\mu i}e^{\epsilon_i t^{(\alpha)}} C_{\sigma a}e^{-\epsilon_a t^{(\alpha)}}\left\{\sum_{\nu\lambda}\left(\mu\alpha\mid\underline{\nu\lambda}\right)^{(\alpha)}\tilde{K}_{\lambda\nu}\right\} \\
&= \frac{c_{os}}{2}\sum_{alpha}\sum_{ia}\left|w^{(\alpha)}\right|^{1/2} C_{\mu i}e^{\epsilon_i t^{(\alpha)}} C_{\sigma a}e^{-\epsilon_a t^{(\alpha)}} I_{\mu\sigma}^{(1)(\alpha)T,AO-SOS}
\end{aligned} \tag{5.66}$$

where $\tilde{\mathbf{J}}$ and $\tilde{\mathbf{K}}$ are the same matrices needed for the CIS contributions. Note that the matrices are *transposed*, i.e. the index order is $\lambda\nu$, and not $\nu\lambda$. The time-determining step is the formation of the Laplace AO intermediates $I_{\mu\nu}^{\alpha(1)}$. Their sparsity diagrams read

$$\mu \overset{\mathrm{S}}{\longleftrightarrow} \sigma \qquad \underbrace{\nu \overset{\mathrm{P}}{\longleftrightarrow} \nu' \overset{\mathrm{S}}{\longleftrightarrow} \lambda' \overset{\mathrm{P}}{\longleftrightarrow} \lambda}_{\mathrm{J/K}}$$

The singlet and triplet AO intermediates are therefore evaluated in $\mathcal{O}(N^2)$ time.

5.3.4 Second Order: Part 2D

Now consider part 2D

$$\begin{aligned}
r_{ia}^{S,SOS,2D} &= -\frac{1}{2}\sum_{jb}\left[2\,(ia\mid jb)\,(ib\mid ja)\right] I_{jb}^{(2)S,SOS} \\
&= -\frac{1}{2}\sum_{jb} K_{iajb} I_{jb}^{(2)S,SOS}
\end{aligned} \tag{5.67}$$

$$r_{ia}^{T,SOS,2D} = \frac{1}{2}\sum_{jb}(ib \mid ja)\, I_{jb}^{(2)T,SOS} \tag{5.68}$$

Applying the Laplace transform gives the singlet expression

$$\begin{aligned} r_{ia}^{S,AO-SOS,2D} &= -\frac{c_{os}}{2}\sum_{jb} K_{iajb} \sum_{kc}\sum_{\alpha} \left|w^{(\alpha)}\right| e^{\Delta_{iajb}tpa}\,(jb \mid kc)\, u_{kc}^{S} \\ &= -\frac{c_{os}}{2}\sum_{ia} C_{\mu i}C_{\sigma a}\left[K_{\mu\sigma\nu\lambda}\left(\sum_{\alpha}\left(\underline{\nu}\overline{\lambda} \mid \kappa\gamma\right)^{(\alpha)} u_{\underline{\kappa}\overline{\gamma}}^{(\alpha)S}\right)\right] \\ &= -\frac{c_{os}}{2}\sum_{ia} C_{\mu i}C_{\sigma a} I_{\mu\sigma}^{(2)S,AO-SOS} \end{aligned} \tag{5.69}$$

Similarly, the triplet expressions

$$\begin{aligned} r_{ia}^{T,AO-SOS,2D} &= \frac{c_{os}}{2}\sum_{jb}(ia \mid jb) \sum_{kc}\sum_{\alpha} \left|w^{(\alpha)}\right| e^{\Delta_{iajb}tpa}\,(jb \mid kc)\, u_{kc}^{T} \\ &= \frac{c_{os}}{2}\sum_{ia} C_{\mu i}C_{\sigma a}\left[(\mu\sigma \mid \nu\lambda)\left(\sum_{\alpha}\left(\underline{\nu}\overline{\lambda} \mid \kappa\gamma\right)^{(\alpha)} u_{\underline{\kappa}\overline{\gamma}}^{(\alpha)T}\right)\right] \\ &= \frac{c_{os}}{2}\sum_{ia} C_{\mu i}C_{\sigma a} I_{\mu\sigma}^{(2)T,AO-SOS} \end{aligned} \tag{5.70}$$

With the transition density in the pseudo atomic orbital basis

$$u_{\underline{\mu}\overline{\sigma}}^{(\alpha)} = \left|w^{(\alpha)}\right|^{1/2} C_{\mu i} e^{\epsilon_i t^{(\alpha)}} u_{ia} C_{\sigma a} e^{-\epsilon_a t^{(\alpha)}} \tag{5.71}$$

The computation of the AO intermediates $I^{(2)SOS-AO}$ is the time-determining step, and is best evaluated as

$$\tilde{J}_{\mu\sigma}^{(\alpha)} = (\mu\sigma \mid \nu\lambda)\, u_{\underline{\nu}\overline{\lambda}}^{(\alpha)} \tag{5.72}$$

$$\tilde{J}_{\underline{\mu}\overline{\sigma}}^{(\alpha)} = P_{\mu\mu'}^{(\alpha)} \tilde{J}_{\mu\nu}^{(\alpha)} Q_{\nu\nu'}^{(\alpha)} \tag{5.73}$$

$$I_{\mu\sigma}^{(2)SOS-AO} = \sum_{\alpha}\left[2\,(\mu\sigma \mid \nu\lambda) - (\mu\lambda \mid \nu\sigma)\right] \tilde{J}_{\underline{\nu}\overline{\lambda}}^{(\alpha)} \tag{5.74}$$

Every individual step can be computed with $\mathcal{O}(N^2)$ complexity, meaning the intermediate is also evaluated with overall quadratic effort.

5.3.5 Second Order: Part 2E

The final part is given by

$$r_{ia}^{S,SOS,2E}(\omega) = c_{osc}\left\{\sum_{kcl}(ik \mid lc)\, u_{kalc}^{S,SOS}(\omega) - \sum_{ckd} u_{ickd}^{S,SOS}(\omega)\,(kd \mid ac)\right\} \tag{5.75}$$

$$r_{ia}^{T,SOS,2E}(\omega) = c_{osc}\left\{\sum_{kcl}(ik \mid lc)\, u_{kalc}^{T,SOS}(\omega) - \sum_{ckd} u_{ickd}^{T,SOS}(\omega)\,(kd \mid ac)\right\} \tag{5.76}$$

With the doubles intermediates as given in Equation ... and .... The Laplace transform needs to be applied to the energy denominator present in these intermediates. The optimal Laplace parameters are however different from the ones used for the t-amplitudes, due to the additional factor of the excitation energy ω. For each different excitation energy ω, a new Laplace quadrature needs to be computed, alongside a new set of pseudo-density matrices $\mathbf{P}$ and $\mathbf{Q}$. The additional time is however negligebale for the standard number of qudrature points ($n_{lap} < 10$). The symbol θ is used to designate the Laplace quadrature for the doubles denominator to differentiate them from the ones for the t-amplitudes.

First, an AO formulation of the doubles amplitudes will be derived such that

$$u_{iajb}(\omega) = C_{\mu i} C_{\sigma a} u_{\mu\sigma\nu\lambda} C_{\nu j} C_{\lambda b} \tag{5.77}$$

For qunatities like the MO integrals $(ia \mid jb)$, this is straight-forwardly done by factoring out the coefficient matrices. However, the situation is more complex in the doubles intermediates, due to the presence of terms like $u_{ka}\,(ki \mid bj)$. For the MO transition densties, the non-orthogonality of the AO basis needs to be taken into consideration. The MO coefficient matrices are factored out by a PAO backtransform (cf. Eqaution ...)

$$u_{ia} = C_{\mu i} S_{\mu\mu'} u_{\underline{\mu}'\overline{\sigma}'} S_{\sigma'\sigma} C_{\sigma a} \tag{5.78}$$

The doubles intermediates can then be expressed as (shown for singlet only)

$$\begin{aligned} u^{S/T}_{iajb}(\omega) = -c_{osc} \sum_{\theta} \sum_{\mu\sigma\nu\lambda} |w^{(\theta)}| e^{(\omega - \epsilon_a - \epsilon_b + \epsilon_i + \epsilon_j) t^{(\theta)}} C_{\mu i} C_{\sigma a} C_{\nu j} C_{\lambda b} \Bigg\{ \\ \sum_{\kappa} \left[u^{S/T}_{\underline{\kappa}\overline{\sigma}'} S_{\sigma'\sigma} \left(\kappa\mu \mid \nu\lambda\right) \pm u^{S/T}_{\underline{\kappa}\overline{\lambda}'} S_{\lambda'\lambda} \left(\nu\kappa \mid \mu\sigma\right) \right] \\ - \sum_{\gamma} \left[S_{\mu\mu'} u^{S/T}_{\underline{\mu}'\overline{\sigma}} \left(\nu\lambda \mid \sigma\gamma\right) \pm S_{\nu\nu'} u^{S/T}_{\underline{\nu}'\overline{\gamma}} \left(\mu\sigma \mid \gamma\lambda\right) \right] \Bigg\} \\ = - \sum_{\theta} \sum_{\mu\sigma\nu\lambda} |w^{(\theta)}| e^{(\omega - \epsilon_a - \epsilon_b + \epsilon_i + \epsilon_j) t^{(\theta)}} C_{\mu i} C_{\sigma a} u^{S/T}_{\mu\sigma\nu\lambda} C_{\nu j} C_{\lambda b} \end{aligned} \tag{5.79}$$

Note the additional minus sign in front of the Laplace summation. After the Laplace transform, the sign of the denominator is swapped, i.e. $\frac{1}{\pm x} \rightarrow exp(\mp x t^{(\theta)})$. For large negative occupied molecular orbital energies ϵ_i or large postive virtual molecular orbital energies ϵ_a, this would lead to very large values and numerical instabilities. For this reason, the minus sign is factored out to reverse the sign in the exponent.

Inserting the above expression into Eqaution ... gives the expression for part 2E

constructed via AO intermediates:

$$\begin{aligned} r_{ia}^{S/T,AO-SOS,2E}(\omega) &= -c_{osc}^2 \sum_{\theta} e^{\omega t^{(\theta)}} \left\{ C_{\mu i} |w^{(\theta)}|^{1/4} C_{\sigma a} e^{-\epsilon_a t^{(\theta)}} \left[\sum_{\kappa\gamma\tau} \left(\mu\underline{\kappa} \mid \underline{\tau}\overline{\gamma}\right)^{(\theta)} u_{\kappa\sigma\tau\gamma}^{S/T} \right] \right. \\ &\left. - |w^{(\theta)}|^{1/4} C_{\mu i} e^{\epsilon_i t^{(\theta)}} C_{\sigma a} \left[\sum_{\gamma\kappa\delta} u_{\mu\gamma\kappa\delta}^{S/T} \left(\underline{\kappa}\overline{\delta} \mid \sigma\overline{\gamma}\right)^{(\theta)} \right] \right\} \\ &= -c_{osc}^2 \sum_{\theta} e^{\omega t^{(\theta)}} \left\{ C_{\mu i} |w^{(\theta)}|^{1/4} C_{\sigma a} e^{-\epsilon_a t^{(\theta)}} R_{\mu\overline{\sigma}}^{(\theta)(1)S/T} \right. \\ &\left. - |w^{(\theta)}|^{1/4} C_{\mu i} e^{\epsilon_i t^{(\theta)}} C_{\sigma a} R_{\underline{\mu}\sigma}^{(\theta)(2)S/T} \right\} \end{aligned} \tag{5.80}$$

Similarly to previous expressions, the AO electron repulsion integrals are not completely transformed into the pseudo-AO basis, but only three-quarter transformed integrals are obtained. To obtain fully-transformed integrals, it is beneficial to perform the following transformation:

$$r_{ia} = \bar{C}_{\mu i'} r_{\mu\sigma} \bar{C}_{\sigma a'} = \bar{C}_{\mu i'} C_{\mu i} r_{ia} C_{\sigma a} \bar{C}_{\sigma a'} \tag{5.81}$$

Inserting this expression into Equation ... yields

$$\begin{aligned} r_{ia}^{S/T,AO-SOS,2E}(\omega) &= -c_{osc}^2 \sum_{\theta} e^{\omega t^{(\theta)}} \left\{ \bar{C}_{\mu i} P_{\mu\nu} \bar{C}_{\sigma a} \left[\sum_{\kappa\gamma\tau} \left(\underline{\nu\kappa} \mid \underline{\tau}\overline{\gamma}\right)^{(\theta)} u_{\kappa\sigma\tau\gamma}^{S/T} \right] \right. \\ &\left. - \bar{C}_{\mu i} \bar{C}_{\sigma a} Q_{\sigma\nu} \left[\sum_{\gamma\kappa\delta} u_{\mu\gamma\kappa\delta}^{S/T} \left(\underline{\kappa}\overline{\delta} \mid \overline{\nu\gamma}\right)^{(\theta)} \right] \right\} \\ &= -c_{osc}^2 \sum_{\theta} e^{\omega t^{(\theta)}} \left\{ \bar{C}_{\mu i} \bar{C}_{\sigma a} R_{\underline{\mu}\overline{\sigma}}^{(\theta)(1)S/T} - \bar{C}_{\mu i} \bar{C}_{\sigma a} R_{\underline{\mu}\overline{\sigma}}^{(\theta)(2)S/T} \right\} \end{aligned} \tag{5.82}$$

which gives fully transformed integrals. This step is necessary to get a obtain a better factorization for part 2E in the density fitting approximation. Note that there are other cases of non-fully transformed integrals in the previous parts - however a full transformation does not give any significant advantage for a DF formulation, so they are left unchanged.

