PhD Thesis

Size matters: Enabling large–scale coupled cluster calculations

Patrick Ettenhuber

qLEAP - Center for Theoretical Chemistry
Department of Chemistry
Aarhus University
Denmark

April 30, 2015

Preface

This thesis describes the work I have done during my three years and four months as PhD student at the qLEAP Center for Theoretical Chemistry, Department of Chemistry, Aarhus University, under the supervision of Poul Jørgensen from February 2012 to April 2015. All the research projects I have worked on during my time as a PhD student have been linked to the linear–scaling Divide–Expand–Consolidate (DEC) coupled cluster (CC) method that had been developed in Aarhus and is implemented in the **LSDALTON** electronic structure program.

From the beginning, my work has been focused on the implementation of flexible, large–scale algorithms, targeting supercomputers. At this stage, the DEC algorithm has been operational for MP2, where an efficient multi–scale parallel scheme has been developed. The main focus, however has been a conventional parallel algorithm for solving the DEC–fragment CCSD vector equations on targeting supercomputers. From the start, technical challenges dominated the research and thus my focus shifted more and more towards the technical aspects of supercomputing, parallelization, portability of the code, and workarounds for many not–self–inflicted issues. During my time as a PhD student several visits of the Oak Ridge National Laboratory (ORNL) and one visit at the Argonne National Laboratory (ANL), USA, have broadened my understanding of supercomputing and parallelization. At ANL I have worked together with Jeff Hammond, where I have learned many details about the message passing interface (MPI); knowledge, which has been used to write a general framework for handling distributed tensors. With this library at hand, it has been simple to implement a solver algorithm with minimal memory footprint and use the technique in the CCSD algorithm in **LSDALTON**. Lately, I have been working together with Dmitry Liakh from ORNL in order to improve the performance of some of the tensor routines and as a consequence extent the application range of the, now working, CCSD implementation. Besides some technical challenges and setbacks, which clearly have dominated my time as a PhD student, I have been concerned with the accuracy of the DEC method when moving beyond the MP2 model and the utilization of techniques in order to reduce the pre–factor of a DEC calculation. *All* the details will be given in the individual chapters of this thesis, which for sure is an enjoyable piece of literature.

Papers

My PhD studies has resulted in the following articles, which are part of this thesis:

Published:

1. Kristensen, K., Kjærgaard, T., Høyvik, I.-M., Ettenhuber, P., Jørgensen, P., Jansik, B., Reine, S. and Jakowski, J.
   The divide-expand-consolidate MP2 scheme goes massively parallel
   Mol. Phys, **111**, 1196 (2013), http://dx.doi.org/10.1080/00268976.2013.783941

2. Ettenhuber, P. and Jørgensen, P
   Discarding Information from Previous Iterations in an Optimal Way To Solve the Coupled Cluster Amplitude Equations
   JCTC, **11**, 1518 (2015), http://dx.doi.org/10.1021/ct501114q

3. Kristensen, K., Ettenhuber, P., Eriksen, J. J., Jensen, F. and Jørgensen, P.
   The same number of optimized parameters scheme for determining intermolecular interaction energies
   J. Chem. Phys.,**142**, 114116 (2015), http://dx.doi.org/10.1063/1.4915141

Submitted:

4. Eriksen, J.J., Baudin, P.,Ettenhuber, P, Kristensen, K., Kjærgaard, T. and Jørgensen, P
   Linear-scaling coupled cluster with perturbative triples correction: The Divide-Expand-Consolidate CCSD(T) model
   In revision at JCTC, Manuscript ID: ct-2015-000869.R1

In preparation:

5. Ettenhuber, P, Baudin, P., Kristensen, K., Kjærgaard, T. and Jørgensen, P.
   Orbital spaces in the Divide-Expand-Consolidate coupled cluster method

Acknowledgements

First of all, I would like to thank my supervisor Poul Jørgensen for being the driving force behind all the research I have carried out during my PhD studies. Sometimes when it seemed to be impossible to continue, with his overall positive attitude Poul made it even more impossible to stop and thus the scientific progress, once more, could continue. The resulting many small fights I have had with him turned out to give me self–confidence in scientific discussions since it turned out that sometimes even a supervisor may be wrong. It has been a great experience and pleasure working together with Poul as a supervisor because of his engagement, profound knowledge and helpfulness at every stage of the research. Furthermore, I would like to thank Kasper Kristensen for being a formidable mini–supervisor and all–kinds–of–stuff corrector. Besides proofreading large parts of this thesis (in spite of the two–sided printouts) he has been the anti–Poul in many discussions, often realistic (at times downright pessimistic) but always to the point. Also, I would like to thank Thomas Kjærgaard, for many fruitful discussions, proofreading parts of this thesis and for stopping a conversation when there was no point in prolonging it. I would like to thank Janus Juul Eriksen for improving the language in parts of this thesis and correcting many of my spelling mistakes. Derudover vil jeg meget gerne takke Ian Heide Godtliebsen for at hjælpe med den danske overvsættelse af resuméet af denne afhandling, som skal altså virkelig læses på dansk. Also, I would like to thank Yang Min Wang for proofreading and (sometimes) doing a good job in running the Theoretical Chemical Fridays Bar (TCFB). I thank Dmytro Bykov and Pablo Baudin for providing me with data, while I was stressed out writing. I also thank the rest of the qLEAP group at Aarhus University for nice discussions, cake sessions and a *very* nice working environment.

I would like to acknowledge support from The European Research Council under the European Union's Seventh Framework Programme (FP/2007–2013)/ERC Grant Agreement No. 291371, and support from The Danish Council for Independent Research – Natural Sciences. This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725 provided through an INCITE allocation as well as the PRACE Research Infrastructure resource Curie at the Très Grand Centre de Calcul (TGCC) operated by CEA near Paris, France.

Contents

1 Introduction **3**
1.1 The electronic Schrödinger equation 4
1.2 Hartree–Fock theory 6
1.3 CC correlation methods 8

I The Coupled Cluster singles and doubles model **15**

2 Solving the CCSD equations **17**
2.1 Introduction 17
2.2 Solution of the nonlinear coupled cluster amplitude equations 18
2.3 The subspace CR method 19
2.4 The DIIS method 21
2.5 Implementation of DIIS and CROP 23
2.6 Numerical results 24
2.7 Conclusion 26

3 PDM tensors **27**
3.1 The tensor interface 28
3.2 Results 33
3.3 Conclusion and outlook 35

4 Massively parallel CCSD **37**
4.1 Introduction 37
4.2 Evaluation of the CCSD amplitude equations 38
4.3 Implementation 43
4.4 Numerical results 46
4.5 Conclusion and outlook 49

A Technical appendix: **51**
A.1 CCSD technical notes 51

5 Interaction energies **53**
5.1 Introduction 53
5.2 Summary of UC, CP and SNOOP 54
5.3 Computational details 57
5.4 Results 59
5.5 Conclusion and perspectives 64

II The Divide-Expand-Consolidate method revisited 65

6 DEC Locality Analysis 69
6.1 The DEC algorithm summarized . . . 70
6.2 Orbital spaces in DEC–CC fragment energy calculations . . . 73
6.3 The FOP algorithm . . . 85
6.4 Numerical illustrations . . . 90
6.5 Summary and outlook . . . 97

B Pair fragments and atomic extent 99
B.1 The single fragment extent . . . 99
B.2 Pair spaces as unions of spaces . . . 100

7 Parallelization of the DEC algorithm 103
7.1 Overview of the DEC scheme . . . 103
7.2 Parallelization of the DEC scheme . . . 104
7.3 Results . . . 107
7.4 Conclusion . . . 113

8 The DEC–PNO–CC approach 115
8.1 Errors and redundancies in a DEC–CC calculation . . . 115
8.2 Fragment reduction using fragment-adapted orbitals . . . 116
8.3 Results . . . 124
8.4 Conclusion and outlook . . . 126

9 Summary and outlook 129
9.1 English summary . . . 129
9.2 Dansk resumé . . . 130

III Attached articles 141

C Articles 143
C.1 Article 1 . . . 143
C.2 Article 2 . . . 152
C.3 Article 3 . . . 156
C.4 Manuscript for article 4 . . . 163
C.5 Manuscript for article 5 . . . 179

Chapter 1

Introduction

In natural sciences, the interplay between theory and experiment is one of the main driving forces for obtaining ever more profound insights into natural processes. In molecular and solid–state sciences, such as chemistry, physics, molecular–engineering, and molecular biology quantum chemical calculations have become a standard tool to interpret and/or supplement experimental data, to determine properties which are not possible (or difficult) to access experimentally, or even to replace experiments. The accuracy and predictive power of modern quantum chemical calculations have even developed to surpass certain experimental measurements, usually at a lower cost, making quantum theoretical calculations attractive for science and industry.

In order to calculate molecular energies and properties, two principal methods have established their usefulness, i.e. density–functional theory (DFT) and wave–function (WF) theory. Whilst both are in principle *exact*, in the sense that all properties are described exactly in the mathematical construction (the density or the WF), only WF theory provides a way to systematically approach the *exact* solution at a (usually) increasing computational cost, e.g. using the highly successful coupled cluster (CC) hierarchy of WF models. Because of the cost of the WF models, the DFT method has developed to become the method of choice, whenever the cost of the corresponding WF calculation is high, the WF calculation is utterly impossible, or if a qualitative result is sufficient. However, when highly accurate results are critical and the cost is acceptable, WF calculations are usually used.

In the recent years, an extension of the application range of quantum chemical calculations has emerged, both on the computational and theoretical levels. One of the prerequisites for performing precise WF calculations for chemical systems of interest are computers, which are able to process the immense amount of parameters occurring in such a calculation in a reasonable time. Besides the speed of individual processors, the increasing degree of parallel processing of modern supercomputers has proven to be exceptionally valuable when seeking a fast and precise answer using WF theory. Yet, the progress in computational engineering alone cannot remedy the high cost associated with a WF calculation, e.g., if a system of size N takes a day to be processed on a modern computer, increasing the system size by a factor 10 (assuming a $\mathcal{O}(N^6)$ scaling algorithm and Moore's Law [1]) it would take 34 years before a computer has been developed which can process the increased calculation in a single day, and even longer for larger systems (and/or more precise WF methods). Obviously, theoretical developments have to complement the advances in computing for large systems to be treated in the near future.

In order to calculate molecular energies and properties of interest for the aforementioned scientific disciplines, theories have been developed which are able to include the environment of a region described by the expensive quantum mechanical (QM) calculation in a cheaper

way, e.g. by coupling quantum and classical mechanics, a method which was awarded with the Nobel prize 2013 [2]. However, also for these methods one faces the problem that an accurate description can only be obtained if the QM region is large enough and treated at an appropriate level of theory. It is therefore necessary to develop highly accurate WF methods, which are applicable to systems of considerable size and, preferably, taking full advantage of the currently available computational resources. All the developments in this thesis will therefore be centered around pushing the limits for the application range for WF methods.

In Section 1.1 we give a brief introduction to WF theory based on second quantization. We will then explain how WF calculations traditionally start out with a Hartree–Fock calculation [3], and then being augmented with an additional correlation calculation. A summary of HF theory is given in Section 1.2. For augmenting the Hartree–Fock result we have chosen to use the coupled cluster hierarchy of methods [4–6], which will be briefly outlined in Section 1.3.

This thesis is divided into two parts. In Part I the focus will be on how a conventional model of the coupled cluster hierarchy may be implemented to be run on modern super computer clusters and Part II will be focused on how a linear–scaling coupled cluster algorithm with energy based error control may be realized. Specifically, in Chapter 2 we discuss how the nonlinear sets of equations, occurring in the WF models, may be solved in an optimal way. In Chapter 3 we will discuss a parallel distributed data structure which is used throughout all chapters of this thesis. In Chapter 4 we focus on how the time–determining part of a coupled cluster calculation may be implemented in an efficient way and in Chapter 5 we discuss an application of the implementation for calculating molecular interaction energies. Part II will start out with analyzing the equations occurring in a conventional formulation of the coupled cluster model under consideration in order to obtain a linear–scaling algorithm with rigorous error control in Chapter 6. In Chapter 7 the parallelization of the linear–scaling algorithm is discussed, and finally in Chapter 8 a possibility for reducing the overhead in the algorithm is explored.

1.1 The electronic Schrödinger equation and second quantization

The basis for all every WF theory is the well–known partial differential Schrödinger equation, introduced by Erwin Schrödinger in 1926 [7]

$$\hat{H}\Psi = E\Psi, \tag{1.1}$$

given here in the time–independent version. Ψ is the WF and E is the electronic energy when the Hamilton operator $\hat{H}$ is given in the Born-Oppenheimer (BO) approximation [8]. The BO Hamiltonian reads

$$\hat{H} = -\frac{1}{2}\sum_{i}^{N_\mathrm{e}} \nabla_i^2 - \sum_{i}^{N_\mathrm{e}}\sum_{I}^{N_n} \frac{Z_I}{|\mathbf{r}_i - \mathbf{r}_I|} + \sum_{i\leq j}^{N_\mathrm{e}} \frac{1}{|\mathbf{r}_i - \mathbf{r}_j|} + \sum_{I\leq J}^{N_n} \frac{Z_I Z_J}{|\mathbf{r}_I - \mathbf{r}_J|}, \tag{1.2}$$

where atomic units have been used. A summation over i/j (I/J) in the BO Hamiltonian runs over all electrons N_e (nuclei N_n) of the system, ∇ is the differentiation operator, $\mathbf{r}$ denotes a position, and Z is the atomic number of the respective nucleus.

Solving the Schrödinger equation results in the exact WF Ψ, which contains all the physical information about the system, but finding an exact solution is not possible except for the

Table 1.1: Overview over the indices used in all the Chapters of this thesis unless stated otherwise.

$a, b, c, d, \ldots$	virtual MO labels
$i, j, k, l, \ldots$	occupied MO labels
$p, q, r, s, \ldots$	labels for MOs of unspecified occupancy
$\alpha, \beta, \gamma, \delta, \ldots$	AO labels
$\mathfrak{p}, \mathfrak{q}, \mathfrak{r}, \mathfrak{s}, \ldots$	generic labels referring to an AO or an MO basis

simplest systems. Usually, solving the Schrödinger equation computationally for multi–electron systems involves the introduction of a set of one–electron WFs, which is used for the algebraization. The spacial one–electron WFs $\phi_p(\mathbf{r})$ are called molecular orbitals (MOs) and they are usually expanded in an underlying atomic orbital (AO) basis $\chi_\alpha(\mathbf{r})$ (see Table 1.1 for index conventions used in this thesis), where the MOs are expressed as

$$\phi_p(\mathbf{r}) = \sum_\alpha \chi_\alpha(\mathbf{r}) C_{\alpha p}, \tag{1.3}$$

where $C_{\alpha p}$ are the orbital coefficients collected as a matrix $\mathbf{C}$. In this thesis, we will only consider real orbitals. In non–relativistic theories, the electronic spin may be taken into account by multiplying the MO with one of the spin WFs $\alpha(m_s)$ or $\beta(m_s)$ (m_s is conventionally the projection of the electronic spin angular momentum on the z axis of the chosen coordinate system) to obtain a spin–orbital $\phi_p(\mathbf{r}, m_s)$. When using an orbital basis for expressing the approximate WF $\tilde{\Psi}$, it is convenient to work in terms of the second quantization formalism, which by construction is antisymmetric with respect to fermion exchange. In the second quantization formalism the BO Hamiltonian becomes [9]

$$\hat{H} = \sum_{pq} h_{pq} \hat{E}_{pq} + \sum_{pqrs} g_{pqrs} \hat{e}_{pqrs} + h_{\text{nuc}}, \tag{1.4}$$

where h_{pq} (g_{pqrs}) and $\hat{E}_{pq}$ ($\hat{e}_{pqrs}$) are the one (two) electron integrals and one (two) electron excitation operators in the MO basis. The one– and two–electron singlet excitation operators may be written as

$$\hat{E}_{pq} = a^\dagger_{p\rho} a_{q\rho} + a^\dagger_{p\sigma} a_{q\sigma}, \tag{1.5}$$

$$\hat{e}_{pqrs} = \hat{E}_{pq} \hat{E}_{rs} - \delta_{qr} \hat{E}_{ps}, \tag{1.6}$$

where $a^\dagger$ and a are creation and annihilation operators, respectively, ρ and σ refer to any of the two possible values of the spin functions $\alpha(m_s)$ and $\beta(m_s)$ and the rest of the indices refer to the MO basis according to Table 1.1. The one and two electron integrals are then given as

$$h_{pq} = \int \phi_p(\mathbf{r}, m_s) \left(-\frac{\nabla^2}{2} - \sum_I^{N_n} \frac{1}{|\mathbf{r} - \mathbf{r}_I|} \right) \phi_q(\mathbf{r}, m_a) \; d\mathbf{r} dm_s, \tag{1.7}$$

$$g_{pqrs} = \int \phi_p(\mathbf{r}_1, m_{s1}) \phi_r(\mathbf{r}_2, m_{s2}) \frac{1}{|\mathbf{r}_1 - \mathbf{r}_2|} \phi_q(\mathbf{r}_1, m_{s1}) \phi_q(\mathbf{r}_2, m_{s2}) \; d\mathbf{r}_1 dm_{s1} \; d\mathbf{r}_2 dm_{s2}, \tag{1.8}$$

$$h_{\text{nuc}} = \sum_{I \leq J}^{N_n} \frac{Z_I Z_J}{|r_I - r_J|}, \tag{1.9}$$

where we have used Mulliken notation for the two–electron integral in Eq. (1.8). If the one–electron basis $\{\phi_p\}$ is complete, we may express the *exact* WF of Eq. (1.1), as

$$|\Psi\rangle = \sum_{i}^{N_\mathrm{e}} \sum_{\mu_i} D_{\mu_i} \hat{\tau}_{\mu_i} |0\rangle, \tag{1.10}$$

where Dirac's Bra-Ket notation was used, $|0\rangle$ is a reference WF, D_{μ_i} are the *configuration interaction* (CI) coefficients and $\hat{\tau}_i$ is an excitation operator which excites i electrons in the reference WF $|0\rangle$ to obtain a specific electronic configuration, e.g. for $i = 1, 2, 3$

$$\hat{\tau}_i^a = \hat{E}_i^a, \tag{1.11}$$
$$\hat{\tau}_{ij}^{ab} = \hat{E}_i^a \hat{E}_j^b, \tag{1.12}$$
$$\hat{\tau}_{ijk}^{abc} = \hat{E}_i^a \hat{E}_j^b \hat{E}_k^b, \tag{1.13}$$
$$\vdots$$

Note that there are infinitely many possible configurations if a complete one electron basis is used. In general, a complete basis is computationally intractable, ans only approximations to the correct WF Ψ can be calculated. In practice, the one–electron basis will be truncated and the number of configurations entering Eq. (1.10) will be limited. The *exact* WF in a given truncated basis is called the *full CI* WF $|\mathrm{FCI}\rangle$. From now on we will only be concerned with truncated one–electron bases unless stated otherwise.

The reference WF for the CI expansion $|0\rangle$ in Eq. (1.10) is usually obtained by a Hartree–Fock (HF) calculation, which will be discussed briefly in Section 1.2. Obtaining the full CI WF is expensive due to a factorial scaling of the number of parameters with the number of one electron WFs in the basis set and with the number of electrons in the system. Therefore, truncated CI models have been proposed, where the configuration space is truncated at a given excitation level, giving rise to the CI hierarchy of models — CI with singles and doubles (CISD), CISD with triples (CISDT), ... — which systematically approach the full CI WF [10]. This hierarchy of models has the feature that all ground state energies calculated will be an upper bound to the exact ground state energy for the system. This feature is conventionally referred to as the *variation theorem* and the CI energies are therefore *variational*. However, the CI hierarchy of models (except for the full CI WF) is not size–extensive [9], i.e. the total WF for a combined system is not multiplicatively separable into WFs for the individual systems, and more importantly the energy determined for two non–interacting systems in a single calculation will not equal the sum of the energies for two individual calculations on the subsystems. For this reason, the CI hierarchy been superseded by the size–extensive coupled cluster (CC) hierarchy of models which we will discuss in Section 1.3. Note that, in all the following chapters we will only consider closed–shell WFs.

1.2 Hartree–Fock theory

For a closed–shell molecule the reference WF $|0\rangle$ is well approximated by a single Slater determinant $|\mathrm{cs}\rangle$ and the optimization of the HF WF may be parameterized as [9],

$$|\mathrm{HF}(\boldsymbol{\kappa})\rangle = \exp(\hat{\kappa})|\mathrm{cs}\rangle, \tag{1.14}$$

where $\hat{\kappa}$ is an anti-Hermitian one-electron operator,

$$\hat{\kappa} = \sum_{ai} \kappa_{ia}(\hat{E}_{ia} - \hat{E}_{ai}). \tag{1.15}$$

In Eq. (1.15), κ_{ia} is an element of the associated anti-Hermitian $\boldsymbol{\kappa}$ rotation matrix, and $\hat{E}_{ai}$ ($\hat{E}_{ia}$) is a singles excitation (de–excitation) operator. In the HF approach the orbitals constituting $|\text{cs}\rangle$ are determined variationally, i.e. the orbital coefficients $\mathbf{C}$ in Eq. (1.3) are optimized by minimizing the HF energy. The optimization condition for the HF state then becomes

$$\langle \text{HF}(\boldsymbol{\kappa})|[E_{ai}, \hat{H}]|\text{HF}(\boldsymbol{\kappa})\rangle = 0, \tag{1.16}$$

also known as Brillouin-Theorem. When this condition is fulfilled, $|\text{HF}(\boldsymbol{\kappa})\rangle$ becomes the optimized HF state $|\text{HF}\rangle$. Furthermore, this condition is equivalent to requiring that the occupied-virtual block of the Fock matrix $\mathbf{F}$ is zero. A Fock matrix element may be written as

$$F_{pq} = h_{pq} + \sum_{i} (2g_{pqii} - g_{piiq}). \tag{1.17}$$

Finally, the closed–shell HF energy may be written as

$$E_{\text{HF}} = \langle \text{HF}|\hat{H}|\text{HF}\rangle = 2\sum_{i} h_{ii} + \sum_{ij} (2g_{iijj} - g_{ijji}) + h_{\text{nuc}}. \tag{1.18}$$

The HF approach is a mean–field approach in the sense that an electron moves independent of the other electrons in the field generated by the nuclei and the other electrons while obeying Pauli's principle, i.e. the N_{e}-electron WF is antisymmetric upon the exchange of spin and position of any two electrons. However, the electronic motion in molecules is correlated in two ways. *Static correlation* arises in situations of near degeneracy of orbitals and requires a multi–determinantal description of the ground state (unless the goal is to arrive at a full CI WF). Since we work in the regime of closed–shell molecules in this thesis, the ground state will be dominated by a single determinant and thus we are only concerned with describing *dynamical correlation* effects. Electrons are dynamically correlated in the sense that minor fluctuations in the electron density at a certain place in the molecule will have an effect on the neighboring regions. These fluctuations may be categorized as coulomb–hole (electrons avoiding each other because of the mutual repulsion), and dispersion effects. The mean field Fock operator ($\hat{f}$) is not capable of describing these two–electron effects and we therefore introduce the *fluctuation potential* as the difference between the exact two–electron part of the Hamilton operator ($\hat{g}$) and the two electron part of the Fock operator ($\hat{V}$) as

$$\hat{\Phi} = \hat{g} - \hat{V}. \tag{1.19}$$

The exact Hamiltonian may then be expressed as

$$\hat{H} = \hat{f} + \hat{\Phi} + h_{\text{nuc}}. \tag{1.20}$$

Describing the electron correlation effects, e.g. with the full CI WF $|\text{FCI}\rangle$, will result in a total energy that is lower than the energy captured by the HF approach. The difference between the FCI energy (E_{FCI}) and the HF energy is denoted the exact correlation energy ($E_{\text{corr}}^{\text{exact}}$)

$$E_{\text{corr}}^{\text{exact}} = E_{\text{FCI}} - E_{\text{HF}}. \tag{1.21}$$

1.3 CC correlation methods

Since the FCI WF is practically intractable for chemical systems of interest, approximations, which should most desirably be hierarchical, have to be introduced. One way of approaching the FCI WF in a systematic manner was mentioned before, the CI hierarchy of methods. It was briefly discussed that the CI hierarchy is not size–extensive and may therefore not give a physically correct description of large molecular systems. Here we will focus on the size–extensive coupled cluster (CC) approach and establish the CC hierarchy of methods — second order Møller–Plesset theory (MP2) [11], CC with singles and doubles excitations (CCSD) [12], CCSD with a perturbative triples correction (CCSD(T)) [13], CCSD with triples (CCSDT) [14] ... where the focus is on the first three models of the hierarchy. We will briefly give the explicit equations for each of these models as these will form the foundation for the rest of the thesis.

1.3.1 The coupled cluster parametrization of the WF

In coupled cluster theory the WF is parametrized in terms of excitation operators

$$\hat{T} = \sum_{i}^{N_\mathrm{e}} \sum_{\mu_i} \hat{T}_{\mu_i}, \tag{1.22}$$

where μ_i is an excitation manifold of excitations i ranging from 1 to the number of electrons N_e in the system. For $i = 1, 2, 3 \ldots$ the operators $\hat{T}_{\mu_i} = t_{\mu_i} \hat{\tau}_{\mu_i}$ read

$$\hat{T}_1 = t_i^a \hat{\tau}_i^a, \tag{1.23}$$

$$\hat{T}_2 = \frac{1}{2!} t_{ij}^{ab} \hat{\tau}_{ij}^{ab}, \tag{1.24}$$

$$\hat{T}_3 = \frac{1}{3!} t_{ijk}^{abc} \hat{\tau}_{ijk}^{abc}, \tag{1.25}$$

$$\vdots$$

where the amplitudes t_{μ_i} may be interpreted as the probability for a certain excitation. The CC WF may be written in an exponential form

$$|\mathrm{CC}\rangle = \exp(\hat{T})|\mathrm{HF}\rangle \tag{1.26}$$

and the coupled cluster Schrödinger equation becomes

$$\hat{H} \exp(\hat{T})|\mathrm{HF}\rangle = E_\mathrm{CC} \exp(\hat{T})|\mathrm{HF}\rangle. \tag{1.27}$$

We may introduce the excitation manifolds

$$\langle \mathrm{HF}|\hat{\tau}_{\mu_i}^\dagger = \langle \mu_i|, \tag{1.28}$$

to obtain the CC energy and linked CC equations by projecting Eq. (1.27) with the reference state $\langle \mathrm{HF}|$ and the excitation manifolds $\langle \mu_i|$ from Eq. (1.28), respectively,

$$\langle \mathrm{HF}| \exp(-\hat{T}) \hat{H} \exp(\hat{T})|\mathrm{HF}\rangle = E_\mathrm{CC}, \tag{1.29}$$

$$\langle \mu_i| \exp(-\hat{T}) \hat{H} \exp(\hat{T})|\mathrm{HF}\rangle = 0. \tag{1.30}$$

Since the $|\mathrm{CC}\rangle$ state is expanded exponentially in the parameters t_{μ_i} it is more convenient to solve for the cluster parameters by projection, than by variationally optimizing the energy. However, when using the projection technique, E_{CC} will not be variational in the parameters t_{μ_i}. On the other hand, the exponential expansions in Eqs. (1.29) and (1.30) terminate after a few terms. Thus, we may write the CC energy as

$$\begin{aligned} E_{\mathrm{CC}} &= \langle \mathrm{HF}|\hat{H}(1+\hat{T}+\frac{1}{2}\hat{T}^2+\dots)|\mathrm{HF}\rangle \\ &= E_{\mathrm{HF}} + E_{\mathrm{corr}}, \end{aligned} \tag{1.31}$$

$$E_{\mathrm{corr}} = \langle \mathrm{HF}|\hat{H}(\hat{T}+\frac{1}{2}\hat{T}^2)|\mathrm{HF}\rangle, \tag{1.32}$$

where E_{corr} is the correlation energy recovered with a given CC model. Since the Hamilton operator is a two–electron operator and due to Brillouin's Theorem in Eq. (1.16) we may write the correlation energy E_{corr} from Eq. (1.32) for any CC model as

$$E_{\mathrm{corr}} = \sum_{iajb} \tau_{ij}^{ab} L_{iajb}, \tag{1.33}$$

where

$$\tau_{ij}^{ab} = t_{ij}^{ab} + t_i^a t_j^b, \tag{1.34}$$

and

$$L_{iajb} = 2g_{iajb} - g_{ibja}. \tag{1.35}$$

A given CC model is obtained by truncating the cluster operator in Eq. (1.22) at a given order of excitations, e.g., for the CCSD model the cluster operator includes singles and doubles excitations (i.e. $i = 1, 2$), for CCSDT additionally triples excitations (i.e. $i = 1, 2, 3$) etc. Note that only doubles amplitudes appear for any CC model, which may include higher excitations than doubles. The doubles amplitudes are affected by the higher excitations through the coupled amplitude equations in Eq. (1.30) giving rise to a different correlation energy depending on the CC model.