The time-determining step is the formation of the **R** intermediates, which in turn depend on the AO doubles intermediates. Consider the sparsity diagram for the following term encountered in Equation ... :

$$u_{\underline{\kappa}\overline{\sigma}'} S_{\sigma'\sigma} \left(\kappa\mu \mid \nu\lambda\right) \tag{5.83}$$

$$\nu \overset{\mathrm{S}}{\longleftrightarrow} \lambda \qquad \mu \overset{\mathrm{S}}{\longleftrightarrow} \kappa \overset{\mathrm{P}}{\longleftrightarrow} \sigma' \overset{\mathrm{S}}{\longleftrightarrow} \sigma$$

Similar diagrams can be derived for the other three contractions in Equation .... This shows an overall quadratic scaling in computational effort and number of non-zero elements for the AO doubles intermediates $u_{\mu\sigma\nu\lambda}$. The indices μ/σ and ν/λ are connected by either

an S/P junction, but no sparsity relationship can be established between those pairs, similar to the AO electron integrals.

This information can be used to find the scaling of the $\mathbf{R}$ intermediates. For example, the sparsity digram for $\mathbf{R}^{(1)}_{\underline{\mu}\overline{\sigma}} = \left(\underline{\mu}\underline{\kappa} \mid \underline{\tau}\overline{\gamma}\right)^{(\theta)} u_{\kappa\sigma\tau\gamma}$ reads

$$\mu \overset{\mathrm{P}}{\longleftrightarrow} \mu' \overset{\mathrm{S}}{\longleftrightarrow} \kappa' \overset{\mathrm{P}}{\longleftrightarrow} \kappa \overset{\mathrm{S/P}}{\longleftrightarrow} \sigma \qquad \tau \overset{\mathrm{P}}{\longleftrightarrow} \tau' \overset{\mathrm{S}}{\longleftrightarrow} \gamma' \overset{\mathrm{P}}{\longleftrightarrow} \gamma$$

and a similar diagram can be drawn for $\mathbf{R}^{(2)}$, which again shows quadratic scaling.

5.3.6 Potential Linear Scaling Formulation

5.3.7 Summary

The SOS-ADC(2) matrix-vector product is finally computed as

$$\begin{aligned}
r_{ia}^{S,AO-DF-SOS}(\omega) &= (\epsilon_a - \epsilon_i) u_{ia}^S + \sum_{\mu\nu} C_{\mu i} C_{\nu a} \left(2\tilde{J}^S_{\mu\nu} - \tilde{K}^S_{\mu\nu}\right) \\
&\quad + \sum_b I^{AO-DF-SOS}_{ab} u^S_{ib} + \sum_j I^{AO-DF-SOS}_{ij} u^S_{ja} \\
&\quad - \frac{c_{os}}{2} \sum_\alpha \sum_{ia} \left|w^{(\alpha)}\right|^{1/2} C^{(\alpha)}_{\mu i} C^{(\alpha)}_{\sigma a} I^{(1)(\alpha)S,AO-DF-SOS}_{\mu\sigma} \\
&\quad - \frac{c_{os}}{2} \sum_{ia} C_{\mu i} C_{\sigma a} I^{(2)S,AO-DF-SOS}_{\mu\sigma} \\
&\quad - c^2_{osc} \sum_\theta e^{\omega t^{(\theta)}} \left\{ \bar{C}_{\mu i} \bar{C}_{\sigma a} R^{(\theta)(1)S}_{\underline{\mu}\overline{\sigma}} - \bar{C}_{\mu i} \bar{C}_{\sigma a} R^{(\theta)(2)S}_{\underline{\mu}\overline{\sigma}} \right\}
\end{aligned} \tag{5.84}$$

$$\begin{aligned}
r_{ia}^{T,AO-DF-SOS}(\omega) &= (\epsilon_a - \epsilon_i) u_{ia}^T + \sum_{\mu\nu} C_{\mu i} C_{\nu a} \tilde{K}^T_{\mu\nu} \\
&\quad + \sum_b I^{AO-DF-SOS}_{ab} u^T_{ib} + \sum_j I^{AO-DF-SOS}_{ij} u^T_{ja} \\
&\quad - \frac{c_{os}}{2} \sum_\alpha \sum_{ia} \left|w^{(\alpha)}\right|^{1/2} C^{(\alpha)}_{\mu i} C^{(\alpha)}_{\sigma a} I^{(1)(\alpha)T,AO-DF-SOS}_{\mu\sigma} \\
&\quad - \frac{c_{os}}{2} \sum_{ia} C_{\mu i} C_{\sigma a} I^{(2)T,AO-DF-SOS}_{\mu\sigma} \\
&\quad - c^2_{osc} \sum_\theta e^{\omega t^{(\theta)}} \left\{ \bar{C}_{\mu i} \bar{C}_{\sigma a} R^{(\theta)(1)T}_{\underline{\mu}\overline{\sigma}} - \bar{C}_{\mu i} \bar{C}_{\sigma a} R^{(\theta)(2)T}_{\underline{\mu}\overline{\sigma}} \right\}
\end{aligned} \tag{5.85}$$

where the intermediates are evaluated as presented in the previous sections. The 2-index intermediates still need to be transformed back to the canonical MO basis for the Davidson procedure, but the computational effort required is negligeable. The overall cost of the AO-SOS-ADC(2) scales quadratically both in time and memory requirements.

It is expected that AO-SOS-ADC(2) has the same drawback as LinK or AO-MP2, namely a late onset of the low-scaling regime for larger basis sets. This additional overhead is even worse for AO-ADC due to the complexity of the formulas which involve many more tensor contractions than the ground state, and more S and P junctions which

makes the method much more dependent on basis set size. This is further aggravated by the fact that diffuse basis functions are essential to obtain accurate excitation energies as opposed to ground state correlation energies.

5.4 Working Equations for Restricted AO-DF-SOS-ADC(2) with Doubles-Folding

To lower the steep scaling associated with increasing basis set size, the density fitting approximation is introduced. The two-electron repulsion integrals are approximated using the generalized form

$$(\mu\sigma \mid \nu\lambda) = B_{\mu\sigma X} M_{XY} B_{Y\nu\lambda} \tag{5.86}$$

where the quantities **B** and **M** depend on the density fitting method. Furthermore, the J-,K- and Z-kernels are introduced:

$$\mathcal{J}\{M,P\}_{\mu\nu} = B_{\mu\nu X} M_{XY} B_{Y\sigma\lambda} P_{\lambda\sigma} \tag{5.87}$$

$$\mathcal{K}\{M,P\}_{\mu\nu} = B_{\mu\sigma X} M_{XY} B_{Y\nu\lambda} P_{\lambda\sigma} \tag{5.88}$$

$$\mathcal{Z}\{P,Q\}_{XY} = B_{X\mu\nu} P_{\mu\mu'} B_{\mu'\nu'Y} Q_{\nu'\nu} \tag{5.89}$$

This notation allows to reduce the complexity of the formulas to some degree.

The working equations for AO-DF-SOS-ADC(2) are given in Tables 5.1, 5.2 and 5.3. Table 5.1 shows the pre-iteration steps, that is, the computation of the Laplace parameters and the intermediates I_{ij} and I_{ab}. The construction of the intermediates can be formulated in terms of the Z- and K-kernels. In the K-kernel, the metric matrix **M** is replaced by the Laplace matrix **G** formed by the Z-kernel. The contraction of **G** with $B_{X\mu\nu}$ in the K-kernel is the most expensive step in the pre-iteration procedure. The resulting tensor can be computed on-the-fly as it is not needed for any other contraction.

Table 5.2 shows the steps to form part 1 and 2A-2D of the matrix-vector product. The steps are listed for both singlet and triplet. First the CIS Fock-like matrix $\mathbf{F}^{CIS}$ is formed (1a-b), as it represents an important intermediate for the subsequent steps. This is easily done using the J/K-kernels. Zero and first order (2a+b) contributions, as well as part 2A and 2B (3a+b) of the second order contributions are formed trivially afterwards using the intermediates. Part 2C is most conveniently computed using the J-kernel where the density matrix is replaced by a pseudo-density matrix **H** (4b) formed by contracting with a pseudo-AO CIS matrix (4a). Part 2D is a bit more involved. First, the matrix **T** is formed by looping over the laplace points and using the J-kernel (5a+b). Then, this matrix is used in a fock-like construction scheme (5c) to get the final result.

Finally, Table 5.3 shows the construction of part 2E of the MVP. This is the most expensive step of the whole procedure. First, the Laplace parameters are (re-)computed for the current excitation energy ω. The algorithm then enters the Laplace loop in step 2. First, the intermediate tensors $B_{X\underline{\mu}\bar{\nu}}$ (2b) and $R_{X\underline{\mu}\underline{\nu}}$ (2c-2e) are formed and stored. They are then used in step 2f and 2g to form two intermediate matrices in the auxiliary basis. The final intermediate $D_{X\underline{\mu}\bar{\nu}}$ is formed in step (2h). After two major contraction steps (2i+j), the contributions are added to part 2E in (2k).

Using the density fitting approximation, the four-index intermediates $u_{\mu\sigma\nu\lambda}$ from AO-SOS-ADC(2) can be avoided, and the procedure only depends on 3-index intermediates.