1.3.2 The MP2 model

To arrive at the MP2 model, a perturbation expansion of the CC energy in Eq. (1.27) in orders of the fluctuation potential $\hat{\Phi}$ is truncated at first order [9]. This removes any than doubles amplitudes from Eqs. (1.29) and (1.30). Thus, the correlation energy for the MP2 model may be written as [9]

$$E_{\mathrm{corr}} = \sum_{ijab} t_{ij}^{ab} L_{iajb}, \tag{1.36}$$

and the amplitude equations in Eq 1.30 are simplified to become the MP2 amplitude equation [9],

$$g_{aibj} + \sum_c (t_{ij}^{cb} F_{ac} + t_{ij}^{ac} F_{bc}) - \sum_k (t_{kj}^{ab} F_{ki} + t_{ik}^{ab} F_{kj}) = 0. \tag{1.37}$$

These equations are valid for any optimized HF molecular orbital basis. A conventional MP2 calculation employs the canonical HF basis where the Fock matrix is diagonal,

$$F_{pq} = \delta_{pq}\varepsilon_p, \tag{1.38}$$

where ε_p is a HF orbital energy. In the canonical basis the amplitude equations simplify to,

$$t_{ij}^{ab} = -\frac{g_{aibj}}{\varepsilon_a + \varepsilon_b - \varepsilon_i - \varepsilon_j}. \tag{1.39}$$

In the canonical basis the solution of the linear equation in Eq. (1.37) is thus obtained directly from the integrals without resorting to an iterative scheme.

1.3.3 The CC singles and doubles model: CCSD vector equations expressed as effective CCD equations

To obtain the CCSD ansatz, the cluster operator in Eq. (1.22) includes singles and doubles excitations ($i = 1, 2$). Using the T_1 transformed formalism [15], the CCSD energy in Eq. (1.29) and amplitude equations in Eq. (1.30) may be written as

$$\langle \mathrm{HF}|e^{-\hat{T}_2}\hat{H}^{T_1}e^{\hat{T}_2}|\mathrm{HF}\rangle = E_{\mathrm{corr}}, \tag{1.40}$$

$$\langle {}_i^{\bar{a}}|e^{-\hat{T}_2}\hat{H}^{T_1}e^{\hat{T}_2}|\mathrm{HF}\rangle = 0, \tag{1.41}$$

$$\langle {}_{ij}^{\bar{a}\bar{b}}|e^{-\hat{T}_2}\hat{H}^{T_1}e^{\hat{T}_2}|\mathrm{HF}\rangle = 0, \tag{1.42}$$

We have used the notation of Ref. 15 where $\langle {}_i^{\bar{a}}|$ and $\langle {}_{ij}^{\bar{a}\bar{b}}|$ are shorthand notations for the singles and doubles excitation manifolds in the biorthonormal representation. The label specification used for the various orbital basis can be found in Table 1.1. $\hat{H}^{T_1}$ is the T_1 transformed Hamiltonian

$$\begin{aligned}\hat{H}^{T_1} &= e^{-\hat{T}_1}\hat{H}e^{\hat{T}_1} \\ &= \sum_{pq}\tilde{h}_{pq}E_{pq} + \frac{1}{2}\sum_{pqrs}\tilde{g}_{pqrs}e_{pqrs} + h_{\mathrm{nuc}},\end{aligned} \tag{1.43}$$

which contains the T_1 transformed integrals $\tilde{g}_{pqrs}$, $\tilde{h}_{pq}$. The T_1 transformed one– and two–electron MO integrals are obtained by a transformation from a generic basis

$$\tilde{h}_{pq} = \sum_{\mathfrak{tu}}\Lambda^p_{\mathfrak{t}p}\Lambda^h_{\mathfrak{u}q}h_{\mathfrak{tu}}, \tag{1.44}$$

$$\tilde{g}_{pqrs} = \sum_{\mathfrak{tuvw}}\Lambda^p_{\mathfrak{t}p}\Lambda^h_{\mathfrak{u}q}\Lambda^p_{\mathfrak{v}r}\Lambda^h_{\mathfrak{w}s}g_{\mathfrak{tuvw}}. \tag{1.45}$$

When the generic basis refers to the MO basis, $\mathbf{\Lambda}^p$ and $\mathbf{\Lambda}^q$ are given as

$$\mathbf{\Lambda}^p = (\mathbf{1} - \mathbf{t}_1^{\mathrm{T}}), \tag{1.46}$$

$$\mathbf{\Lambda}^h = (\mathbf{1} + \mathbf{t}_1). \tag{1.47}$$

When the generic basis refers to the AO basis, $\mathbf{\Lambda}^p$ and $\mathbf{\Lambda}^q$ become

$$\mathbf{\Lambda}^p = \mathbf{C}(\mathbf{1} - \mathbf{t}_1^{\mathrm{T}}), \tag{1.48}$$

$$\mathbf{\Lambda}^h = \mathbf{C}(\mathbf{1} + \mathbf{t}_1). \tag{1.49}$$

Table 1.2: The CCSD residual equations using T_1 transformed integrals as given in Ref. 9.

Contributions to the singles equations:

(A1)	$\Omega_{ai}^{A1} = \sum_{cdk} u_{ki}^{cd} \tilde{g}_{adkc}$
(B1)	$\Omega_{ai}^{B1} = -\sum_{ckl} u_{kl}^{ac} \tilde{g}_{kilc}$
(C1)	$\Omega_{ai}^{C1} = \sum_{ck} u_{ik}^{ac} \, {}^{I}\tilde{F}_{kc}$
(D1)	$\Omega_{ai}^{D1} = {}^{I}\tilde{F}_{ai}$

Contributions to the doubles equations:

(A2)	$\Omega_{aibj}^{A2} = \tilde{g}_{aibj} + \sum_{cd} t_{ij}^{cd} \tilde{g}_{acbd}$
(B2)	$\Omega_{aibj}^{B2} = \sum_{kl} t_{kl}^{ab} \left(\tilde{g}_{kilj} + \sum_{cd} t_{ij}^{cd} \tilde{g}_{kcld} \right)$
(C2)	$\Omega_{aibj}^{C2} = -\left(1 + \frac{1}{2}\hat{P}_{ij}\right) \sum_{ck} t_{ki}^{bc} \left(\tilde{g}_{kjac} - \frac{1}{2} \sum_{dl} t_{lj}^{ad} \tilde{g}_{kdlc} \right)$
(D2)	$\Omega_{aibj}^{D2} = \frac{1}{2} \sum_{ck} u_{jk}^{bc} \left(\tilde{L}_{aikc} + \frac{1}{2} \sum_{dl} u_{il}^{ad} \tilde{L}_{ldkc} \right)$
(E2)	$\Omega_{aibj}^{E2} = \sum_{c} t_{ij}^{ac} \left({}^{I}\tilde{F}_{bc} - \sum_{dkl} u_{kl}^{bd} \tilde{g}_{ldkc} \right) - \sum_{k} t_{ik}^{ab} \left({}^{I}\tilde{F}_{kj} - \sum_{cdl} u_{lj}^{cd} \tilde{g}_{kdlc} \right)$

Notation and symmetries of intermediates:

(A3)	$\hat{P}_{ij} X_{aibj} = X_{ajbi}$
(B3)	$u_{ij}^{ab} = 2t_{ij}^{ab} - t_{ij}^{ba}$
(C3)	$\tilde{L}_{pqrs} = 2\tilde{g}_{pqrs} - \tilde{g}_{psrq}$
(D3)	${}^{I}\tilde{F}_{pq} = \sum_{\mu\nu} \Lambda_{\mu p}^{p} \Lambda_{\nu q}^{h} \, {}^{I}F_{\mu\nu}^{\mathrm{AO}}$
(E3)	${}^{I}F_{\mu\nu}^{\mathrm{AO}} = 2h_{\mu\nu} + \sum_{\sigma\lambda} \tilde{D}_{\sigma\lambda} (2g_{\mu\nu\sigma\lambda} - g_{\mu\lambda\sigma\nu})$
(F3)	$\tilde{D}_{\sigma\lambda} = \sum_{i} \Lambda_{\sigma i}^{p} \Lambda_{\lambda i}^{h}$
(G3)	$t_{ij}^{ab} = t_{ji}^{ba}$
(H3)	$\Omega_{aibj} = \Omega_{bjai}$

The $\mathbf{t}_1$ matrix, which refers to the singles amplitudes in the MO basis, is given as

$$[\mathbf{t}_1]_{pq} = \begin{Bmatrix} t_q^p & \text{if } p \text{ virtual and } q \text{ occupied} \\ 0 & \text{otherwise} \end{Bmatrix}. \tag{1.50}$$

Note that the T_1 transformed integrals have lower symmetry than the conventional MO integrals and only the symmetry with particle exchange $\tilde{g}_{pqrs} = \tilde{g}_{rspq}$ remains.

The CCSD amplitude equations in Eqs. (1.41) and (1.42) constitute a set of nonlinear equations that may be solved using an iterative algorithm, e.g. the DIIS or CROP algorithm [16–18], as will be detailed in Chapter 2. In each iteration the vector function

$$\Omega_{ai} = \langle {}^{\bar{a}}_{i} | e^{-\hat{T}_2} \hat{H}^{T_1} e^{\hat{T}_2} | \mathrm{HF} \rangle, \tag{1.51}$$

$$\Omega_{aibj} = \langle {}^{\bar{a}\bar{b}}_{ij} | e^{-\hat{T}_2} \hat{H}^{T_1} e^{\hat{T}_2} | \mathrm{HF} \rangle, \tag{1.52}$$

is evaluated for the trial solution and the DIIS or CROP algorithm is then used to determine an improved trial solution. The time dominating step in an iteration sequence is the evaluation of the vector function in Eqs. (1.51) and (1.52) for a trial solution.

Using the T_1 transformed formalism and a biorthonormal representation of the singles and doubles projection manifolds the CCSD doubles amplitude equations in Eqs. (1.51) and (1.52) may be written as

$$\Omega_{ai} = \Omega_{ai}^{A1} + \Omega_{ai}^{B1} + \Omega_{ai}^{C1} + \Omega_{ai}^{D1}, \tag{1.53}$$

$$\Omega_{aibj} = P_{ij}^{ab}(\frac{1}{2}\Omega_{aibj}^{A2} + \frac{1}{2}\Omega_{aibj}^{B2} + \Omega_{aibj}^{C2} + \Omega_{aibj}^{D2} + \Omega_{aibj}^{E2}), \tag{1.54}$$

where we have used the permutation operator $\hat{P}_{ij}^{ab}\Omega_{aibj} = \Omega_{aibj} + \Omega_{bjai}$. The individual contributions for Eqs. (1.53) and (1.54) are given in Table 1.2. The number of terms entering Eqs. (1.53) and (1.54) is considerably reduced compared to a formulation of the CCSD amplitude equations, which does not use the T_1 formalism, at the cost of reduced symmetry in the integrals. We describe in Chapter 4 how the individual terms in Eqs. (1.53) and (1.54) may be evaluated in a parallel computing environment without the use of local disks.

1.3.4 The CCSD(T) model

When moving up the CC hierarchy in the models defined by the cluster operator in Eq. (1.22), the subsequent step after CCSD would be CCSDT. CCSDT, however, scales as $\mathcal{O}(N^8)$ with the system size and is therefore immensely expensive, especially recalling that Eq 1.30 is solved iteratively. The method that has proven itself the "*golden standard*" of quantum chemistry [9] as an approximation to the CCSDT model is called CCSD(T), where the effect of triples excitations is included as a perturbative correction to the CCSD. The (T) correction to the CCSD energy may be calculated without the need for an iterative scheme, which in combination with its scaling behavior, i.e. $\mathcal{O}(N^7)$, and a good description of triples effects, have given the CCSD(T) method its reputation. The CCSD(T) energy is defined as [19]

$$E_{\mathrm{CCSD(T)}} = E_{\mathrm{HF}} + E_{\mathrm{CCSD}} + E_{(\mathrm{T})}, \tag{1.55}$$

where $E_{(\mathrm{T})}$ is the so-called (T) perturbative correction to the CCSD energy for the effect of triple excitations. The structure of $E_{(\mathrm{T})}$ is defined by two energy contributions rationalized from many-body perturbation theory [20] (MBPT): $E^{[4]}$, a fourth-order term involving CCSD doubles amplitudes, and $E^{[5]}$, a fifth-order term involving CCSD singles amplitudes

$$E_{(\mathrm{T})} = E^{[4]} + E^{[5]}. \tag{1.56}$$

In a basis of canonical HF spatial orbitals, the expressions for the fourth- and fifth-order contributions may be derived from MBPT as [21]

$$E^{[4]} = \sum_{abc}\sum_{ijk} \tilde{t}_{ijk}^{abc} t_{ijk}^{abc} \varepsilon_{ijk}^{abc}, \tag{1.57a}$$

$$E^{[5]} = \sum_{abc}\sum_{ijk} \tilde{z}_{ijk}^{abc} t_{ijk}^{abc} \varepsilon_{ijk}^{abc}. \tag{1.57b}$$

In Eq. (1.57), the triples amplitudes $\{t_{ijk}^{abc}\}$ are given as

$$t_{ijk}^{abc} = -P_{ijk}^{abc} \frac{\sum_d t_{ij}^{ad} g_{ckbd} - \sum_l t_{il}^{ab} g_{cklj}}{\varepsilon_{ijk}^{abc}}, \tag{1.58}$$

where t_{ij}^{ab} are CCSD doubles amplitudes, ε_{ijk}^{abc} denotes the orbital energy difference between the virtual and occupied orbitals of the excitation

$$\varepsilon_{ijk}^{abc} = \varepsilon_a + \varepsilon_b + \varepsilon_c - (\varepsilon_i + \varepsilon_j + \varepsilon_k), \tag{1.59}$$

and P_{ijk}^{abc} is a symmetrization operator

$$P_{ijk}^{abc} x_{ijk}^{abc} = x_{ijk}^{abc} + x_{ikj}^{acb} + x_{jik}^{bac} + x_{jki}^{bca} + x_{kij}^{cab} + x_{kji}^{cba}. \tag{1.60}$$

The z_{ijk}^{abc} coefficients in Eq. (1.57b) are given as

$$z_{ijk}^{abc} = \frac{-(t_i^a g_{jbkc} + t_j^b g_{iakc} + t_k^c g_{iajb})}{\varepsilon_{ijk}^{abc}}, \tag{1.61}$$

where t_i^a are the CCSD singles amplitudes, and an arbitrary six-index quantity $\tilde{x}_{ijk}^{abc}$ in Eq. (1.57) is defined as

$$\tilde{x}_{ijk}^{abc} = 2(\tfrac{2}{3} x_{ijk}^{abc} - x_{ijk}^{acb} + \tfrac{1}{3} x_{ijk}^{bca}). \tag{1.62}$$

The expressions for the fourth- and fifth-order contributions in Eq. (1.57) both contain orbital energy differences, and the evaluation of the (T) energy correction by this conventional formulation will thus be restricted to a basis of canonical HF orbitals.

Alternatively, the two correction terms in Eq. (1.56) may be derived from Lagrangian-based CC perturbation theory for a HF reference state as in Refs. 9 and 22

$$E^{[4]} = 2 \sum_{ab} \sum_{ij} (2t_{ij}^{ab} - t_{ji}^{ab}) T_{ij}^{ab}, \tag{1.63a}$$

$$E^{[5]} = 2 \sum_{a} \sum_{i} t_i^a T_i^a. \tag{1.63b}$$

In Eq. (1.63), the fourth- and fifth-order contributions to the (T) correction are formulated in terms of the T_{ij}^{ab} and T_j^a intermediates, respectively

$$T_{ij}^{ab} = \sum_{cd} \sum_{k} \left(t_{ijk}^{acd} L_{bckd} - t_{kji}^{acd} g_{kdbc} \right) - \sum_{c} \sum_{kl} \left(t_{ikl}^{abc} L_{kjlc} - t_{lki}^{abc} g_{kjlc} \right), \tag{1.64a}$$

$$T_i^a = \sum_{cd} \sum_{kl} \left(t_{ikl}^{acd} - t_{lki}^{acd} \right) L_{kcld}. \tag{1.64b}$$

These expressions are valid in any basis, but for a non–canonical basis the t_{ijk}^{abc} amplitudes may not be obtained by Eq. (1.58) anymore. Instead, it will be necessary to solve the amplitude equations

$$\sum_{e} (t_{ijk}^{ebc} F_{ae} + t_{ijk}^{aec} F_{be} + t_{ijk}^{abe} F_{ce}) - \sum_{m} (t_{mjk}^{abc} F_{mi} + t_{imk}^{abc} F_{mj} + t_{ijm}^{abc} F_{mk}) = -P_{ijk}^{abc} (\sum_{d} t_{ij}^{ad} g_{ckbd} - \sum_{l} t_{il}^{ab} g_{cklj}), \tag{1.65}$$

iteratively, to obtain the amplitudes.

Comparing the evaluation of $E_{(\mathrm{T})}$ from Eq. (1.57) and from Eqs. (1.63) and (1.64), we note that both expressions contain the triples amplitudes of Eq. (1.58), the generation of which scales as O^3V^4 (where O and V denote the number of occupied and virtual orbitals, respectively). In Eq. (1.63a), the evaluation of the T_{ij}^{ab} intermediates in Eq. (1.64a) contains an additional O^3V^4 scaling term, while the conventional expression in Eq. (1.57) contains no such term since each triples amplitude is immediately contracted with either itself or a z coefficient in an O^3V^3 step. Thus if no additional cost–reductions are performed, it is more expensive to evaluate the (T) contribution in a general basis, but it is important for the developments in Part II that the (T) energy may be expressed in a non–canonical basis.

Now we have discussed the foundations this thesis rests upon and we will begin describing the individual research projects of this thesis beginning with Part I, the massively parallel implementation of the CCSD algorithm.

Part I

The Coupled Cluster singles and doubles model

Chapter 2

Solving the CCSD equations

This chapter summarizes the work of article 2

2.1 Introduction

The coupled cluster amplitude equations in Eq. (1.30) constitute a set of nonlinear equations that in most electronic structure programs are solved using the DIIS (direct inversion of the iterative subspace) algorithm [16]. For solving the coupled pair and cluster amplitude equations the DIIS algorithm was introduced by Ahlrichs et al. [23] and Scuseria et al. [17], respectively. In the DIIS algorithm trial vectors and corresponding vector function values (error vectors, residual vectors) from the previous iterations are stored and used to obtain an improved trial vector of the next iteration. The storage requirement of these trial vectors may become a bottleneck in the DIIS algorithm. To circumvent this bottleneck, the trial vectors of the initial iterations have been discarded, storing only the trial vectors of the last iterations. However, this may lead to a serious abatement in the convergence and even divergence when the DIIS algorithm is used.

When solving the CC amplitude equations, an optimal way of discarding information from the previous iterations of a DIIS sequence is discussed, leading to the conjugate residual with optimal trial vectors (CROP) sequence in which the trial vectors are stored in an optimal way with respect to the linear part of the vector function, thus avoiding the abatement in the convergence rate upon discarding trial vectors, present in the DIIS sequence. If no trial vectors are discarded the CROP sequence and the DIIS sequence become identical. When CC theory is applied for a state dominated by a single configuration, the CC equations have a weak nonlinearity and in practice solving these only requires information from the last three trial vectors. However also for a more pronounced nonlinearity, it is sufficient to keep the information from the last three trial vectors, as will be demonstrated.

When solving a set of linear equations $\mathbf{A}\mathbf{x}+\mathbf{b}=0$, two algorithms – the conjugate gradient (CG) and the conjugate residual (CR) algorithms – share the common features that information from the last three iterations are sufficient to maintain the convergence of the algorithm. In the CG method this unique feature is obtained as a result of minimizing the quadratic function $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} + \mathbf{x}^T\mathbf{b}$ while for the CR algorithm it is obtained as a result of minimizing the residual norm $\mathbf{r}^T\mathbf{r}$ (with $\mathbf{r} = \mathbf{A}\mathbf{x} + \mathbf{b}$). For the CG algorithm, it is the conjugacy of the directions with respect to the matrix $\mathbf{A}$ that makes the subspace truncations possible, while for the CR algorithm, it is the conjugacy with respect to the matrix $\mathbf{A}^T\mathbf{A}$ and a generalization of the residual concept to the regime of nonlinear systems of equations is straightforward.

The CG algorithm was introduced into quantum chemistry more than 30 years ago by Wormer, Visser and Paldus [24]. Van Lenthe and Pulay have demonstrated that a CI eigenvalue problem could be solved with the CG algorithm using only three subspace vectors because the eigenvalue is quadratic in the coefficients [25]. For solving sets of nonlinear equations Ziolkowski et al. [18] introduced the CROP algorithm as a subspace generalization of the CR algorithm and showed the connection between the CROP and DIIS algorithms.

To describe the simplified route for deriving the CROP algorithm, the properties of the CR subspace method are summarized. Then consider a standard DIIS implementation is considered, highlighting the connection to the CR subspace method, and describe how the DIIS algorithm may be changed to a CR subspace implementation, and thereby turned into the CROP algorithm. Finally, numerical results are presented for solving the nonlinear CCSD equations, demonstrating the advantage of using the CROP algorithm compared to the DIIS algorithm.

2.2 Solution of the nonlinear coupled cluster amplitude equations

The CC amplitude Eq. (1.30) constitute a set of nonlinear equations in all (singles, doubles, ...) cluster parameters $\mathbf{t}$

$$\langle \boldsymbol{\mu_i}| \exp(-\hat{T})\hat{H}\exp(\hat{T})|\mathrm{HF}\rangle = \mathbf{0}, \tag{2.1}$$

where $|\boldsymbol{\mu_i}\rangle$ are the excitation manifolds, written as a vector of the same dimension as the sum of the dimensions of the individual manifolds. Interpreting this set of equations as a general vector function

$$\mathbf{w}(\mathbf{t}) = \langle \boldsymbol{\mu_i}| \exp(-\hat{T})H\exp(\hat{T})|\mathrm{HF}\rangle, \tag{2.2}$$

our task is to determine the solution vector $\mathbf{t}^*$ (the amplitudes), which satisfies

$$\mathbf{w}(\mathbf{t}^*) = \mathbf{0}, \tag{2.3}$$

employing an iterative algorithm.

In iteration n of an algorithm for solving Eq. (2.3) we assume that a trial solution $\mathbf{t}^{(n)}$ is known and try to find an update $\Delta\mathbf{t}^{(n)}$ that gives an improved trial solution of iteration $n+1$

$$\mathbf{t}^{(n+1)} = \mathbf{t}^{(n)} + \Delta\mathbf{t}^{(n)}. \tag{2.4}$$

Expanding the vector function $\mathbf{w}(\mathbf{t})$ around $\mathbf{t}^{(n)}$

$$\mathbf{w}(\mathbf{t}) = \mathbf{w}^{(n)} + \mathbf{W}^{(n)}\Delta\mathbf{t} + \mathcal{O}(\|\Delta\mathbf{t}^2\|), \tag{2.5}$$

where $\mathbf{w}^{(n)}$ and $\mathbf{W}^{(n)}$ are the vector function and the Jacobian at $\mathbf{t}^{(n)}$, respectively, and

$$\Delta\mathbf{t} = \mathbf{t} - \mathbf{t}^{(n)}, \tag{2.6}$$

we may determine an iteration function for the update $\Delta\mathbf{t}^{(n)}$ by setting Eq. (2.5) equal to zero and neglecting the second order terms $\mathcal{O}(\|\Delta\mathbf{t}^2\|)$,

$$\mathbf{w}^{(n)} + \mathbf{W}^{(n)}\Delta\mathbf{t} = \mathbf{0}. \tag{2.7}$$

The accuracy of the solution vector $\mathbf{t}$ for the set of linear equations Eq. (2.7) may be measured by the residual

$$\mathbf{r}^{(n)}(\mathbf{t}) = \mathbf{w}^{(n)} + \mathbf{W}^{(n)}(\mathbf{t} - \mathbf{t}^{(n)}). \tag{2.8}$$

When the linear equations in Eq. (2.7) are solved and a $\mathbf{t}^{(n+1)}$ is determined such that

$$\|\mathbf{r}^{(n)}(\mathbf{t}^{(n+1)})\|^2 = 0, \tag{2.9}$$

the iterative procedure defined by Eq. (2.4) will be quadratically convergent.

The linear equations Eq. (2.7) are usually solved using an iterative procedure. Standard iterative algorithms, as the conjugate gradient (CG) and conjugate residual (CR) methods [26] require that a linear transformation of the Jacobian on a trial vector $\mathbf{W}^{(n)}\mathbf{t}_{\text{trial}}$, is calculated in each iteration. Since such a linear transformation is as costly as the evaluation of the vector function $\mathbf{w}(\mathbf{t}_{\text{trial}})$ and since a large number of iterations are required to obtain a solution vector to Eq. (2.7) that satisfies (2.9) it is in practice too costly to use a quadratically convergent algorithm for obtaining $\mathbf{t}^*$.

In the next section we summarize the CR subspace algorithm for solving the linear equations in Eq. (2.7). In the subsequent section we describe a conventional DIIS implementation and how it may be viewed in the context of the iterative scheme developed in this section. This then allows to show how the DIIS algorithm may be modified to become a CR subspace algorithm that in turn identifies the CROP algorithm.