Step-by-step

1. Compute Laplace quadrature parameters $\{w^{(\alpha)}, t^{(\alpha)}\}$ for $\frac{1}{\epsilon_i+\epsilon_j-\epsilon_a-\epsilon_b}$
2. If needed by the J,K or Z kernels, compute the cholesky decompositions of the occupied and virtual pseudo-density matrix
3. Compute the intermediate matrices I_{ij} and I_{ab}

   for $\alpha = 0$ **to** n_{lap} **do**

   (a) $G^{(\alpha)}_{XY} \leftarrow M_{XR}\mathcal{Z}\left\{P^{(\alpha)}, Q^{(\alpha)}\right\}_{RS} M_{SR}$

   (b) $I_{ab} \leftarrow -\frac{c_{os}}{2} C_{\nu b} \sum_\alpha C_{\mu a} \mid w^{(\alpha)} \mid^{1/4} e^{-\epsilon_a t^{(\alpha)}} \mathcal{K}\left\{P^{(\alpha)}, G^{(\alpha)}\right\}_{\mu\nu}$

   (c) $I_{ij} \leftarrow -\frac{c_{os}}{2} C_{\nu j} \sum_\alpha C_{\mu i} \mid w^{(\alpha)} \mid^{1/4} e^{\epsilon_i t^{(\alpha)}} \mathcal{K}\left\{Q^{(\alpha)}, G^{(\alpha)}\right\}_{\mu\nu}$

   end

   (d) Symmetrize: $I_{ij} \leftarrow I_{ji}$

   (e) Symmetrize: $I_{ab} \leftarrow I_{ba}$

Table 5.1: Intermediates

Step-by-step (2)

1. Compute the CIS Fock-like matrix
   (a) if singlet: $U_{\mu\nu} \leftarrow C_{\mu i} u^S_{ia} C_{\nu a}$
   if triplet: $U_{\mu\nu} \leftarrow C_{\mu i} u^T_{ia} C_{\nu a}$
   (b) if singlet: $F^{CIS}_{\mu\nu} \leftarrow 2 * \mathcal{J}\left\{U, M\right\}_{\nu\mu} - \mathcal{K}\left\{U, M\right\}_{\nu\mu}$
   if triplet: $F^{CIS}_{\mu\nu} \leftarrow -\mathcal{K}\left\{U, M\right\}_{\nu\mu}$
2. Add zero- and first-order terms
   (a) $r_{ia} \leftarrow (\epsilon_a - \epsilon_i) u_{ia}$
   (b) $r_{ia} \leftarrow C_{\mu i} F^{CIS}_{\mu\nu} C_{\nu a}$
3. Compute part (A) and (B) of second-order term
   (a) $r_{ia} \leftarrow u_{ib} I_{ab}$
   (b) $r_{ia} \leftarrow u_{ja} I_{ij}$
4. Compute part (C) of seond-order term

   for $\alpha = 0$ **to** n_{lap} **do**
   (a) if singlet: $H^{(\alpha)}_{\mu\nu} \leftarrow P^{(\alpha)}_{\mu'\mu} Q^{(\alpha)}_{\nu'\nu} F^{CIS}_{\nu'\mu'}$
   if triplet: $H^{(\alpha)}_{\mu\nu} \leftarrow -P^{(\alpha)}_{\mu'\mu} Q^{(\alpha)}_{\nu'\nu} F^{CIS}_{\nu'\mu'}$
   (b) $r_{ia} \leftarrow \frac{-c_{os}}{4} \mid w^{(\alpha)} \mid^{1/2} C_{\mu i} e^{\epsilon_i t^{(\alpha)}} C_{\nu a} e^{-\epsilon_a t^{(\alpha)}} \mathcal{J}\left\{H^{(\alpha)}, M\right\}_{\mu\nu}$
   end
5. Compute part (D) of second-order term

   for $\alpha = 0$ **to** n_{lap} **do**
   (a) if singlet: $U^{(\alpha)}_{\mu\nu} \leftarrow \mid w^{(\alpha)} \mid^{1/2} C_{\mu i} e^{\epsilon_i t^{(\alpha)}} C_{\nu a} e^{-\epsilon_a t^{(\alpha)}} u^S_{ia}$
   if triplet: $U^{(\alpha)}_{\mu\nu} \leftarrow - \mid w^{(\alpha)} \mid^{1/2} C_{\mu i} e^{\epsilon_i t^{(\alpha)}} C_{\nu a} e^{-\epsilon_a t^{(\alpha)}} u^T_{ia}$
   (b) $T_{\mu\nu} \leftarrow \frac{1}{2} \sum_\alpha P^{(\alpha)}_{\mu\mu'} Q^{(\alpha)}_{\nu\nu'} \mathcal{J}\left\{U^{(\alpha)}, M\right\}_{\nu\mu}$
   end

   (c) $r_{ia} \leftarrow -\frac{c_{os}}{2} C_{\mu i} C_{\nu a} \left[2\mathcal{J}\left\{T, M\right\}_{\mu\nu} - \mathcal{K}\left\{T, M\right\}_{\mu\nu}\right]$

Table 5.2: Intermediates

Step-by-step (3)

1. Compute Laplace quadrature parameters $\{w^{(\theta)}, t^{(\theta)}\}$ for $\frac{1}{-\omega+\epsilon_i+\epsilon_j-\epsilon_a-\epsilon_b}$
2. Compute part (E) of second-order contributions

for $\alpha = 0$ **to** n_{lap} **do**

(a) Compute doubles pseudo-matrices $P^{(\theta)}$ and $Q^{(\theta)}$

(b) $B^{(\theta)}_{X\underline{\mu}\overline{\sigma}} \leftarrow P^{(\theta)}_{\mu\mu'} B_{X\mu'\nu'} Q^{(\theta)}_{\nu'\nu}$

(c) $v^{(1)(\theta)}_{\underline{\mu}\overline{\sigma}} \leftarrow P^{(\theta)}_{\mu\lambda} S_{\lambda\nu} u_{\underline{\nu}\overline{\sigma}}$

(d) $v^{(2)(\theta)}_{\underline{\mu}\overline{\sigma}} \leftarrow u_{\underline{\mu}\overline{\gamma}} S_{\gamma\lambda} Q^{(\theta)}_{\lambda\sigma}$

(e) $R^{(\theta)}_{X\underline{\mu}\overline{\sigma}} \leftarrow P^{(\theta)}_{\mu\lambda} B_{X\lambda\nu} v^{(1)(\theta)}_{\underline{\nu}\overline{\sigma}} - v^{(2)(\theta)}_{\underline{\mu}\overline{\gamma}} B_{X\gamma\nu} Q^{(\theta)}_{\nu\sigma}$

(f) $H^{(\theta)}_{XY} \leftarrow B^{(\theta)}_{X\underline{\mu}\overline{\sigma}} B_{Y\mu\sigma}$

(g) $G^{(\theta)}_{XY} \leftarrow R^{(\theta)}_{X\underline{\mu}\overline{\sigma}} B_{Y\mu\sigma}$

(h) $D^{(\theta)}_{X\underline{\mu}\overline{\sigma}} \leftarrow H^{(\theta)}_{XY} R^{(\theta)}_{Y\underline{\mu}\overline{\sigma}} + G^{(\theta)}_{XY} B_{Y\underline{\mu}\overline{\sigma}}$

(i) $r^{(A)}_{ia}(\omega) \leftarrow \overline{C}_{\mu i} P_{\mu\mu'} \left[D^{(\theta)}_{X\underline{\nu}\overline{\sigma}} B_{X\nu\mu'} \right] \overline{C}_{\sigma a}$

(j) $r^{(B)}_{ia}(\omega) \leftarrow \overline{C}_{\mu i} \left[D^{(\theta)}_{X\underline{\mu}\overline{\gamma}} B_{X\gamma\sigma'} \right] Q_{\sigma'\sigma} \overline{C}_{\sigma a}$

(k) $r_{ia} += c^2_{os-coupling} e^{\omega t^{(\alpha)}} \left[-r^{(A)}_{ia}(\alpha,\omega) + r^{(B)}_{ia}(\alpha,\omega) \right]$

end

Table 5.3: Algorithm for singlet-excitations

Similarly to AO-MP2, the prefactor can be significantly lowered by virtue of the incomplete cholesky decomposition. Table 5.4 and 5.5 show how the intermediates in part 2E can be formed in a mixed pseudo-AO/MO basis ("OB" algorithm) or in a complete pseudo-MO basis ("OV" algorithm). While the fully transformed OV version has a lower memory footprint by reducing the dimension form $N_X N_{AO}^2$ to $N_X OV$, the overhead due to the additional cholesky decomposition of the virtual pseudo-density might outweigh the benefit: first, the virtual space may be quite large, especially if large basis sets are used, increasing the rank of the matrix and hence the computational effort of the Cholesky decomposition. Second, the decompositions need to be recomputed for each ω, and foreach Laplace point. The OB version might offer a compromise between memory savings and additional overhead.

It should be noted that the Cholesky decomposition can also be used in the Z-kernels in steps 3a of Table 5.1 to reduce its prefactor as well.

... Cholesky in K of intermediates ???

By using local density fitting, the inherent quadratic scaling of the AO-SOS-ADC(2) method also applies for AO-DF-SOS-ADC(2).

???? DAVIDSON ????

Step-by-step (3)

1. Compute Laplace quadrature parameters $\{w^{(\theta)}, t^{(\theta)}\}$ for $\frac{1}{-\omega+\epsilon_i+\epsilon_j-\epsilon_a-\epsilon_b}$
2. Compute part (E) of second-order contributions

for $\alpha = 0$ **to** n_{lap} **do**

(a) Compute doubles pseudo-matrices $P^{(\theta)}$ and $Q^{(\theta)}$ and the cholesky decomposition $L_{\mu\underline{i}}$

(b) $B^{(\theta)}_{X\underline{i}\overline{\sigma}} \leftarrow L^{(\theta)}_{\mu\underline{i}} B_{X\mu\nu} Q^{(\theta)}_{\nu\sigma}$

(c) $v^{(1)(\theta)}_{\underline{i}\overline{\sigma}} \leftarrow L^{(\theta)}_{\mu\underline{i}} S_{\mu\nu} u_{\underline{\nu}\overline{\sigma}}$

(d) $v^{(2)(\theta)}_{\underline{\mu}\overline{\sigma}} \leftarrow u_{\underline{\mu}\overline{\gamma}} S_{\gamma\lambda} Q^{(\theta)}_{\lambda\sigma}$

(e) $R^{(\theta)}_{X\underline{i}\overline{\sigma}} \leftarrow L^{(\theta)}_{\mu\underline{i}} B_{X\mu\nu} v^{(1)(\theta)}_{\underline{\nu}\overline{\sigma}} - v^{(2)(\theta)}_{\underline{i}\overline{\gamma}} B_{X\gamma\nu} Q^{(\theta)}_{\nu\sigma}$

(f) $H^{(\theta)}_{XY} \leftarrow L^{(\theta)}_{\mu\underline{i}} B^{(\theta)}_{X\underline{i}\overline{\sigma}} B_{Y\mu\sigma}$

(g) $G^{(\theta)}_{XY} \leftarrow L_{\mu\underline{i}} R^{(\theta)}_{X\underline{i}\overline{\sigma}} B_{Y\mu\sigma}$

(h) $D^{(\theta)}_{X\underline{i}\overline{\sigma}} \leftarrow H^{(\theta)}_{XY} R^{(\theta)}_{Y\underline{i}\overline{\sigma}} + G^{(\theta)}_{XY} B_{Y\underline{i}\overline{\sigma}}$

(i) $r^{(A)}_{ia}(\omega) \leftarrow \overline{C}_{\lambda i} P_{\lambda\mu} \left[D^{(\theta)}_{X\underline{k}\overline{\sigma}} B_{X\nu\mu} L^{(\theta)}_{\nu\underline{k}} \right] \overline{C}_{\sigma a}$

(j) $r^{(B)}_{ia}(\omega) \leftarrow \overline{C}_{\mu i} L_{\mu\underline{i}} \left[D^{(\theta)}_{X\underline{i}\overline{\gamma}} B_{X\sigma\gamma} \right] Q_{\gamma\lambda} \overline{C}_{\lambda a}$

(k) $r_{ia} += c^2_{os-coupling} e^{\omega t^{(\alpha)}} \left[-r^{(A)}_{ia}(\alpha,\omega) + r^{(B)}_{ia}(\alpha,\omega) \right]$

end

Table 5.4: Algorithm for singlet-excitations

Step-by-step (3)

1. Compute Laplace quadrature parameters $\{w^{(\theta)}, t^{(\theta)}\}$ for $\frac{1}{-\omega+\epsilon_i+\epsilon_j-\epsilon_a-\epsilon_b}$
2. Compute part (E) of second-order contributions

for $\alpha = 0$ **to** n_{lap} **do**

(a) Compute doubles pseudo-matrices $P^{(\theta)}$ and $Q^{(\theta)}$ and their cholesky decompositions $L^{(\theta)}_{\mu\underline{i}}$ and $L^{(\theta)}_{\sigma\overline{a}}$