2.3 The subspace CR method

We now summarize the CR subspace algorithm for solving the set of linear equations in Eq. (2.7). We assume that we have carried out p iterations of a CR subspace algorithm and we thus know the trial solution $\mathbf{t}_{[p]}$ and the optimal directions of the previous iterations

$$P_{[p-1]} = \{\mathbf{p}_{[1]}, \mathbf{p}_{[2]}, \ldots, \mathbf{p}_{[p-1]}\}, \tag{2.10}$$

where $\mathbf{p}_{[i]} = \mathbf{t}_{[i+1]} - \mathbf{t}_{[i]}, i = 1, \ldots p-1$, ($\mathbf{t}_{[1]} = \mathbf{t}^{(n)}$). Since the $\mathbf{p}_{[i]} = \mathbf{t}_{[i+1]} - \mathbf{t}_{[i]}$ are optimal directions (including length) in iteration i, the trial vectors $\mathbf{t}_{[i]}$ may be interpreted as optimal points. We further know the Jacobian linear transformed vectors $\mathbf{s}_{[i]} = \mathbf{W}^{(n)}\mathbf{p}_{[i]}$ for the optimal directions in Eq. (2.10)

$$S_{[p-1]} = \{\mathbf{s}_{[1]}, \mathbf{s}_{[2]}, \ldots, \mathbf{s}_{[p-1]}\}. \tag{2.11}$$

Iteration p starts with generating the residual for the trial solution

$$\mathbf{r}_{[p]} = \mathbf{r}^{(n)}(\mathbf{t}_{[p]}) = \mathbf{w}^{(n)} + \mathbf{W}^{(n)}(\mathbf{t}_{[p]} - \mathbf{t}^{(n)}). \tag{2.12}$$

The trial solution of the $p+1^{\text{th}}$ CR subspace iteration may be parametrized as

$$\mathbf{t}_{[p+1]} = \mathbf{t}_{[p]} + \sum_{i=1}^{p-1} \alpha_i \mathbf{p}_{[i]} + \alpha_p \mathbf{r}_{[p]}. \tag{2.13}$$

The optimal search direction of iteration p therefore is

$$\mathbf{p}_{[p]} = \sum_{i=1}^{p-1} \alpha_i \mathbf{p}_{[i]} + \alpha_p \mathbf{r}_{[p]}, \tag{2.14}$$

and may be determined minimizing the residual norm $\|\mathbf{r}_{[p+1]}\|^2$. The minimization may be carried out solving a set of linear equations (see Ref [18] for details)

$$\begin{pmatrix} \langle \mathbf{s}_{[1]}|\mathbf{s}_{[1]}\rangle & 0 & \dots & 0 \\ 0 & \ddots & 0 & 0 \\ \vdots & \langle \mathbf{s}_{[p-2]}|\mathbf{s}_{[p-2]}\rangle & 0 & 0 \\ 0 & 0 & \langle \mathbf{s}_{[p-1]}|\mathbf{s}_{[p-1]}\rangle & \langle \mathbf{s}_{[p-1]}|\mathbf{b}_{[p]}\rangle \\ 0 & 0 & \langle \mathbf{b}_{[p]}|\mathbf{s}_{[p-1]}\rangle & \langle \mathbf{b}_{[p]}|\mathbf{b}_{[p]}\rangle \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_{p-2} \\ \alpha_{p-1} \\ \alpha_p \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 0 \\ \langle \mathbf{r}_{[p]}|\mathbf{b}_{[p]}\rangle \end{pmatrix}, \quad (2.15)$$

where

$$\mathbf{b}_{[p]} = \mathbf{W}^{(n)}\mathbf{r}_{[p]}, \quad (2.16)$$

$\mathbf{p}_{[p]}$ and $\mathbf{t}_{[p+1]}$ may now be determined and $\mathbf{s}_{[p]} = \mathbf{W}^{(n)}\mathbf{p}_{[p]}$ calculated using Eq. (2.16) and $\mathbf{s}_{[i]}, i = 1, p-1$ from Eq. (2.11). $\mathbf{p}_{[p]}$ and $\mathbf{s}_{[p]}$ may now be added to $P_{[p-1]}$ and $S_{[p-1]}$ and the sequence of CR subspace iterations is thus established.

The important feature of the CR subspace algorithm is the structure of the subspace equation Eq. (2.15) where the solution does not depend on the initial $p-2$ optimal trial directions. The trial vectors from these $p-2$ iterations therefore may be discarded without affecting the subspace solution. It is this feature that may be taken over in the DIIS sequence of iterations when it is transferred to a CROP sequence.

To understand how a DIIS sequence of iterations may be modified to a CR subspace sequence we write the CR subspace equation in Eq. (2.15) in a slightly modified form. The CR subspace equations arise from a minimization of the residual norm in the basis spanned by $\mathbf{p}_{[i]}, i = 1, \dots, p-1$ and $\mathbf{r}_{[p]}$ (see Eqs. (2.14), (2.15)). As the norm of a vector is independent of the basis in which it is expressed, any basis that spans the same space produces the same trial solution $\mathbf{t}_{[p+1]}$.

Considering $\mathbf{t}_{[p+1]}$ in Eq. (2.13) and introducing a preliminary improvement

$$\tilde{\mathbf{t}}_{[p+1]} = \mathbf{t}_{[p]} + \mathbf{r}_{[p]}, \quad (2.17)$$

Introducing a new set of variational parameters $c_i, i = 1, \dots, p-1$ and c_{p+1} we may write Eq. (2.13) after some modifications as

$$\mathbf{t}_{[p+1]} = \mathbf{t}_{[p]} + \sum_{i=1}^{p-1} c_i(\mathbf{t}_{[i]} - \mathbf{t}_{[p]}) + c_{p+1}(\tilde{\mathbf{t}}_{[p+1]} - \mathbf{t}_{[p]}), \quad (2.18)$$

which describes a linear parametrization of the basis

$$T^{\mathrm{CR}}_{[p+1]} = \{\mathbf{t}_{[1]}, \dots, \mathbf{t}_{[p]}, \tilde{\mathbf{t}}_{[p+1]}, \}, \quad (2.19)$$

with $\mathbf{t}_{[p]}$ as reference point. From the above derivation we see that $T^{\mathrm{CR}}_{[p+1]}$ spans the same space as is used for determining the optimal trial vector in Eq. (2.14). Using

$$\tilde{\mathbf{r}}_{[p+1]} = \mathbf{w}^{(n)} + \mathbf{W}^{(n)}(\tilde{\mathbf{t}}_{[p+1]} - \mathbf{t}^{(n)}), \quad (2.20)$$

the residual of $\mathbf{t}_{[p+1]}$ in Eq. (2.18) becomes

$$\begin{aligned} \mathbf{r}_{[p+1]} &= (1 - \sum_{i=1}^{p-1} c_i - c_{p+1})\mathbf{r}_{[p]} + \sum_{i=1}^{p-1} c_i \mathbf{r}_{[i]} + c_{p+1}\tilde{\mathbf{r}}_{[r+1]} \\ &= \sum_{i=1}^{p} c_i \mathbf{r}_{[i]} + c_{p+1}\tilde{\mathbf{r}}_{[p+1]}, \end{aligned} \quad (2.21)$$

where we have introduced the coefficient

$$c_p = 1 - \sum_{i=1}^{p-1} c_i - c_{p+1}. \tag{2.22}$$

Minimizing $\|\mathbf{r}_{[p+1]}\|^2$ of Eq. (2.21), with the constraint Eq. (2.22) will give the same subspace solution as solving Eq. (2.15) since the subspaces of Eqs. (2.13) and (2.18) are identical.

From Eq. (2.15) we know that the optimal solution is spanned by $\mathbf{r}_{[p]}$ and $\mathbf{p}_{[p-1]} = \mathbf{t}_{[p]} - \mathbf{t}_{[p-1]}$. The optimal trial vector is thus contained in the space that is spanned by $\mathbf{t}_{[p]}, \mathbf{t}_{[p-1]}$, and $\mathbf{r}_{[p]} = \tilde{\mathbf{t}}_{[p+1]} - \mathbf{t}_{[p]}$. The coefficients $c_i, i = 1 \ldots, p-2$ are therefore vanishing and the optimal coefficients may thus be determined in the subspace

$$T^{\mathrm{Red}}_{[p+1]} = \{\mathbf{t}_{[p-1]}, \mathbf{t}_{[p]}, \tilde{\mathbf{t}}_{[p+1]}\}, \tag{2.23}$$

by minimizing the residual

$$\mathbf{r}_{[p+1]} = c_{p-1}\mathbf{r}_{[p-1]} + c_p\mathbf{r}_{[p]} + c_{p+1}\tilde{\mathbf{r}}_{[p+1]}, \tag{2.24}$$

where the variational coefficients satisfy

$$c_{p-1} + c_p + c_{p+1} = 1. \tag{2.25}$$

In iteration p of a conjugate residual subspace iteration the subspace solution may thus be obtained either from a basis containing $\mathbf{p}_{[p-1]}$ and $\mathbf{r}_{[p]}$ solving Eq. (2.15) or from the basis $T^{\mathrm{Red}}_{[p+1]}$ minimizing the residual norm $\|\mathbf{r}_{[p+1]}\|^2$ of Eq. (2.24) with the constraint Eq. (2.25).

2.4 The DIIS method and its connection to the CR subspace method

When the DIIS method is used to determine the solution vector $\mathbf{t}^*$ in Eq. (2.3), the vector function value $\mathbf{w}(\mathbf{t})$ may be interpreted as an *error vector* since $\mathbf{w}(\mathbf{t}) - \mathbf{w}(\mathbf{t}^*) = \mathbf{w}(\mathbf{t})$. At iteration p of a DIIS sequence the sets of previous trial vectors

$$T^{\mathrm{DIIS}}_{[p]} = \left\{\mathbf{t}_{[1]}, \mathbf{t}_{[2]}, \ldots, \mathbf{t}_{[p]}\right\}, \tag{2.26}$$

and their associated error vectors

$$W^{\mathrm{DIIS}}_{[p]} = \left\{\mathbf{w}_{[1]}, \mathbf{w}_{[2]}, \ldots, \mathbf{w}_{[p]}\right\}, \tag{2.27}$$

are known where $\mathbf{w}_{[i]} = \mathbf{w}(\mathbf{t}_{[i]}), i = 1, \ldots p$. The minimal error in the space of $W^{\mathrm{DIIS}}_{[p]}$ is determined from a linear parametrization of the error vector space

$$\mathbf{w}^{\mathrm{opt}}_{[p]} = \sum_{i=1}^{p} \omega_i \mathbf{w}_{[i]}, \tag{2.28a}$$

$$\sum_{i=1}^{p} \omega_i = 1, \tag{2.28b}$$

where the weights ω_i are determined from a constrained minimization using Eq. (2.28b) as constraint. The optimal trial vector $\mathbf{t}_{[p]}^{\text{opt}}$ in DIIS is obtained assuming a *linear relation between trial vectors and error vectors*

$$\mathbf{t}_{[p]}^{\text{opt}} = \sum_{i=1}^{p} \omega_i \mathbf{t}_{[i]}, \tag{2.29}$$

and the trial solution $\mathbf{t}_{[p+1]}$ is constructed as the sum of the optimal contributions in their respective spaces

$$\mathbf{t}_{[p+1]} = \mathbf{t}_{[p]}^{\text{opt}} + \mathbf{w}_{[p]}^{\text{opt}}. \tag{2.30}$$

$\mathbf{w}_{[p+1]} = \mathbf{w}(\mathbf{t}_{[p+1]})$ is then calculated and the basis $T_{[p]}^{\text{DIIS}}$ and $W_{[p]}^{\text{DIIS}}$ are updated with $\mathbf{t}_{[p+1]}$ and $\mathbf{w}_{[p+1]}$, respectively, giving

$$T_{[p+1]}^{\text{DIIS}} = \{\mathbf{t}_{[1]}, \ldots, \mathbf{t}_{[p+1]}\}, \tag{2.31a}$$

$$W_{[p+1]}^{\text{DIIS}} = \{\mathbf{w}_{[1]}, \ldots, \mathbf{w}_{[p+1]}\}, \tag{2.31b}$$

and the DIIS sequence of subspace iterations is thus established.

Let us now consider a sequence of iterations where the optimal trial vectors $\mathbf{t}_{[i]}^{\text{opt}}$ of Eq. (2.29) are stored together with $\mathbf{t}_{[p+1]}$ of Eq. (2.30). We thus consider the basis of trial vectors and error vectors

$$T_{[p+1]}^{\text{aux}} = \{\mathbf{t}_{[1]}^{\text{opt}}, \ldots, \mathbf{t}_{[p]}^{\text{opt}}, \mathbf{t}_{[p+1]}\}, \tag{2.32a}$$

$$W_{[p+1]}^{\text{aux}} = \{\mathbf{w}_{[1]}^{\text{opt}}, \ldots, \mathbf{w}_{[p]}^{\text{opt}}, \mathbf{w}_{[p+1]}\}. \tag{2.32b}$$

From Eq. (2.29) it is seen that the basis $T_{[p+1]}^{\text{aux}}$ and $T_{[p+1]}^{\text{DIIS}}$ are spanning the same subspace and the same subspace solution is therefore obtained whether the basis in Eq. (2.31) or the basis in Eq. (2.32) is used, assuming a linear relation between trial vectors and error vectors. From Eqs. (2.5) and (2.8) we further obtain

$$\begin{aligned} \mathbf{w}(\mathbf{t}_{[i]}) &= \mathbf{w}^{(n)} + \mathbf{W}^{(n)}(\mathbf{t}_{[i]} - \mathbf{t}^{(n)}) + \mathcal{O}(\|(\mathbf{t}_{[i]} - \mathbf{t}^{(n)})\|^2) \\ &= \mathbf{r}^{(n)}(\mathbf{t}_{[i]}) + \mathcal{O}(\|(\mathbf{t}_{[i]} - \mathbf{t}^{(n)})\|^2). \end{aligned} \tag{2.33}$$

Neglecting terms of order $\mathcal{O}(\|(\mathbf{t}_{[i]} - \mathbf{t}^{(n)})\|^2)$ we can thus formally replace error vectors $\mathbf{w}_{[i]}$ with residuals $\mathbf{r}^{(n)}(\mathbf{t}_{[i]}) = \mathbf{r}_{[i]}$.

New trial vectors in DIIS are obtained from Eq. (2.30) and in the CR subspace algorithm from Eq. (2.17). For both the DIIS and the CR subspace algorithm an improved trial vector is thus generated by adding the residual for the optimal trial vector to the optimal trial vector. Further, the optimal trial vectors are obtained minimizing residual norms in both the DIIS and the CR subspace algorithm. If the trial vectors of the DIIS algorithm are stored as in Eq. (2.32) the DIIS and the CR subspace algorithm become therefore identical. When trial vectors in DIIS are stored in accordance with (2.32) we may neglect trial vectors from all but the last two iterations and thus determine the optimal expansion coefficients using Eqs. (2.23), (2.24) and (2.25). The above is of course valid under the assumption of a weak nonlinearity in the CC equation but this is anyway the fundamental assumption we have made in DIIS by using a linear relationship between trial vectors and error vectors. The bases of Eq. (2.32) were originally introduced by Ziolkowski et al. [18] and the algorithm that uses these vectors was denoted the conjugate residual with optimal trial vectors (CROP) algorithm. The above derivation clearly shows that a standard DIIS implementation may be changed to a CROP implementation by altering a few lines of code as we discuss in Section 2.5.