(b) $B^{(\theta)}_{X\underline{i}\overline{a}} \leftarrow L^{(\theta)}_{\mu\underline{i}} B_{X\mu\nu} L^{(\theta)}_{\nu\overline{a}}$

(c) $v^{(1)(\theta)}_{\underline{i}\overline{\sigma}} \leftarrow L^{(\theta)}_{\mu\underline{i}} S_{\mu\nu} u_{\underline{\nu}\overline{\sigma}}$

(d) $v^{(2)(\theta)}_{\underline{\mu}\overline{a}} \leftarrow u_{\underline{\mu}\overline{\gamma}} S_{\gamma\lambda} L^{(\theta)}_{\lambda\overline{a}}$

(e) $R^{(\theta)}_{X\underline{i}\overline{a}} \leftarrow L^{(\theta)}_{\mu\underline{i}} B_{X\mu\nu} v^{(1)(\theta)}_{\underline{\nu}\overline{\alpha}} - v^{(2)(\theta)}_{\underline{i}\overline{\gamma}} B_{X\gamma\nu} L^{(\theta)}_{\nu\overline{a}}$

(f) $H^{(\theta)}_{XY} \leftarrow B^{(\theta)}_{X\underline{i}\overline{a}} B_{Y\underline{i}\overline{a}}$

(g) $G^{(\theta)}_{XY} \leftarrow R^{(\theta)}_{X\underline{i}\overline{a}} B_{Y\underline{i}\overline{a}}$

(h) $D^{(\theta)}_{X\underline{i}\overline{a}} \leftarrow H^{(\theta)}_{XY} R^{(\theta)}_{Y\underline{i}\overline{a}} + G^{(\theta)}_{XY} B_{Y\underline{i}\overline{a}}$

(i) $r^{(A)}_{ia}(\omega) \leftarrow \overline{C}_{\lambda i} P_{\lambda\mu} \left[D^{(\theta)}_{X\underline{k}\overline{a}} B_{X\nu\mu} L_{\nu\underline{k}} \right] L_{\sigma\overline{a}} \overline{C}_{\sigma a}$

(j) $r^{(B)}_{ia}(\omega) \leftarrow \overline{C}_{\mu i} L_{\mu\underline{i}} \left[D^{(\theta)}_{X\underline{i}\overline{b}} B_{X\gamma\sigma} L_{\sigma\overline{b}} \right] Q_{\gamma\lambda} \overline{C}_{\lambda a}$

(k) $r_{ia} += c^2_{os-coupling} e^{\omega t^{(\alpha)}} \left[-r^{(A)}_{ia}(\alpha,\omega) + r^{(B)}_{ia}(\alpha,\omega) \right]$

end

Table 5.5: Algorithm for singlet-excitations

Chapter 6

Benchmarking AO-DF-SOS-ADC(2)

The theoretical groundwork for AO-DF-SOS-ADC(2) was laid out in detail in the previous chapters. In this chapter, the performance of the method and its components will be anylyzed *in silico*. First, a brief overview is given on the computational details, as well as on the types of molecular systems that are taken into consideration. Then, the scaling, memory foot-print and accuracy of the J,K and Z kernels is analyzed for different metrics in the context of Hartree-Fock and MP2. It serves also also as a mean to get a first grip on the performance of quasi-robust density fitting. Finally, the AO-DF-SOS-ADC(2) method is anaylzed in the last part.

6.1 Computational Details

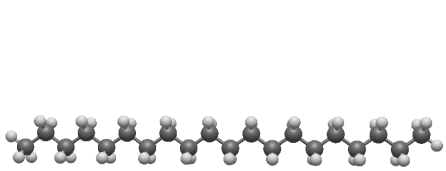
All results presented in this chapter were obtained using MEGALOchem, an open-source quantum chemistry package which specializes in algorithms expliting sparsity. For more details, the reader is referred to the user's guide and the developer's guide in chapter ... and ...

All calculations were performed on ...

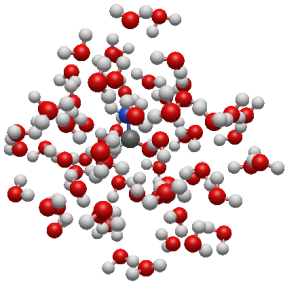
6.2 Molecular Test Systems

Virtually all works that present some form of low-scaling electronic structure methods use linear alkanes (LA) as their test systems. Linear molecular system represent a best case scenario where the overlap between basis functions decays very rapidly as function of distance. The low scaling regime can generally be reached quite quickly, which is great for getting a first impression on the performance on the methods. Unfortunately, linear systems like alkanes are chemically uninteresting. For this reason, the present benchmarking also looks at the worst case scenario of spherically shaped molecular system, in this case solvated formamide (FW) with differently sized solvation shells. For systems such as these, electron correlation descreases very slowly as function of increasing system size N. The LA and FW systems are used to analyze the performance of the kernels.

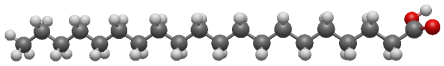
Another type of system is introduced for AO-DF-SOS-ADC(2): linear carboxlyic acids (LC). The domain for the lowest lying domain is entirely located on the functional group



(a) $y = x$



(b) $y = 5/x$



(c) $y = 3sinx$

Figure 6.1: Three simple graphs

COOH ($\pi \rightarrow \pi*$, Figure ...). This class of molecules represents the absolute best case scenario for local excitation methods: fast decay of correlation effects as a function of distance due to its linearity, and a very localized excitation domain.

This other form of locality is also the reason why the FW systems were chosen instead of simple water clusters: while correlation effects decay slowly, the excitation domain for the lowest reason is localized on the formamide molecule (ref). These systems represent an interesting mix between different types of locality and non-locality and give insight on how local excited state methods handle these fringe cases.

Four different sizes are chosen for each system type in a manner that they have a comparable number of basis functions to allow a better comparison.

Only basis sets with double-zeta quality were considered (cc-pVDZ, def2-SVP, aug-cc-pVDZ, def2-SVPD) due to time and resource constraints. It is expected that the methods still suffer from an $\mathcal{O}(N^4)$ scaling as a function of basis set size. The number of basis functions for the different

LA		FW		LC	
Molecule	N_{AO}	Molecule	N_{AO}	Molecule	N_{AO}
$H_{42}C_{20}$	490	$H_{33}CNO_{16}$	417	$H_{40}C_{20}O_2$	508
$H_{82}C_{40}$	970	$H_{63}CNO_{31}$	777	$H_{80}C_{40}O_2$	988
$H_{162}C_{80}$	1930	$H_{129}CNO_{64}$	1569	$H_{160}C_{80}O_2$	1948
$H_{322}C_{160}$	3850	$H_{291}CNO_{145}$	3513	$H_{320}C_{160}O_2$	3868

Table 6.1: Molecular formula for the considered systems and the number of basis functions when using cc-pVDZ

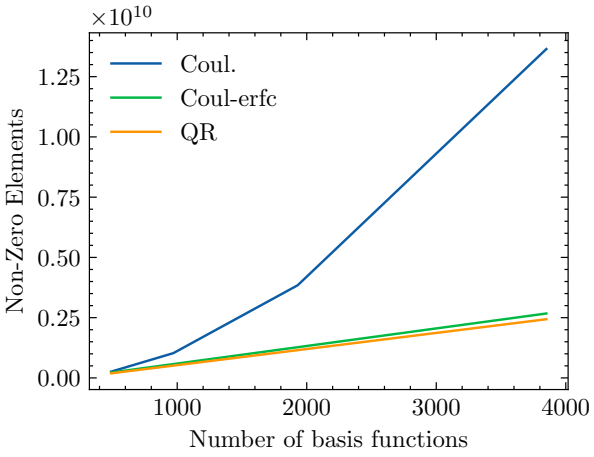


Figure 6.2: Figure

N_{AO}	Coul.	Coul-erfc	QR
490	—	—	—
970	2.01	1.37	1.45
1930	1.92	1.16	1.18
3850	1.83	1.07	1.08

Table 6.2: Table

6.3 Ground-State Prerequisites

6.3.1 Hartree-Fock

6.3.2 MP2

6.4 Excited-State

	J			K (STEP 1)			K (STEP 2)			
N_{AO}	Coul.	Erfc	QR	Coul.	Erfc	QR	Coul.	Erfc	QR	Coul. (MO)
417	—	—	—	—	—	—	—	—	—	—
777	2.18	2.20	1.88	2.77	2.77	2.51	2.55	2.55	2.33	2.33
1569	2.32	1.71	1.28	2.77	2.18	1.85	2.82	2.13	1.67	2.86
3513	2.08	—	1.94	2.79	—	2.74	2.59	—	2.65	3.13

Table 6.10: Here

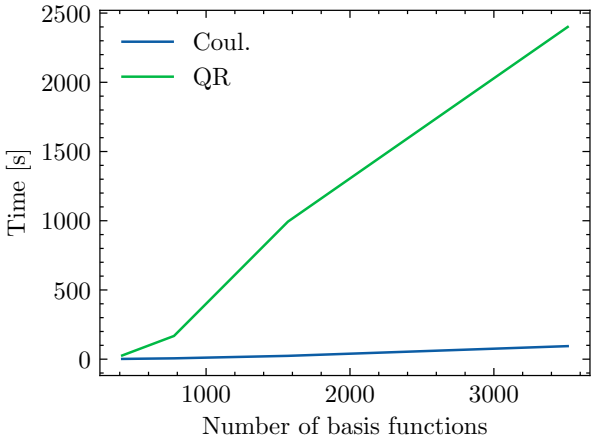


Figure 6.3: Figure

N_{AO}	Coul.	QR
490	—	—
970	1.94	2.95
1930	1.96	2.53
3850	1.70	1.09

Table 6.3: Table

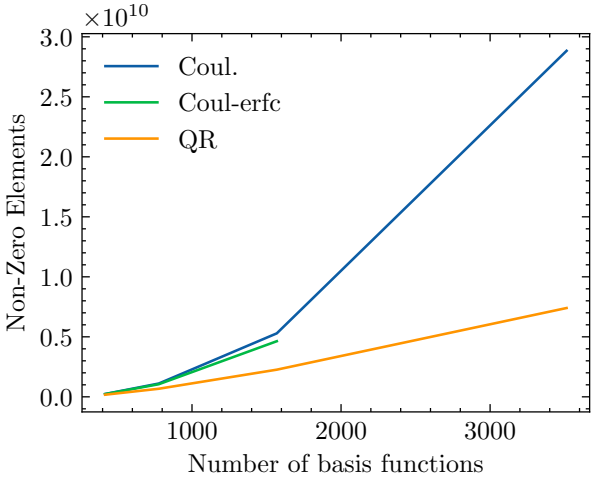


Figure 6.4: Figure

N_{AO}	Coul.	Coul-erfc	QR
417	—	—	—
777	2.50	2.45	2.15
1569	2.23	2.11	1.73
3513	2.10	—	1.47

Table 6.4: Table

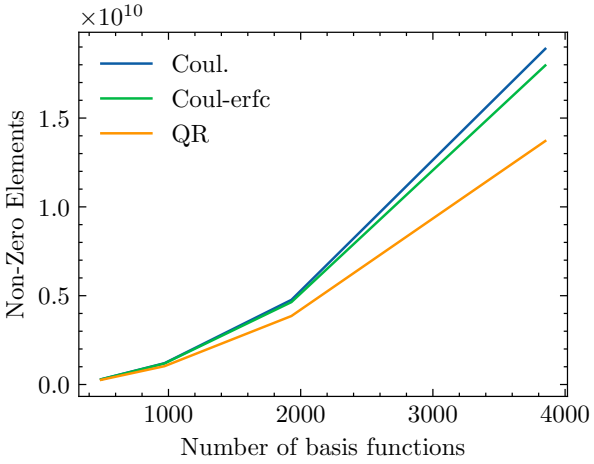


Figure 6.5: Figure

N_{AO}	Coul.	Coul-erfc	QR
490	—	—	—
970	2.07	2.06	2.01
1930	2.01	1.98	1.92
3850	1.99	1.96	1.84

Table 6.5: Table

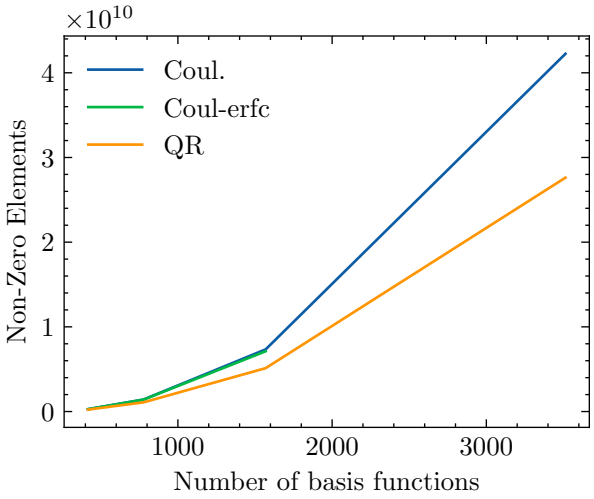


Figure 6.6: Figure

N_{AO}	Coul.	Coul-erfc	QR
417	—	—	—
777	2.27	2.26	2.20
1569	1.96	1.94	1.88
3513	1.71	1.68	1.57

Table 6.6: Table

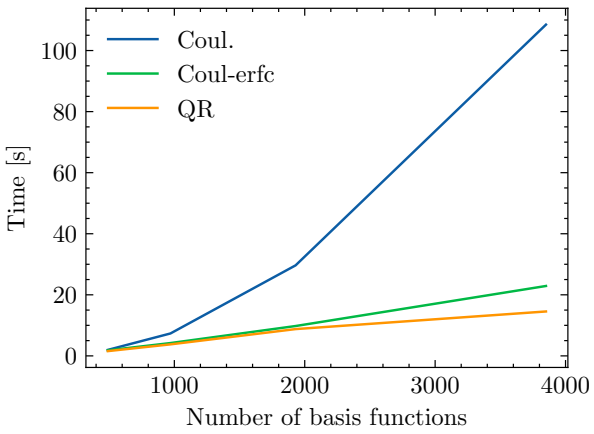


Figure 6.7: Figure

N_{AO}	Coul.	Coul-erfc	QR
490	—	—	—
970	1.93	1.23	1.29
1930	2.03	1.23	1.23
3850	1.88	1.23	0.73

Table 6.7: Table

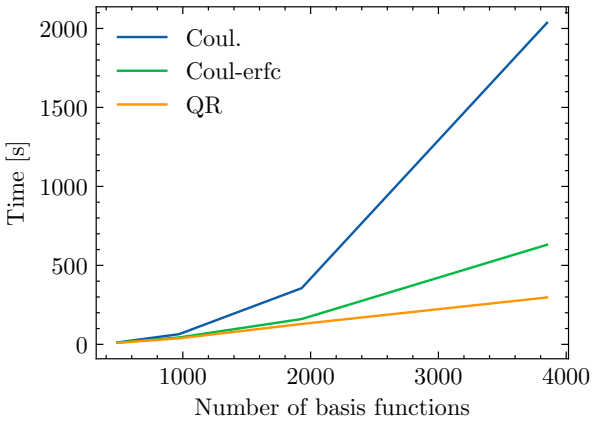


Figure 6.8: Figure

N_{AO}	Coul.	Coul-erfc	QR
490	—	—	—
970	2.43	1.97	1.96
1930	2.49	1.89	1.78
3850	2.53	1.98	1.21

Table 6.8: Table

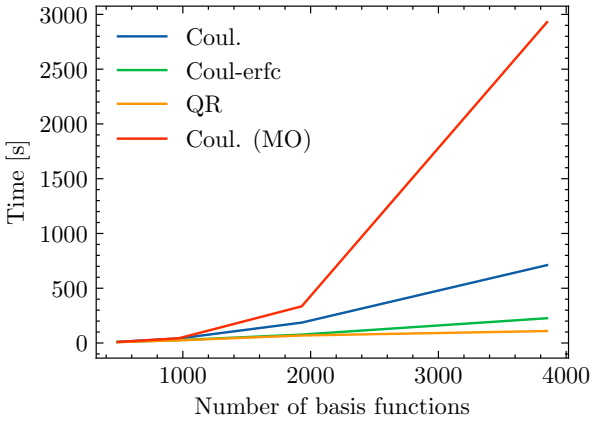


Figure 6.9: Figure

N_{AO}	Coul.	Coul-erfc	QR	Coul. (MO)
490	—	—	—	—
970	1.58	1.58	1.60	2.34
1930	1.57	1.57	1.50	2.96
3850	1.56	1.56	0.67	3.14

Table 6.9: Table

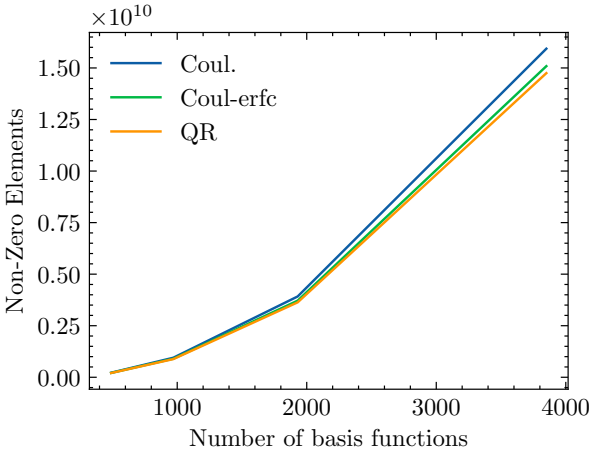


Figure 6.10: Figure

N_{AO}	Coul.	Coul-erfc	QR
490	—	—	—
970	2.14	2.13	2.12
1930	2.06	2.06	2.06
3850	2.03	2.03	2.03

Table 6.11: REDO THIS!!!!

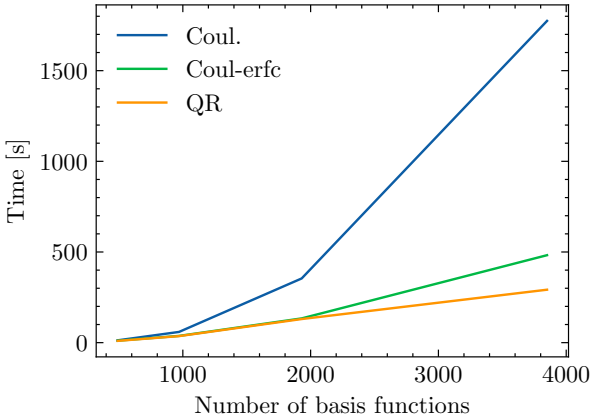


Figure 6.11: Figure

N_{AO}	Coul.	Coul-erfc	QR
490	—	—	—
970	2.45	1.96	2.09
1930	2.55	1.82	1.83
3850	2.00	1.59	1.00

Table 6.12: REDO THIS!!!!

N_{AO}	Coul.	Coul-erfc	QR
490	—	—	—
970	2.51	1.88	1.88
1930	2.59	3.18	2.36
3850	3.26	3.18	—

Table 6.13: Here

Chapter 7

Parallel Computing

The popularity of computational chemistry can be attributed in no small part to the advances and devlopment of highly efficient algorithms in theoretical chemistry. Equally important however is the ever increasing accessibility and performance of computing resources: commercially available work stations can handle chemical systems which could only be modelled on supercomputers a couple decades ago, and firmly cemented the position of computational chemsitry as an important "experimental" tool in the toolbox of a chemist.

As the speed of computers increased over the years, so did the complexity of their components. Nowadays, programmers can choose between several types of architectures, such as shared or distributed memory systems, or accelerators like GPUs. Knowing the strenths and weaknesses of each type is paramount to developing efficient algorithms and tackling larger molecular systems.

This chapter gives an overview on computer architecture, and the different types of parallelism encountered on modern hardware.

7.1 Moore's Law

Moore's Law states that the transistor density in integrated chips doubles every 12 to 24 months. First formulated in 1965 by Gordon More, his prediction has held up fairly well over the years. However, the technology enabling this trend has changed over the years.

Figure ... shows the trends in clock speed, single-thread performance, power consumption and number of logical cores and transistors for microprocessors from 1970 to 2000. Since the early 2000s, clock-speed and single-thread performance have begun to plateau, and have stagnated from 2010 onwards. Increasing the clock speed to values beyond 4 to 5 GHz generates too much stress on the microchip in form of heat, and decreases its performance. This flaw was compensated by using the growing transistor density to instead increase the number of logical cores on a single chip.

Shifting towards increasing core count however entails that the most ideal perfpormance of a CPU can only be achieved though parallel programming. With the rise of parallel computing, the number of different parallel hardware features drastically increased, and it can be difficult for programmers to fully exploit the available computing resources. Moreover, different programming language and compiler extensions have emerged alongside, with numerous competing standards, especially for GPUs.

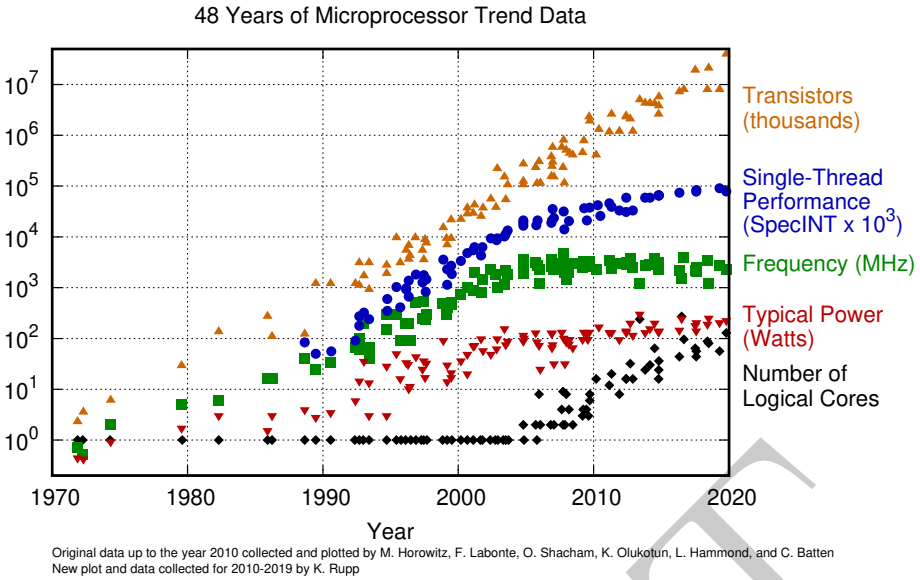


Figure 7.1: Taken from `https://github.com/karlrupp/microprocessor-trend-data`

7.2 Benefits and Limits of Parallel Computing

While the different available programming models can seem daunting at first, one of the major advantages of parallel computing is improved *scalabilty*. An application that exposes parallelism can be sped up by several orders of magnitude, simply by adding more compute power, with several different architectures to choose from. The limit of what problem sizes can be tackled is mostly dictated by the *amount* of available computing resources and storage, rather than individual processor charcterstics.

As important as parallel computing has become in recent years, there is a reason why increasing clock speed was seen as the foremost strategy in keeping Moore's law alive. First, modifying a serial program to exploit parallelism can be a time-consuming endeavour, and second, not all tasks can be effectively parallelized. This means that the potential amount of seed-up is limited by the amount of parallel code. This is known as *Amdahl's law*. The speed-up for a number of cores N_c is given by

$$Speed - Up(N_c) = \frac{1}{S + \frac{P}{N_c}} \tag{7.1}$$

where S is the fraction of serial code and P is the fraction of parallel code. The speedup for a fixed-size problem as a number of cores is known as *strong scaling*, and the time-to-solution on each indivual core *decreases* when more cores are added.