2.5 Implementation of DIIS and CROP

In the previous discussion common features of the CR subspace and DIIS algorithms were identified and it was shown that merely the subspaces $T^{\text{DIIS}}_{[n]}$ and $W^{\text{DIIS}}_{[n]}$ of Eq. (2.31) have to be replaced by $T^{\text{aux}}_{[n]}$ and $W^{\text{aux}}_{[n]}$ of Eq. (2.32) to move from a DIIS implementation to a CR subspace implementation. In Section 2.3, new trial vectors are obtained from Eq. (2.17) while in Section 2.4 they are obtained from the corresponding Eq. (2.30). To align the notation of Eqs. (2.17) and (2.30) in the algorithms we use the notation of Eq. (2.30), i.e. $\tilde{\mathbf{t}}_{[p+1]} = \mathbf{t}_{[p]} + \mathbf{r}_{[p]}$ becomes $\mathbf{t}_{[p+1]} = \mathbf{t}^{\text{opt}}_{[p]} + \mathbf{w}^{\text{opt}}_{[p]}$ ($\mathbf{r}_{[p]}$ is replaced by $\mathbf{w}^{\text{opt}}_{[p]}$ according to Eq. (2.33)). The notation of the equations referenced in the CR section has to be changed accordingly. It is furthermore convenient to introduce a third type of basis, consisting only of the optimal trial/error vectors

$$T^{\text{opt}}_{[p]} = \{\mathbf{t}^{\text{opt}}_{[1]}, \ldots, \mathbf{t}^{\text{opt}}_{[p]}\}, \tag{2.34a}$$

$$W^{\text{opt}}_{[p]} = \{\mathbf{w}^{\text{opt}}_{[1]}, \ldots, \mathbf{w}^{\text{opt}}_{[p]}\}. \tag{2.34b}$$

The DIIS (`Algorithm 1`) and the CROP (`Algorithm 2`) algorithms are given side by side to highlight the code changes necessary to move from an existing DIIS implementation to a CROP implementation. In both algorithms, code lines `1 - 3` are identical initial steps, where we have chosen to solve the (CCSD) equations in a (pseudo–) canonical basis. Note that $\mathbf{t}_{[1]}$ in $T^{\text{DIIS}}_{[1]}$ is identical to $\mathbf{t}_{[1]} = \mathbf{t}^{\text{opt}}_{[1]}$ in $T^{\text{aux}}_{[1]}$. The first steps in the iterative loop are identical:

1. performing space truncations if necessary in lines `5-7`,
2. obtaining the error vector in line `8`,
3. checking for convergence in line `9`,
4. adding the error vector to the iterative subspace in line `10`, and
5. solving for an optimal error vector in accordance with Eq. (2.28)/(2.21) for DIIS/CROP in line `11`.

The optimal error/trial vectors in DIIS are constructed according to (2.28)/(2.29) and for CROP according to (2.21)/(2.18) with (2.22) in code lines `12/13`. In the CROP algorithm, the error/-trial vector $\mathbf{w}_{[n]}/\mathbf{t}_{[n]}$ in $W^{\text{aux}}_{[n]}/T^{\text{aux}}_{[n]}$ has to be replaced by $\mathbf{w}^{\text{opt}}_{[n]}/\mathbf{t}^{\text{opt}}_{[n]}$ to arrive at $W^{\text{opt}}_{[n]}/T^{\text{opt}}_{[n]}$ in line `14/15` of `Algorithm 2`. All the following steps in the algorithm are identical in DIIS and CROP, i.e., the construction of the new trial vector according to Eq. (2.30), and adding the trial vector to the set of previous trial vectors.

Algorithm 1: The DIIS algorithm

1: Diagonalize Fock and transform to the diagonal basis
2: get starting guess $\mathbf{t}_{[1]}$ and store $T^{\mathtt{DIIS}}_{[1]}$
3: $f = 1$

4: for n=1,maxiter
5: if(n>m):
6: truncate $W^{\mathtt{DIIS}}_{[n-1]}$ and $T^{\mathtt{DIIS}}_{[n]}$
7: $f = n - m + 1$
8: get $\mathbf{w}_{[n]} = \mathbf{w}(\mathbf{t}_{[n]})$
9: if ($\|\mathbf{w}_{[n]}\| <$ thr)exit
10: add $\mathbf{w}_{[n]}$ to $W^{\mathtt{DIIS}}_{[n-1]} \rightarrow W^{\mathtt{DIIS}}_{[n]}$
11: solve for ω_i
12: $\mathbf{w}^{\mathtt{opt}}_{[n]} = \sum_{i=f}^{n} \omega_i \mathbf{w}_{[i]}$
13: $\mathbf{t}^{\mathtt{opt}}_{[n]} = \sum_{i=f}^{n} \omega_i \mathbf{t}_{[i]}$

14: get $\mathbf{t}_{[n+1]} = \mathbf{t}^{\mathtt{opt}}_{[n]} + \mathbf{w}^{\mathtt{opt}}_{[n]}$
15: add $\mathbf{t}_{[n+1]}$ to $T^{\mathtt{DIIS}}_{[n]} \rightarrow T^{\mathtt{DIIS}}_{[n+1]}$
16: end iter

17: transform integrals and amplitudes back to the original basis

Algorithm 2: The CROP algorithm

1: Diagonalize Fock and transform to the diagonal basis
2: get starting guess $\mathbf{t}_{[1]}$ and store $T^{\mathtt{aux}}_{[1]}$
3: $f = 1$

4: for n=1,maxiter
5: if(n>m):
6: truncate $W^{\mathtt{opt}}_{[n-1]}$ and $T^{\mathtt{aux}}_{[n]}$
7: $f = n - m + 1$
8: get $\mathbf{w}_{[n]} = \mathbf{w}(\mathbf{t}_{[n]})$
9: if ($\|\mathbf{w}_{[n]}\| <$ thr)exit
10: add $\mathbf{w}_{[n]}$ to $W^{\mathtt{opt}}_{[n-1]} \rightarrow W^{\mathtt{aux}}_{[n]}$
11: solve for ω_i
12: $\mathbf{w}^{\mathtt{opt}}_{[n]} = \sum_{i=f}^{n-1} \omega_i \mathbf{w}^{\mathtt{opt}}_{[i]} + \omega_n \mathbf{w}_{[n]}$
13: $\mathbf{t}^{\mathtt{opt}}_{[n]} = \sum_{i=f}^{n-1} \omega_i \mathbf{t}^{\mathtt{opt}}_{[i]} + \omega_n \mathbf{t}_{[n]}$
14: replace $\mathbf{w}_{[n]}$ by $\mathbf{w}^{\mathtt{opt}}_{[n]}$ in $W^{\mathtt{aux}}_{[n]} \rightarrow W^{\mathtt{opt}}_{[n]}$
15: replace $\mathbf{t}_{[n]}$ by $\mathbf{t}^{\mathtt{opt}}_{[n]}$ in $T^{\mathtt{aux}}_{[n]} \rightarrow T^{\mathtt{opt}}_{[n]}$
16: get $\mathbf{t}_{[n+1]} = \mathbf{t}^{\mathtt{opt}}_{[n]} + \mathbf{w}^{\mathtt{opt}}_{[n]}$
17: add $\mathbf{t}_{[n+1]}$ to $T^{\mathtt{opt}}_{[n]} \rightarrow T^{\mathtt{aux}}_{[n+1]}$
18: end iter

19: transform integrals and amplitudes back to the original basis

The present implementation of the CROP algorithm in **LSDALTON** is written in terms of the tensor library which will be discussed in Chapter 3. Therefore, the algorithm may handle both dense tensors and parallel distributed tensors. Since the tensor library is used, the CROP solver has minimal local storage requirements both because only three subspace vectors are needed and because the vectors are allocated in a distributed fashion among all nodes. Hence, both the algorithm and the computational realization are directed towards treating large molecular systems on supercomputers. The time–determining step of a given iteration is the evaluation of the vector function (line 8 in `Algorithms 1` and `2`). In Chapter 4 we will explicitly discuss how the residual (error) vector may be constructed for the CCSD method.

2.6 Numerical results

In this section we compare the convergence of the DIIS and CROP algorithms numerically. To illustrate the convergence properties we report CCSD (coupled cluster singles and doubles) calculations on anthracene/ozone using Dunning's cc-pVDZ/cc-pVTZ [28] basis set and solving the CCSD equations in the canonical HF basis and in a local basis [27] obtained by minimizing the second power of the second moment localization function, for both the occupied and virtual orbital space.

For the anthracene (ozone) molecule the norms of the error vectors are plotted both, for the CROP and DIIS algorithms, against the number of iterations in the canonical basis in Figure 2.1 (Figure 2.3) and in the local MO basis [27] in Figure 2.2 (Figure 2.4). We report calculations

Figure 2.1: CROP (*left*) and DIIS (*right*) iteration sequences for a CCSD calculation on anthracene using a cc-pVDZ basis in the canonical basis for the subspace truncation parameters $m = 3, 5, \infty$.

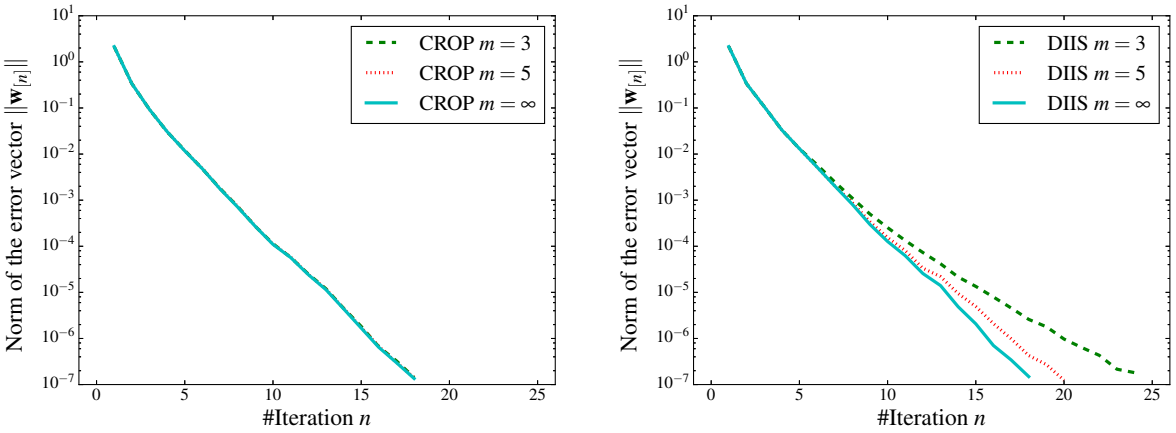
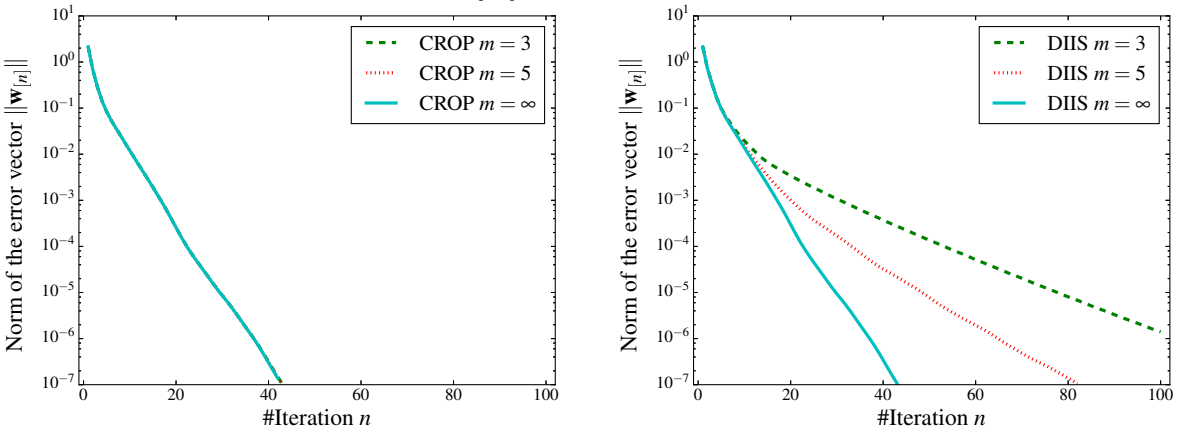


Figure 2.2: CROP (*left*) and DIIS (*right*) iteration sequences for a CCSD calculation on anthracene using a cc-pVDZ basis in the local basis [27] for the subspace truncation parameters $m = 3, 5, \infty$.



for subspaces containing the last $m = 3, 5, \infty$ trial vectors, where in the limit $m = \infty$ DIIS and CROP give the same sequence of iterations. The iteration sequences for the subspace sizes $m = 3, 5, \infty$ are basically superimposed using the CROP algorithm, due to the weak nonlinearity of the CCSD vector function which is independent of basis. For DIIS the choice of subspace does have a considerable influence on the convergence, where the effect of discarding subspace information is more severe in the local basis than in the canonical basis. Due to the weaker diagonal dominance of the CCSD Jacobian in the local basis than in the canonical basis more iterations are necessary to solve the CCSD amplitude equations in the local basis.

In contrast to anthracene, the ground state of ozone has a multi–configurational character, and the nonlinear parts in the CCSD amplitude equations therefore become more prominent. Still the CROP algorithm is capable of handling this situation and the iteration sequences for CROP and subspace truncation parameters $m = 3, 5, \infty$ are nearly indistinguishable. In particular, for ozone using the local basis, the DIIS algorithm suffers from the subspace truncation. For the subspace truncation parameter $m = 3$, 445 iterations are required, while for $m = 5$, convergence is obtained in 182 iterations. The calculations reported in Figures 2.1 - 2.4 clearly demonstrate the efficiency of carrying out subspace truncations using the CROP algorithm.