An alternate way to compute potential speed-up is given by Gustafson-Barsis's Law

$$Speed - Up(N_c) = N_c - S(N_c - 1) \tag{7.2}$$

where the problem size also increases proportionally to the number of cores. The scaling for this trend is known as *weak scaling*. In this scenario, the time-to-solution spend on each core remains constant, as the system size and number of cores increases. Even if this type is called "weak", both forms of scaling are equally important, as they adress different scenarios.

7.3 Types of Parallelism and Memory Hierarchy

Nowadays, a programmer has access to four categories of parallelism:

1. vectorization
2. thread-based parallelism
3. process-based parallelism
4. streaming

Leveraging the power of each type requires some understanding of the underlying hardware.

Figure 7.2 shows the major components and memory pathways in a modern computing cluster. A *cluster* is a collection of individual computers that work together and form a single unit. Individual computers are also called *nodes* and occupy a single rack (or "shelf") each in a large server cabinet. The nodes are connected via a low-latency, high through-put network, e.g. ethernet cables to enable inter-node communication. Each node contains one or more central proecessing units (CPU) and optionally one or more graphical processing units (GPU). Systems where different types of hardware architecture are mixed are also known as *heterogenous* systems. The individual components are fixed on a *motherboard*: CPUs are plugged into *sockets* and GPUs into *PCIe slots*. A CPU is composed of one or more cores, where the actual processing of data is carried out. A GPU is also composed of multiple cores, which are grouped into independent *streaming multiprocessors* (SM -¿ nvidia CU (compute units OpenCL, subslices).

Memory is also a crucial component of computer architecture and is a limited resource. The speed at which data is read from memory can also become a major bottle-neck: No matter how fast a processor is, if the feed rate is too low, it cannot reach its peak performance becuase it wastes cycles while waiting for data to arrive. To optimize data through-put, the memory model in modern computer architecture requires a complex hierarchy, with different sizes and speeds. Computer memory at the top of the hierarchy has a high response rate, but low complexity. It is also very expensive to produce and therefore much smaller. At the bottom of the hierarchy is memory with large storage space and capable of complex tasks. It is cheap but has low reponse rate. Over the years, the number of levels in the memory hierarchy has increased. Most modern computers have six levels: CPU registers, L1 chache, L2 chache, L3 chache, DRAM and disk.

CPU registers sit at the top of the hierarchy and are closest to the cores. It is the region of memory where data is directly manipulated by arithmetic operations and machine code. Its size is typically on the order of several tens to hundreds of bytes. Data is loaded into the registers from *cache*, a region of memory which is further subdivided into different levels named L1, L2 and L3. Each core has its own L1 cache, but might share L2 and L3 cache with other cores. Caches have different sizes and speeds, with L1 being the fastest and smallest at several tens of kB, and L3 being the slowest and largest at several tens to hundreds of kB. Memory is transferred from L3, to L2, to L1 and finally to the CPU registry. The reason why there are multiple levels of chache is to reduce *cache misses*. If data requested by the core is not found in L1, then L2 is searched, then L3. A cache miss is an event where the data is not found anywhere in chache. In that case, a request has to be put out to the dynamical random access memory (DRAM) to retrive

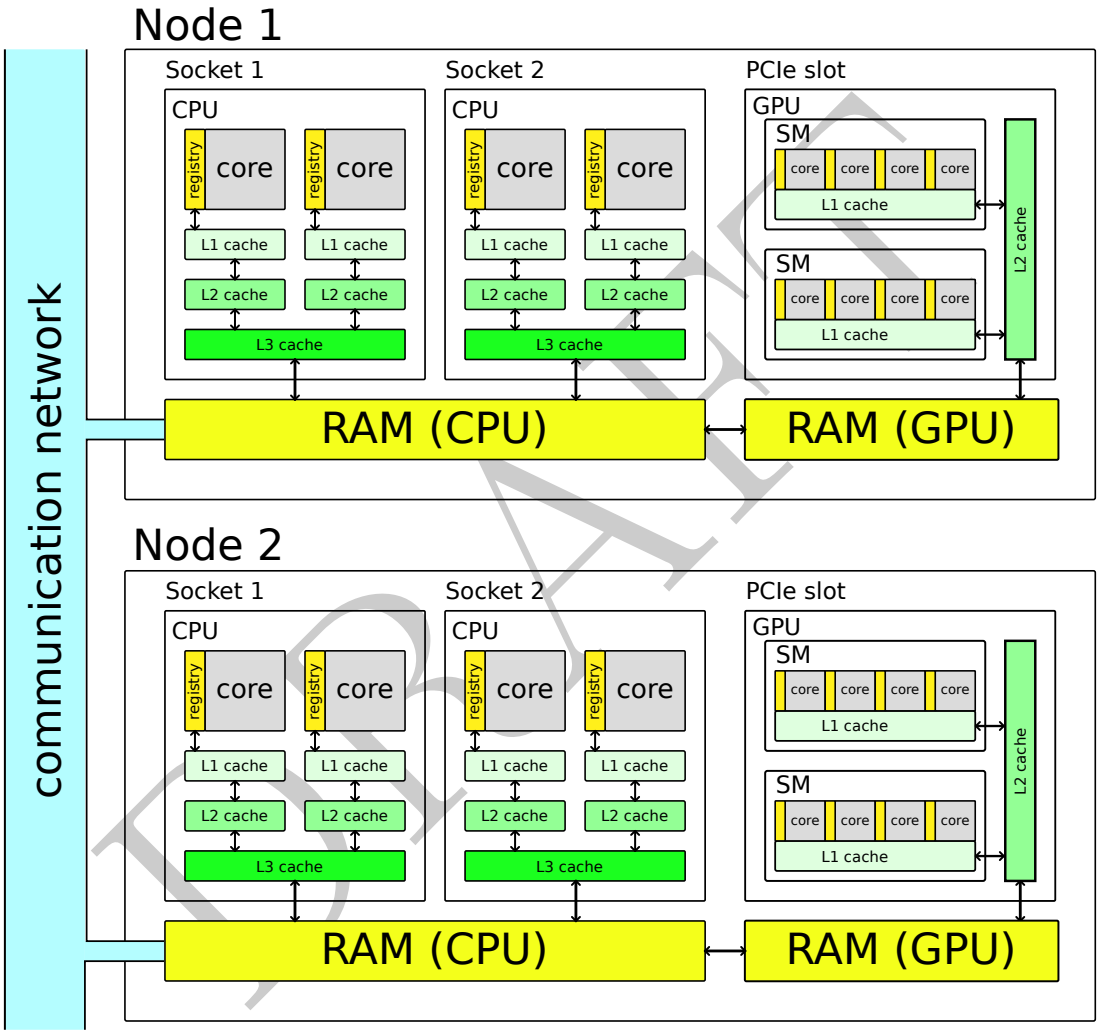


Figure 7.2: Memory hierarchy

Release	Vector Length (bit)	No. Registers
SSE (Streaming SIMD Extension)	128	8
SSE2/SSE3/SSE4	128	16
AVX/AVX2 (advanced vector instructions)	256	16
AVX512	512	32

Table 7.1: Adpated from ...

data. Modern techniques such as *chache prefetching* can minimize the amount of cache misses by loading the data into higher chache levels before it is actually needed by the lower levels.

The speed of DRAM is 10 to 100 time slower than cache, but much larger in size. It is the main memory pool and shared by all cores. CPUs and GPUs have separate DRAM regions which communicate via a PCIe bus. DRAM sizes vary drastically, and are on the order of 10^0 to 10^1 for GPUs and 10^0 to 10^3 for CPUs. Data can be transferred from one node to another via CPU DRAM through the communication network, with transfer rates on the order of several GB/s. For programs which are not reading from disk, this is weakest link in the memory hierarchy ???

7.4 Vectorization

Vectorization is the pocess of operating on multiple variables at the same type. This type of parallelism is encountered at the highest level of the memory hierarchy introduced in the previous section, i.e. CPU registers. Each core has multiple registers, also called *vector registers* with a certain size or *vector length*. Instead of loading each inidividual element from cache and operating on it, one vector operation on a range of elements can replace multiple single operations. For a 512-bit register, two vectors with 8 floats (32 bit) can be summed within one cycle instead of eight (Figure ...). This type of parallelism is also known as single instruction multiple data (SIMD).

The length of vector registers, number of registers as well as the number of supported vector operations have greatly expanded over the years (Table 7.1).

7.4.1 Parallel SAXPY using vectorization

To show how vectorization can be introduced into a program, consider the following vector operation:

$$y \leftarrow \alpha x + y \tag{7.3}$$

where y, x are vectors of equal length, and α is a scaling factor. The vector operation 7.3 is also known as "saxpy" for single precision, and "daxpy" for double precision. A naive implementation of the saxpy-kernel is given in Listing 7.4.1 for vector size N

Listing 7.4.1: Parallelization-unaware implementation of saxpy (`saxpy_nopara.c`)

```
1  #define ARRAY_SIZE 1024
2
3  static float x[] = {[0 ... ARRAY_SIZE] = 1.0};
4  static float y[] = {[0 ... ARRAY_SIZE] = 2.0};
5  static float a = 3.0;
6
7  int main() {
8
9    for (int i = 0; i < ARRAY_SIZE; ++i) {
10     y[i] += a * x[i];
11   }
12
13 }
```

Two arrays are allocated, x and y, and all their entries set to 1 and 2 respectively. Each element is updated individually in the for loop. There are several possibilities to introduce vectorization:

1. auto-vectorization
2. compiler directives
3. intrinsic functions
4. optimized libraries

Auto-vectorization is by far the easiest approach: the compiler automatically recognizes that the loop can be vectorized and generates optimized machine code that uses vector instructions. This does not require any input from the user. Compiling the code on a machine with AVX support, using the GNU C compiler, and passing the compiler flags "-O2 -march=native -ftree-vectorize -fopt-info-vec-optimized" generates the following report:

```
saxpy_nopara.c:9:3: optimized: loop vectorized using 32 byte
   vectors
```

which indicates that the vectors are loaded and operated on in 32 byte chunks, or 8 floats at once. The flag "-ftree-vectorize" (or alternatively "-O3") activates auto-vectorization and "-fopt-info-vec" generates the report. To make sure that the C compiler uses the right vectorization relase, the flag "-march=native" is needed, or else the comiler might fall back to SSE.

In some cases, auto-vectorization cannot take place because the compiler did not recognize that the loop can be vectorized. It can then be beneficial to use *intrinsic functions*. Intrinsic functions are compiler-dependent functions that map to processor

operations. When targeting an AVX architecture with 256-bit registers, the saxpy kernel can be rewritten as

Listing 7.4.2: SAXPY using intrinsics (`saxpy_intrinsic.c`)

```
1  #include <immintrin.h>
2  #define ARRAY_SIZE 1024
3
4  // attribute needed for alignment, misalignment leads to
       errors
5  static float x[] __attribute__ ((aligned(8*ARRAY_SIZE))) = {[0
       ... ARRAY_SIZE] = 1.0};
6  static float y[] __attribute__ ((aligned(8*ARRAY_SIZE))) = {[0
       ... ARRAY_SIZE] = 2.0};
7  static float a = 3.0;
8
9
10 int main() {
11
12   __m256 a_vec, x_vec, y_vec, r_vec;
13
14   // set each entry of a_vec to a
15   a_vec = _mm256_set1_ps(a);
16
17   int stride = 8;
18   for (int i = 0; i < ARRAY_SIZE; i += stride) {
19     // load values into registers with appropriate offset i
20     x_vec = _mm256_load_ps(&x[i]);
21     y_vec = _mm256_load_ps(&y[i]);
22
23     // perform saxpy: (1) multiply, (2) add
24     r_vec = _mm256_add_ps(_mm256_mul_ps(a_vec, x_vec), y_vec);
25
26     // copy results back to y
27     _mm256_store_ps(&y[i], r_vec);
28   }
29
30 }
```

It is apparent that using intrinsics makes the program much more complex. The arrays cannot be fed directly to the functions, but need to be loaded into vectors of type `__mm256`, using "set" or "load" functions. Furthermore, the data needs to be aligned correctly using `__attribute__ ((aligned(...)))`. The arrays are then loaded in chunks into the registers and given to the vector functions for multiplying and adding.

The major problem with using intrinsic functions, besides increased complexity, is *portability*. The code in 7.4.1 does not compile on machines that do not support AVX, and is limited to 256-bit registers even on AVX-512 machines. Portable alternatives include using compiler directives or optimized libraries.

Directives (or "pragmas") are hints that can be given to the compiler that suggest that the loop might be vectorizeable. By far the most popular set of compiler directives that provide vectorization capabilities is undoubtedly included in the OpenMP application programming interface (API). The OpenMP API is standardized accross all compilers, making it highly portable. The OpenMP directives greatly simplify the SAXPY program:

Listing 7.4.3: SAXPY using compiler directives (`saxpy_simd.c`)

```
1  #define ARRAY_SIZE 1024
2
3  static float x[] = {[0 ... ARRAY_SIZE] = 1.0};
4  static float y[] = {[0 ... ARRAY_SIZE] = 2.0};
5  static float a = 3.0;
6
7  int main() {
8
9    #pragma omp simd
10   for (int i = 0; i < ARRAY_SIZE; ++i) {
11     y[i] += a * x[i];
12   }
13
14 }
```

Simply plopping the directive in front of the for loop takes care of generating the appropriate machine code for the targeted architecture.

The last way to introduce vectorization is via external programs, such as the basic linear algebra subprograms (BLAS) library. It provides a set of specific functions for performing basic vector and matrix operations. Similar to OpenMP, it only provides spcifications, and the direct implementation is compiler-dependent. In the BLAS routines, there is a saxpy functions available that can be called directly. It has the advantage of completely removing the loop and clearly states what operation is performed.

Listing 7.4.4: SAXPY using BLAS (`saxpy_blas.c`)

```
1  #include <cblas.h>
2  #define ARRAY_SIZE 1024
3
4  static float x[] = {[0 ... ARRAY_SIZE] = 1.0};
5  static float y[] = {[0 ... ARRAY_SIZE] = 2.0};
6  static float a = 3.0;
7
8  int main() {
9          cblas_saxpy(ARRAY_SIZE, a, x, 1, y, 1);
10 }
```

External libraries can however be associated with a steeper learning curve depending on the complexity of function signatures.

7.5 Thread-based Parallelism

Vectorization is limited to single cores only. For multicore processing, it is important to understand the concept of processes and threads. Processes are executing instances of programs and group related operating system resources together. These resources are exclusive (private) to the process which allocated them, such as system memory, file handles, I/O status information, scheduling information and accounting information.

A process spawns one or more *threads* (Figure 7.3). A thread is the smallest subset of a process that can be scheduled independently by the OS scheduler. Unlike processes, threads of the same process share resources. They can be seen as "light-weight" processes: start-up of individual threads and communication between threads is much faster. Each core executes only one thread at a time, but can quickly switch between different threads (or *contexts*). Threads with a higher *priority* are granted more CPU time than threads with lower priority. One a single-core processor, threads are only executed *concurrently*, i.e. they are paused and resumed at regular intervals depending on their priority. One multi-core processors, threads can be executed at the same time (multithreading), but concurrency is still needed if the number of threads exceeds the number of logical cores.

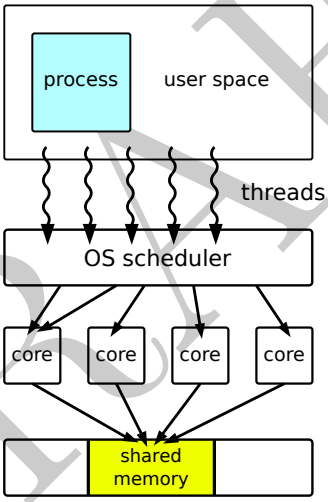


Figure 7.3: Shared memory parallelism

7.5.1 SAXPY using OpenMP

The most popular standard for thread-based parallelism (or *shared-memory parallelism*) is OpenMP, which was briefly discussed in the previous section. OpenMP was originally introduced to parallelize highly regular loops, but the standard has since then been greatly expanded and includes vectorization directives as well. Using pragmas, multiple threads can be spawned that execute the code within the parallel region. Lising ... shows an OpenMP parallel version of SAXPY. Each thread in the parallel region has a unique number associated with it which is used to divide up the arrays. Caution needs to be taken that threads do not operate on the same region at once, as this can lead to undefined behaviour ("race conditions").

Listing 7.5.1: SAXPY using OpenMP (saxpy_omp.c)

```
#include <omp.h>
#define ARRAY_SIZE 1024

static float x[] = {[0 ... ARRAY_SIZE] = 1.0};
static float y[] = {[0 ... ARRAY_SIZE] = 2.0};
static float a = 3.0;

int main() {

  // launch threads
  #pragma omp parallel
  {
    // The code in this region is executed by ALL threads
    int threadnum = omp_get_thread_num();
    int numthreads = omp_get_num_threads();

    // lower and upper bound for this thread are determined
       using the thread number and number of threads
    int lb = ARRAY_SIZE*threadnum/numthreads;
    int ub = ARRAY_SIZE*(threadnum+1)/numthreads;

    for (int i = lb; i < ub; ++i) {
      y[i] += a * x[i];
    }

  } // end parallel region, synchronize

  // tasks share memory, and the result is visible here

}
```

Alternatively, the code may be more easily expressed by using a single compiler directive:

Listing 7.5.2: SAXPY using OpenMP (saxpy_omp2.c)

```
#define ARRAY_SIZE 1024

static float x[] = {[0 ... ARRAY_SIZE] = 1.0};
static float y[] = {[0 ... ARRAY_SIZE] = 2.0};
static float a = 3.0;

int main() {

  #pragma omp parallel for
  for (int i = 0; i < ARRAY_SIZE; ++i) {
```

```
11      y[i] += a * x[i];
12    }
13
14  }
```

The `omp parallel for` directive also offers different scheduling tatctics which may be difficult to programm by hand. Listing 7.5.1 is an example of *static scheduling* the work is divided equally among threads *a priori*. However, in the case where each step requires different computational cost, *dynamic scheduling* offers a more balanced approach, although with a somewhat higher prefactor. Threads request tasks, or a chunk of multiple tasks, from the main thread, which distributes work on a first-come first-serve basis. There are many more directives available in the OpenMP standard with a lot of options for fine-tuning. For further details, the reader is referred to the official specifications (ref)

OpenMP is not the only option available for introducing shared-memory parallelism into a program. Alternatives include the `pthread` (POSIX threads) library for C, the `std::thread` library, or the Intel TBB (thread building blocks) library for C++. While these libraries can offer a more fine-grained control of threads, the learning curve can be very steep, and parallelizing an application can be very time-consuming. In contrast to OpenMP, they are not based on high-level compiler directives, but rather expose low-level functions and structures that create and control threads.

Shared-memory parallelism, as the name implies, is limited to cores sharing the same memory space. The primary disadvantage of this model is scalability. It becomes increasingly expensive to produce shared memory systems with more and more cores. Moreover, a larger number of cores implies heavier CPU traffic and time penalties due to *cache coherence*. Cache coherence is the mechanism by which uniformity of data is guaranteed and propagated among processes. There can be multiple copies of the same memory region in chache, which is operated on by different cores. Checking how to combine data from different caches can become costly for a large number of cores.

7.6 Process-based Parallelism

One of the most scalable aproaches in parallel computing is process-based parallelism, also known as distributed memory parallelism. Multiple copies of the same program run on separate processes, each with thier own local memory and instruction set. In contrast to threads, processes cannot exchange data via shared memory, but need to communicate via a network (Figure ...) using *message passing*. The increased cost of communication is often outweighed by the increased potential for scalability. Larger problems can be tackled simply ba adding more nodes.

Pure message-passing assigns or *binds* one rank to a single CPU core. In hybrid parallel approaches, a single process can be bound to a whole node, using shared-memory parallelism for intra-node communication and message passing for inter-node communication. It is also possible to spawn more processes than there are available cores: in that case, the processes are scheduled and excuted concurrently similar to threads.

The most popular standard for message passing is the message passing interface

(MPI). It defines a set of functions that allow processes to send and receive messages. There are different implementations of MPI, such as the open-source implementations OpenMPI, MPICH, or the vendor implementations Intel MPI or Cray MPI. MPI programs are run using a special start-up command:

```
mpirun -n <nprocs> ./myprogram.exe
```

The exact command and options depends on the MPI implementation. The start-up program is reponsible for duplicating the program on the different processes and establishing the communication network. Most implementations use the flag `-n` to pass the number of processes to be spawned.

All processes are characterized by their *rank*, a unique, portable identifier, normally an integer between $[0 : nprocs - 1]$. Only processes using the same *communicator* can exchange messages. A communicator is a special handle of type `MPI_Comm` and describes a group of processes. Communicators may also have different *topologies*, such as cartesian grids, or graphs, which restrict the flow communication between processes to nearest neighbours. Cartesian grids for example are best suited for matrix operations (cf. section ...). By passing topology information to MPI, it can optimize the runtime environemnt by renumbering tasks such that processes are physically closer to lower communivation overhead. By default, the topology is undefined, and any processor can exchange messages with all other processors.

7.6.1 SAXPY using MPI

Modifiying programs to exploit distrubuted parallelism requires significant effort, the most crucial point being how data is distributed over the processes. Consider again the SAXPY kernel. There are two memory models: either every process has a copy of the whole array, or the arrays are distrubuted in chunks over all processes. If data is duplicated on every rank, this can quickly exhaust memory resources for large datasets. On the other hand, if data is distributed in a non-ideal way, communation may incur major overhead. A programmer has to outweigh the benefits of data duplication and network communication.

Listing 7.6.1 shows a distributed memory approach to the SAXPY kernel

Listing 7.6.1: SAXPY using MPI (`saxpy_mpi.c`)

```
1  #include <mpi.h>
2  #include <stdlib.h>
3  #include <stdio.h>
4  #define ARRAY_SIZE 1024
5
6  static float a = 3.0;
7
8  int main(int argc, char** argv) {
9
10   MPI_Init(&argc,&argv);
11   MPI_Comm comm = MPI_COMM_WORLD;
12
```

```
13   int rank, size;
14   // get information form communicator
15   MPI_Comm_rank(comm, &rank);
16   MPI_Comm_size(comm, &size);
17
18   // size needs to be a divisor of ARRAY_SIZE
19   if (ARRAY_SIZE % size != 0) exit(-1);
20   int num_local = ARRAY_SIZE / size;
21
22   // allocate local arrays on each process
23   float* x_loc = (float*)malloc(num_local*sizeof(float));
24   float* y_loc = (float*)malloc(num_local*sizeof(float));
25
26   // global arrays
27   float *x, *y;
28
29   if (rank == 0) {
30         // init data on rank 0
31         x = (float*)malloc(ARRAY_SIZE*sizeof(float));
32         y = (float*)malloc(ARRAY_SIZE*sizeof(float));
33         for (int i = 0; i < ARRAY_SIZE; ++i) {
34                 x[i] = 1.0; y[i] = 2.0;
35         }
36   }
37
38   // send to other processes
39   MPI_Scatter(x,num_local,MPI_FLOAT,x_loc,num_local,MPI_FLOAT
        ,0,comm);
40   MPI_Scatter(y,num_local,MPI_FLOAT,y_loc,num_local,MPI_FLOAT
        ,0,comm);
41
42   // perform operation on local arrays
43   for (int  i = 0; i < num_local; ++i) {
44           y_loc[i] += a * x_loc[i];
45   }
46
47   // gather data on rank 0
48   MPI_Gather(y_loc,num_local,MPI_FLOAT,y,num_local,MPI_FLOAT
        ,0,comm);
49   // print it out, store it ...
50
51   // clean up
52   free(x_loc); free(y_loc);
53   if (rank == 0) {
54           free(x); free(y);
55   }
56
57   MPI_Finalize();
```

```
58
59 }
```

The program starts by initializing the executing environment using `MPI_Init`, and defining the communicator handle. The communicator that groups all processes is called `MPI_COMM_WORLD`, and is a macro defined in the header `<mpi.h>`. If the programm is stand-alone, it is safe to use as the primary communicator. When writing a library that is used in combination with other MPI libraries, it is crucial to use a communicator that is passed from the main program. There are two reasons for this: (1) the user may want to use a subcommunicator, rather than the whole process group, and (2) MPI messages have a unique identifier (integer) and if another library uses the same identifier, erroneous communication can take place. In this example, the global communicator is suffcient.