Figure 2.3: CROP (*left*) and DIIS (*right*) iteration sequences for a CCSD calculation on ozone using a cc-pVTZ basis in the canonical basis for the subspace truncation parameters $m = 3, 5, \infty$.

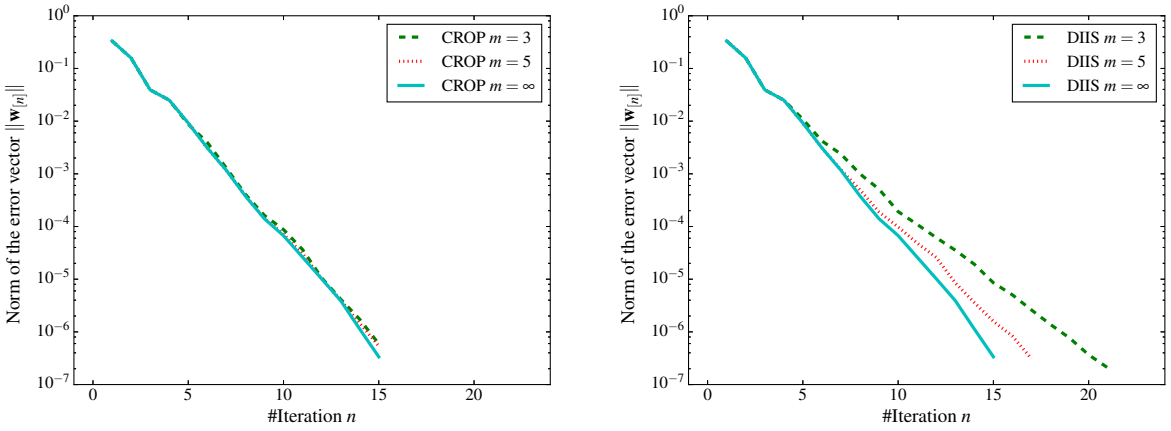
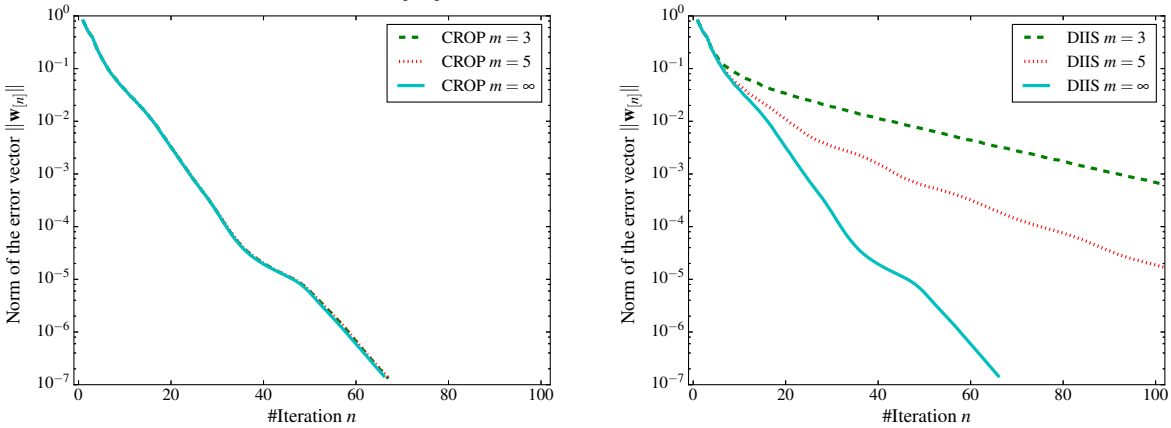


Figure 2.4: CROP (*left*) and DIIS (*right*) iteration sequences for a CCSD calculation on ozone using a cc-pVTZ basis in the local basis [27] for the subspace truncation parameters $m = 3, 5, \infty$.



2.7 Conclusion

The simplified route to obtain a connection between the CROP and the DIIS algorithm described in this chapter simplifies the understanding of the CROP implementation considerably, as compared to previous strategies. Furthermore, the numerical results underline the importance of using the CROP algorithm when solving CCSD equations, as a truncated subspace in the CROP algorithm does not lead to a serious abatement in the convergence. The conclusions drawn in this section become even more pronounced, when solving CCSDT (CCSDTQ, ...) equations as the nonlinearity of these equations is not far more pronounced, while the larger trial vectors may increase the need for a subspace truncation.

The CROP algorithm is the ideal tool for solving non–linear equations, especially when a supercomputer environment is targeted since a) due the reduced number of trial vectors only minimal storage is required and thus I/O may be avoided altogether and b) the uncomplicated implementation simplifies the distribution of the vectors, reducing the per–node memory requirements to a minimum. How the memory distribution may be achieved is the subject of Chapter 3.

Chapter 3

Handling of parallel distributed tensors in LSDALTON

This section describes work published in the form of a **LSDALTON** *software release [29] and is mainly intended for the computationally inclined reader implementing in* **LSDALTON** *which is why it contains many details about the implementation; in other words, this section may be skipped.*

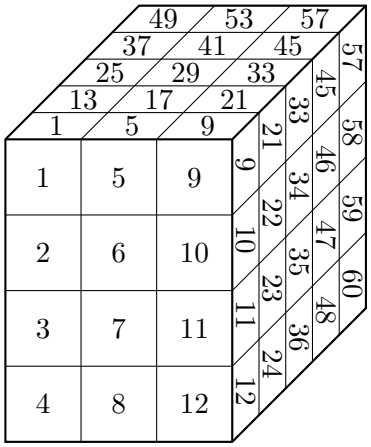
Introduction

When describing many–body correlations in physics, mathematical objects with multiple indices arise naturally in the derived formulae. These objects are known as tensors [30]. The number of indices of a tensor is called the mode of the tensor, and the number of values a mode may assume is called the dimension in that mode. Handling tensors on computers is aggravated by the multiple indices of the tensors, since the data layout is usually linear. It has been an effort in recent years to design computational libraries, which can handle tensors for modern high–level mathematical programs and low–level programming languages, such as the Tensor Tool Box [31], ltensor [32] or Cyclops [33]. However, for the low–level programming languages, one would ideally want a standardized interface to high–performance vendor–optimized libraries, which, currently, is not available. On the other hand, many mathematical methods and thus computational tools, designed for linear algebra, can be applied to tensors directly or after minor modifications of the tensor.

For vectors (one–mode tensors) and matrices (two–mode tensors), well–established low–level tools exist for the most important operations. These, the Basic Linear Algebra Subprograms (BLAS) and Linear Algebra PACKage (LAPACK) libraries, have standardized interfaces and are usually vendor–optimized and they are therefore among the most efficient codes available (in terms of utilizing the available computational resources). In fact, LinPACK, a predecessor of LAPACK, is used to measure the performance of computational systems. Many tensor operations may be reformulated in a way such that they can utilize BLAS and LAPACK routines, e.g. by explicit loops over (one or two mode) sub–tensors, which are fit to enter the standardized BLAS and LAPACK interfaces. Another possibility is to unfold the full tensors as one or two mode tensors (combining multiple indices to a common index, i.e. data sorting) and use BLAS and LAPACK in connection with the unfolded tensor (or unfolding).

When calculations are to be performed with huge amounts of data (big matrices), there are standardized tools (parallel BLAS and scalable LAPACK) that implement the parallel

Figure 3.1: An example three mode tensor where each element of the tensor is indicated by a cube with its column major global index as it would be stored in a linear stretch of memory allocated by FORTRAN. We may write the dimensions in the modes as [4,3,5].



distribution of vectors and matrices among the memory of up to hundreds of nodes as well as the operations on the parallel distributed data. For tensors, a variety (but no standardized set) of tools exist that do the same, but since another dependency layer of the **LSDALTON** code is not desirable, the most important functions for carrying out parallel tensor algebra have been implemented. This chapter will focus on the basic design ideas behind the parallel tensor module in **LSDALTON**.

3.1 The tensor interface

The tensor interface in **LSDALTON** is designed for general tensor handling, while making as much use as possible of the standard BLAS routines, which are usually highly optimized. Currently, there are many tensor operations implemented for different kinds of tensors, e.g., dense tensors, replicated tensors (stored compactly on one node with copies on all other nodes), (replicated) tiled tensors and parallel distributed tiled tensors, and combinations of these. Tensor contractions are the most time–consuming operations in the algorithms we are interested in, and we will therefore mainly focus on the implementation and performance of these. The programming paradigm is always to sort the data, before making use of the BLAS routines. This is motivated by the fact that the scaling of a tensor contraction is more expensive than sorting the data before and after the contraction. For example, when contracting one mode of two three–mode tensor with dimensions $[i_n, k_n, j_n]$ and $[m_n, k_n, l_n]$, sorting the both tensors will scale as $i_n \cdot j_n \cdot k_n + m_n \cdot k_n \cdot l_n$, while the contraction will scale as $i_n \cdot j_n \cdot k_n \cdot l_n \cdot m_n$, where the ratio of the number of operations in the sortings and contractions is obviously dependent on the contraction pattern. Thus, the sorting, when implemented efficiently, will usually be a small part of the contraction. On the other hand, tensor additions (and similar BLAS-1 like operations), which are associated with a sorting of the tensor, will be performed directly by the sorting routines.

3.1.1 Dense tensor algebra by tensor sorting and BLAS routines

In the FORTRAN programming language, tensorial quantities up to mode seven tensors may be allocated directly as "multidimensional arrays", e.g. for a mode three array as `real, pointer, dimension(:,:,:)`. This way of declaring tensorial quantities allows only for easy access to the individual elements and/or sub–tensor extraction and does not straightforwardly allow for an (efficient) algebraic treatment of the data. Furthermore, the underlying physical storage of the data will be vector–like, and therefore cache efficient algorithms need to be used whenever tensor algebra is carried out. In order to use BLAS whenever possible, one of the previously described strategies is required in order to map the tensor data to a matrix like form. We have chosen the strategy of unfolding the tensors completely before calling a BLAS routine with the fully unfolded tensor because the memory bound unfolding of a tensor may almost achieve optimal performance in terms of memory bandwidth.

The tensor unfolding is a sorting of data, which has been implemented in **LSDALTON**, using a python script that generates the FORTRAN code for all necessary data sortings. The code generated by the script is designed to be easily understandable for the compiler, while being cache efficient. In order to take advantage of shared memory processing (SMP), there is a loop over blocks of data, which may be processed by different threads, see `Algorithm 3`. Each operation (simple sorting (`Algorithm 3`), sort and add, and sort-scale-add) add have their own kernel in order to avoid redundant operations (e.g., scaling with 1). Note that all the kernels perform out–of–place operations, since it is simpler to avoid cache misses and thus increase the performance of the algorithm.

`Algorithm 3: Sorting kernel of a tensorial quantity for the [3,1,4,2] sorting`

```
!$OMP PARALLEL DO COLLAPSE(4)
do block4=1,nblocks4
  do block3=1,nblocks3
    do block2=1,nblocks2
      do block1=1,nblocks1

        do elm4 = 1,nelm4
          do elm3 = 1,nelm3
            do elm2 = 1,nelm2
              do elm1 = 1,nelm1

                new_order(elm1,elm2,elm3,elm4) = &
                  &old_order(elm2,elm4,elm1,elm3)
```

With an efficient sorting algorithm like this, all tensor contractions, e.g. the example contraction from the introduction has the contraction pattern

$$C_{iljm} = \alpha \sum_k A_{ikj} B_{mkl} + \beta C_{iljm}, \tag{3.1}$$

and index restrictions

$$i \in i_n,\ j \in j_n,\ k \in k_n,\ l \in l_n \text{ and } m \in m_n. \tag{3.2}$$

The contraction may then be performed using BLAS routines and tensor sorting in the following way

1. `call sort` $\mathbf{A}[i_n, k_n, j_n]$ with $[1, 3, 2]$ to obtain $\mathbf{A}'[i_n, j_n, k_n]$
2. `call sort` $\mathbf{B}[m_n, k_n, l_n]$ with $[2, 1, 3]$ to obtain $\mathbf{B}'[k_n, m_n, l_n]$
3. `call dgemm` with α, $\mathbf{A}'$, $\mathbf{B}'$, β and obtain $\mathbf{C}'[i_n, j_n, m_n, l_n]$ (where the matrix dimensions in `dgemm` are $\mathtt{m}_{gemm}=i_n \cdot j_n$, $\mathtt{n}_{gemm}=m_n \cdot l_n$, $\mathtt{k}_{gemm}=k_n$)
4. `call sort` $\mathbf{C}'[i_n, j_n, m_n, l_n]$ with $[1, 4, 2, 3]$ to add to $\beta \cdot \mathbf{C}[i_n, l_n, j_n, m_n]$

Of course, this strategy increases the memory requirements of the whole algorithm as $\mathbf{A}'$, $\mathbf{B}'$ and $\mathbf{C}'$ need to be stored in addition to $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$, unless the memory may be reused if $\mathbf{A}$ or $\mathbf{B}$ are not needed anymore. On the other hand, using a cache–inefficient in–place sorting (or looping over, possibly non–consecutive, blocks of the tensor) makes the overall algorithm prohibitively slow according to our (unpublished) test calculations. Thus, in order to keep the memory requirements manageable, any algorithm using this strategy will have to use an (explicit) workspace and the algorithm design should take the increased memory requirements into account.

3.1.2 Distributed tensor algebra

Parallel memory distribution is achieved by splitting a tensor into tiles, which may then be distributed over the group of nodes sharing the computation. Thereby the size of a tile will be a compromise between load balancing and communication speed. Here, we will first write the example tensor contraction from Section 3.1.1 for a tiled tensor. Then we will discuss the tiling and tile mapping to the nodes as implemented in **LSDALTON**, and finally we will discuss the implementation of a general BLAS-1 like operation and the current tensor contraction routine.