Each process then allocates a local array of size `num_local` < `ARRAY_SIZE`. In this example, the main data is intialized on rank 0, which is then split into equal chunks and send to the other processes' local array using `MPI_Scatter`. Each process then only needs to perform the SAXPY operation on its local chunk. Afterwards, the results are collected on rank 0 using the routine `MPI_Gather`. Data is then available on rank 0 to be further manipulated, written to console, etc.

It should be noted that the above program only works for a number of processes that is a divisor of `ARRAY_SIZE`, because `MPI_Scatter` and `MPI_Gather` can only handle chunks that are of equal size on each process. If that is not the case, one can use the more general routines `MPI_Scatterv` and `MPI_Gatherv` which also take the different chunk sizes as an input.

Furthermore, the local SAXPY for loop can be parallelized using the techniques described in the previous sections (vectorization, threads) in case where a process is bound to a whole node and has access to further compute resources.

This programm shows only a few of the many functions available for message passing. The MPI standard defines many more functions for sending, receiving, broadcasting, reduction, one-sided communication etc. The official document for the newest MPI standard (ref) is over a thousand pages long, and is updated regularly. Message passing has a very steep learning curve, and can be quite difficult to debug.

7.6.2 MPI and Shared Memory

One may come under the impression that MPI is not efficient on shared-memory systems, as data is exchanged via network-calls. However, MPI is optimized to recognize shared memory and data is then copied within DRAM rather than send through a comunication layer, making MPI competitive even on single nodes. Of course, there is still a slight overhead associated with it compared to thread-based parallelism.

MPI standards of version 3.0 and above furthermore introduce functions to allocate shared memory, also known as *windows*, for processes on the same node. Processes can then read/write to the same memory region without using network calls, much in the same manner as OpenMP. This approach can be seen as a MPI+MPI hybrid parallel approach rather than the more often used MPI+OpenMP approach. Hybrid parallelism is much more efficient than "MPI-everywhere" approaches due to reduced communication

overhead and easier load-balancing.

MPI+MPI has the advantage of only needing one programming model for both distributed and shared memory access, with relatively easy syntax. However, MPI+MPI is much lesser known, and there are is no automatic support for splitting up loops or atomic loads/stores to shared memory. An external library that offers auxiliary functions for MPI+MPI would be most beneficial.

7.7 Stream Processing

The last type of parallelism tackled in this chapter is *stream processing*. A stream is defined as a sequence of data that requires similar (low-level) computation. Streams are processed by *accelerators*, special hardware components with high data throughput, that complement the general purpose CPU by speeding up certain operations.

7.7.1 GPU Architecture

By far the most popular type of accelerators are graphical processing units (GPUs). As the name implies, GPUs were originally conceived to speed up graphics-related computation, but their use was later extended to include non-graphics workloads as well, in what is known as *general-purpose* graphic processing unit (GPGPU) programming. Nowaydays, computing clusters increasingly come with one or more GPGPUs included on each node.

The hardware archituecture design of modern GPUs is much more varied than that of CPUs. GPU vendors, like NVIDIA or AMD, have different hardware variations and use different terminology for similar components, and it can be difficult to abstract hardware as some vendor-specific features need to be omitted. Figure 7.4 shows a simplified block diagram of the most common GPU components. A GPU is composed of multiple compute units (CUs), also known as streaming multi-processors (SMs) in NVIDIA GPUs. Work is assigned to different CUs by a workload distributor. Each CU is further subdivided into an array of processing elements (PEs), or CUDA cores as referred to by NVIDIA. A single PE can operate on multiple data elements at once using SIMD or a variation known as single instruction multiple threads (SIMT) used in CUDA cores.

The overall performance of a GPU is given by the number of CUs and PEs, as well as the bandwidth of the PCIe land connecting CPU and GPU. Data transfer between the two devices is often the time-determinig step.

7.7.2 GPU Programming Model

The rise of GPGPU programming has given rise to a plethora of different programming languages and models. Low-level languages like CUDA, OpenCL or HIP directly reflect features of the targeted GPU hardware as part of their syntax, and the user is reponsible for managing each aspect of parallelism and data transfer. They require a very thorough understanding of how GPUs operate. Due to the complexity of "native" languages, higher-level languages have gained popularity over the years. Pragma based languages like OpenACC or OpenMP offer a much easier approach to GPU programming that ressembles shared memory parallelism on CPUs, and offer a higher degree of portability, at the cost of lower performance ceiling.

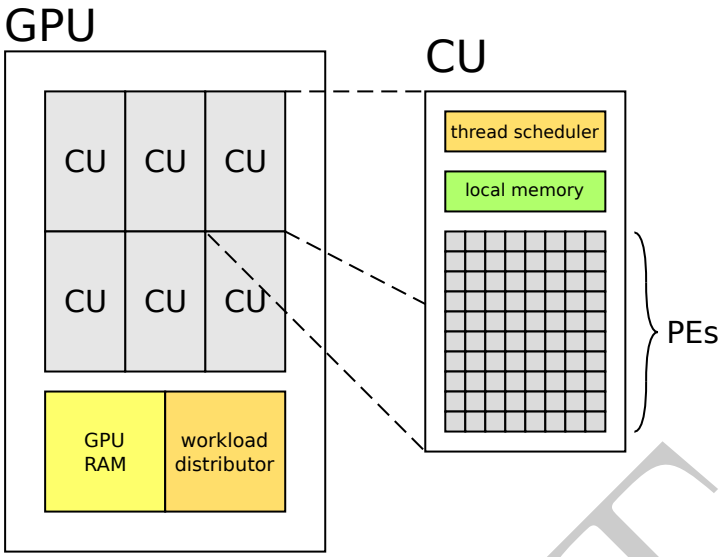


Figure 7.4: GPU architecture

Low level languages all operate on a similar programming model which allow the programmer to define special functions called *kernels*, which, when called on a GPU, are executed N times in parallel by N GPU threads. Rather than using for loops, kernels oparate on a whole range of data at once, known as a *grid* or NDRange. A grid can be one, two or three-dimensional to allow a more intuitive view of the data at hand. Each grid is subdivided into blocks or work groups which can be executed independently. The maximum size of a block is given by the number of threads in a CU. "Good" values are generally 128, 256 or 512, but the optimal size depends on the hardware and the problem addressed. Threads in a block share memory which is useful in case data from neighbouring subblocks is needed. Information on grid and block dimensions are needed by the kernels for distributing work over the GPU.

As an example, consider the SAXPY function using the CUDA programming language:

Listing 7.7.1: SAXPY using CUDA (saxpy_cuda.cu)

```
1  #define ARRAY_SIZE 1024
2  #define BLOCK_SIZE 256
3
4  static float x[ARRAY_SIZE];
5  static float y[ARRAY_SIZE];
6  static float a = 3.0;
7
8  // the identifier __global__ indicates that it should be
       launched on the GPU
9  __global__
10 void saxpy(int N, float d_a, float* d_x, float* d_y) {
11         int i = blockIdx.x*blockDim.x+threadIdx.x;
12         if (i < N) d_y[i] = d_a*d_x[i] + d_y[i];
13 }
```

```
14
15 int main() {
16
17         // initialize memory on CPU
18         for (int i = 0; i < ARRAY_SIZE; ++i) {
19                 x[i] = 1.0; y[i] = 2.0;
20         }
21
22         // allocate memory on GPU
23         float *d_x, *d_y;
24         cudaMalloc(&d_x, ARRAY_SIZE*sizeof(float));
25         cudaMalloc(&d_y, ARRAY_SIZE*sizeof(float));
26
27         // copy data to GPU
28         cudaMemcpy(d_x, x, ARRAY_SIZE*sizeof(float),
              cudaMemcpyHostToDevice);
29         cudaMemcpy(d_y, y, ARRAY_SIZE*sizeof(float),
              cudaMemcpyHostToDevice);
30
31         // launch kernel
32         int nblocks = (ARRAY_SIZE + BLOCK_SIZE) / BLOCK_SIZE;
33         saxpy<<<nblocks,BLOCK_SIZE>>>(ARRAY_SIZE, a, d_x, d_y)
              ;
34
35         // copy back
36         cudaMemcpy(y, d_y, ARRAY_SIZE*sizeof(float),
              cudaMemcpyDeviceToHost);
37
38         // free up memory on GPU
39         cudaFree(d_x);
40         cudaFree(d_y);
41
42 }
```

CPU and GPU do not share memory. The first steps therefore involve allocating memory using `cudaMalloc` and copying data from CPU ("Host") to GPU ("Device") with `cudaMemcpy`. The arrays can then be passed to the kernel. The kernel function is indicated by the identifier `__global__`, and is executed at the same time by all threads in a block. Each thread has a unique block and thread id which indicate its position in the work group. This information is used to make sure that the thread is operating on the correct portions of the array. The GPU threads do not have any information of the loop extent, and it is the user's responsibility to make sure that threads do not write past the allocated memory window by introduing boundary checks (`if (i < N)`).

In this example for a 1D array, the kernel takes two launch parameters: the number of blocks, and the block size which are put between `<<< .... >>>` after the function call. Afterwards, the data can be copied back to the CPU for further processing.

The concept of operating relative to thread coordinates rather than using a for

loop with a single index can be quite confusing. Nowadays, there are directive-based compiler extensions, that allow to easily offload work to accelerators, such as OpenMP and OpenACC (Open ACCelerators). Listing 7.7.2 shows how a for loop can be easily modified using OpenACC

Listing 7.7.2: SAXPY using OpenACC (`saxpy_acc.c`)

```
1  #define ARRAY_SIZE 1024
2
3  static float x[] = {[0 ... ARRAY_SIZE] = 1.0};
4  static float y[] = {[0 ... ARRAY_SIZE] = 2.0};
5  static float a = 3.0;
6
7  int main() {
8
9    #pragma acc data copy(x,y,a)
10   {
11     #pragma acc parallel loop
12     for (int i = 0; i < ARRAY_SIZE; ++i) {
13       y[i] += a * x[i];
14     }
15   }
16
17 }
```

OpenACC offers several pragmas for allocating and copying arrays to GPUs and back. By only adding two directives, the SAXPY code is easily ported to GPUs.

7.7.3 When to Use GPUs

Despite the success of GPU computing, GPUs only complement, rather than replace CPUs. GPU cores are very light-weight and have a limited instruction set compared to CPU cores, making them ill-suited for *task-based parallelism*, e.g. running an operating system with many different processes executing in the background. The strength of GPUs rather lies in *data-based parallelism*. Generally, any code that can benefit from vectorization can be sped up using GPUs.

Chapter 8

Into The Matrix

An overview of matrix algebra libraries

Chapter 9

MEGALOchem: User Guide

A user guide for MEGALOchem

Chapter 10

MEGALOchem: Developer's Guide

Developer's Guide for MEGALOchem, more details

Chapter 11

Annex

Everything else

- ERI deomposition: cholesky, THC, pseudo-spectral - The evil matrix inversion: considerations - mulliken, boughton pulay