Using tiled algebra in connection with tensors has the attractive feature that a contraction may be expressed using tiles in the same way as indices, restricting the indices to the tile. In order to obtain a tile, the dimensions of the modes will be segmented, e.g., i_n will be segmented into I_n segments of length I (similarly for j_n, k_n, l_n and m_n). A tile is then a sub–tensor with the segments as dimensions (see Figure 3.2), e.g. one tile of tensor $\mathbf{A}$ may be written as $\mathbf{A}_{IKL}$. We may then write the contraction of Eq. (3.1) for the $\mathbf{C}_{ILJM}$ tile as

$$\mathbf{C}_{ILJM} = \alpha \sum_K \mathbf{A}_{IKJ}\mathbf{B}_{MKL} + \beta \mathbf{C}_{ILJM}, \tag{3.3}$$

and restrict the indices as

$$i \in I,\ j \in J,\ k \in K,\ l \in L \text{ and } m \in M. \tag{3.4}$$

Furthermore, the individual elements of the tile are contracted as Eq. (3.1), using the restrictions in Eq. (3.4) rather than the restrictions in Eq. (3.2). Therefore, we may reuse the strategy developed for dense tensors with the addition that a certain tile has to loaded into the local memory at the time the contraction is carried out, i.e. before the contraction we add

I. load tile A_{IKJ}

II. load tile B_{MKL}

and then execute the algorithm for local dense tensor contractions. Note that a node only needs to load a subset of the A_{IKJ} and B_{MKL} tiles, i.e. only the tiles with coinciding I, J, L, M for the C_{ILJM} that are stored locally. Also, in practice the tiles will be loaded asynchronously in order to overlap communication with computation.

Tile mapping in LSDALTON

In **LSDALTON**, we have chosen the tile mapping in Figure 3.2, which assigns a global tile index to the tiles in accordance with the column major numbering of elements in an array given by the FORTRAN standard (beginning with 1). In order to calculate the mode tile index $[t_1, t_2, \ldots, t_m]$ and the index within a tile $[d_1, d_2, \ldots, d_m]$ from the global index of an element $[i_1, i_2, \ldots, i_m]$ of an m mode tensor with dimensions $[i_{n_1}, i_{n_2}, \ldots, i_{n_m}]$ and tile dimensions $[d_{n_1}, d_{n_2}, \ldots, d_{n_m}]$

$$[t_1, t_2, \ldots, t_m] \text{ with } t_k = \lfloor (i_k - 1)/d_{n_k} \rfloor + 1, \tag{3.5}$$

$$[d_1, d_2, \ldots, d_m] \text{ with } d_k = \mathrm{mod}(i_k - 1,\ d_{n_k}) + 1. \tag{3.6}$$

Furthermore, it is important to know how many tiles there are in each mode of the tensor. This, we obtain as

$$[t_{n_1}, t_{n_2}, \ldots, t_{n_m}] \text{ with } t_{n_k} = \lceil i_{n_k}/d_{n_k} \rceil, \tag{3.7}$$

and we may then calculate the column major index of a given tile c_1, using the standard expression

$$c_1 = t_1 + \sum_{r=2}^{m} (t_r - 1) \prod_{s=1}^{r-1} t_{n_s}, \tag{3.8}$$

which may be used to calculate the index of the element in the tile c_2, replacing the mode tile index $[t_1, t_2, \ldots, t_m]$ with the index in the tile $[d_1, d_2, \ldots, d_m]$ and replacing the number of tiles in each mode $[t_{n_1}, t_{n_2}, \ldots, t_{n_m}]$ with the actual dimensions of the tile. In **LSDALTON** padding the tiles with zeros if $\mathrm{mod}\,(i_{n_k}, d_{n_k}) \neq 0$ is implemented, but it was found to be simpler to calculate the actual dimensions of the tile as

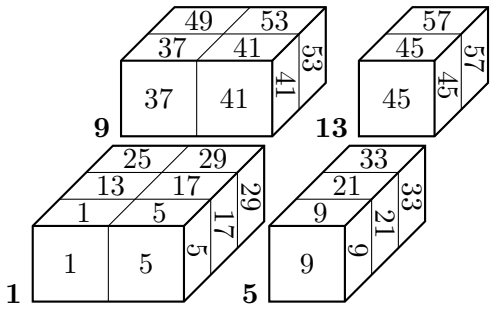
$$[d_{n_1}, d_{n_2}, \ldots, d_{n_m}] \text{ with } d_{n_k} = \begin{Bmatrix} d_{n_k} & \text{if } t_k \cdot d_{n_k} < i_{n_k} \\ \mathrm{mod}(i_{n_k}, d_{n_k}) & \text{otherwise} \end{Bmatrix}, \tag{3.9}$$

and store the actual dimensions of the individual tiles (see Figure 3.2 for an example three mode tensor).

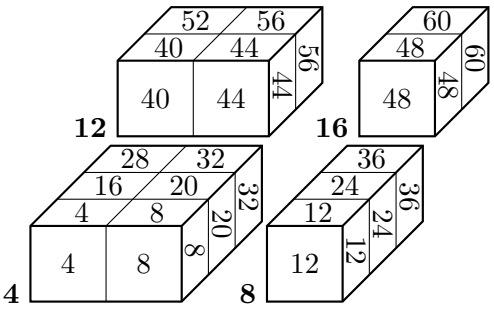
When, for the same m-mode tensor, the identification of an element, in terms of the column major tile index c_1 and the column major position in the tile c_2 are available, the global index $[i_1, i_2, \ldots, i_m]$ of the element may be calculated by first calculating the tile mode index $[t_1, t_2, \ldots, t_m]$ and the mode index of the element in the tile $[d_1, d_2, \ldots, d_m]$ from c_1 and c_2 respectively. This may be done by the following elimination procedure

```
1: x_1 = c_1
2: for k = 1, m:
3:     t_k     = mod(x_k - 1,  t_{n_k}) + 1
4:     x_{k+1} = (x_k - t_k) / t_{n_k} + 1
```

Figure 3.2: Tiling of the example tensor of Figure 3.1 with the dimensions $[i_{n_1}, i_{n_2}, i_{n_3}] = [4, 3, 5]$. The tiles are chosen to be of size $[d_{n_1}, d_{n_2}, d_{n_3}] = [1, 2, 3]$ and the (column major) global tile numbering is written in bold next to the tile. This tiling patterns results in 16 tiles (4 tiles for the first mode, and 2 for the two other modes).



(a) The four tiles of the (1,:,:) plane of the tensor are indexed as $c_2 = 1$ ($[t_1, t_2, t_3] = [1, 1, 1]$, left front), $c_2 = 5$ ($[t_1, t_2, t_3] = [1, 2, 1]$, right front), $c_2 = 9$ ($[t_1, t_2, t_3] = [1, 1, 2]$, left back) and $c_2 = 13$ ($[t_1, t_2, t_3] = [1, 2, 2]$, right back).



(b) The four tiles of the (4,:,:) plane of the tensor are indexed as $c_2 = 4$ ($[t_1, t_2, t_3] = [4, 1, 1]$, left front), $c_2 = 8$ ($[t_1, t_2, t_3] = [4, 2, 1]$, right front), $c_2 = 12$ ($[t_1, t_2, t_3] = [4, 1, 2]$, left back) and $c_2 = 16$ ($[t_1, t_2, t_3] = [4, 2, 2]$, right back).

where, again replacing $[t_1, t_2, \ldots, t_m]$ and $[t_{n_1}, t_{n_2}, \ldots, t_{n_m}]$ with $[d_1, d_2, \ldots, d_m]$ and $[d_{n_1}, d_{n_2}, \ldots, d_{n_m}]$ gives the procedure for c_2. Once the mode indices are obtained, it is straightforward to calculate the global index $[i_1, i_2, \ldots, i_m]$ as

$$[i_1, i_2, \ldots, i_m] \text{ with } i_k = d_k + t_k \cdot t_{n_k}, \tag{3.10}$$

and the position in a column major dense tensor is obtained using Eq. (3.8). Now the mappings have been discussed in great detail, and we are left to state that the tiles are distributed according to their column major index c_2 in a round robin fashion among the nodes (see Figure 3.3), where ,per default, the next tensor will have an offset o (equal to the sum of tiles in all previously allocated tensors) in the distribution, such that the node n and the local tile number of the tile on the node p may be calculated as

$$n = \text{mod}(c_1 - 1 + o,\ n_{\text{nodes}}), \tag{3.11}$$

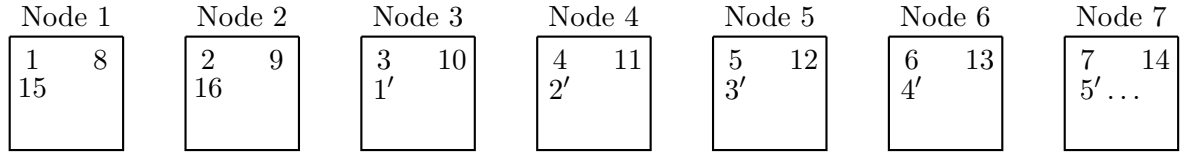
$$p = \lfloor (c_1 - 1)/n_{\text{nodes}} \rfloor + 1, \tag{3.12}$$

where n_{nodes} is the total number of nodes over which the tiles are distributed (see Figure 3.3). Note that the node n will be the rank of the respective node, i.e. the counting begins from 0, while p is the pth tile on the node, beginning from 1. It is known that this type of layout is not ideal with respect to a general tensor contraction, but we will demonstrate numerically in Section 3.2 that also with this layout a reasonable performance may be achieved.

Tiled tensor operations

For both the dense and the tiled tensor operations, a number of routines have been implemented. These routines range from simple memory aligned (BLAS-1-like) operations to tensor contractions and special functions necessary for the implemented correlation methods (CROP solver,

Figure 3.3: Round–robin tile distribution of a tensor with 16 tiles on 7 nodes ($n_{\text{nodes}} = 7$) where the tile index is written in the node where the tile resides. The tiles for the next tensor will be distributed with an offset as indicated by the primed tile indices.



MP2, CCSD and DEC routines, see Chapters 2, 4, and 6). For all the implemented functions, the strategy has been to first fill a buffer with `MPI_GET` and, once it has been filled, begin to work on the tiles using the sorting and BLAS (or specific operations) routines as outlined the beginning of this section. Whenever it is possible, explicit OpenMP parallelization has been introduced in the special routines (BLAS usually is OpenMP parallel). In the following, section we will examine the scaling behavior of a single tensor contraction.

3.2 Results

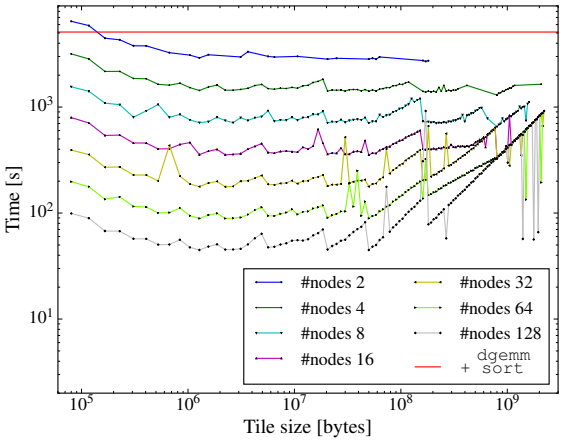
In this section, we would like to demonstrate a) that local sorting in connection with a call to `dgemm` is a fast approach, where the time used for sorting is only a fraction of the total contraction time, and b) how the parallel tensor contractions implemented in the way outlined in this chapter perform i) with respect to the local sorting, and ii) with respect to the number of nodes and the tile size used for the contraction. All calculations have been performed on **Eos** (see Section 9.2) using 1 or 15 OpenMP threads (plus one communication thread, see Appendix A) for all calculations. We will compare results for the contraction

$$C_{jnim} = \sum_{kl} A_{iklj} B_{mlkn}, \tag{3.13}$$

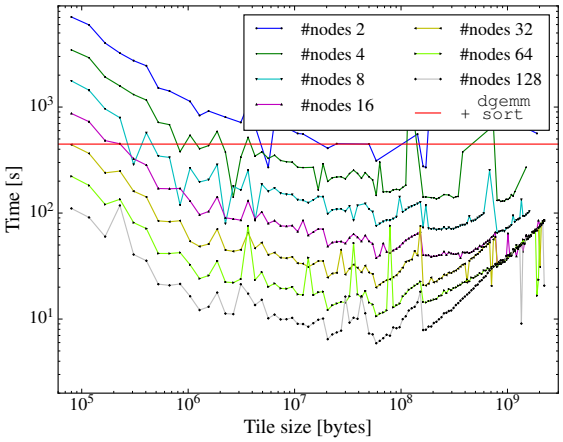
where we have chosen all tensors to be hypercubes with dimensions equal to 200, and the tiles being hypercubes of varying dimensions from 10 to 120 in steps of 1. We have, on purpose, chosen a contraction which involves three tensor sortings and only one call to `dgemm`. When carrying out the contraction with all tensors and using 1 (15) thread(s), the time for the `dgemm`/+sortings amounted to 5085.3 s/5102.5 s (442.8 s/447.3 s), where the reference for the `dgemm`+sortings is shown in Figures 3.4a and 3.4c (3.4b and 3.4d) as a red horizontal line. Also, these timings demonstrate that the sortings only account for about 1% of the total tensor contraction operation, and that the `dgemm` (sorting) scales well (fair) with respect to the number of cores with a factor of 11.4 (3.8) for this tensor size. Of course, the size of the tensor and the contraction pattern are rather beneficial for the ratio `dgemm`/sorting and other contraction patterns and tensor dimensions will lead to a different ratio, which remains to be tested.

Figure 3.4a shows the timings for the contraction run on one thread per node (the red line indicates the calculation on a single core), and thus we obtain the isolated speedup through the MPI parallelization. Increasing the tile size for a given number of nodes initially reduces the time fort the contraction and then becomes roughly constant. When the tiles become large, the time for the contraction increases. The initial decrease occurs from tile sizes of about 78kb to 1,2Mb, and this is probably the region, where the communication changes from being latency

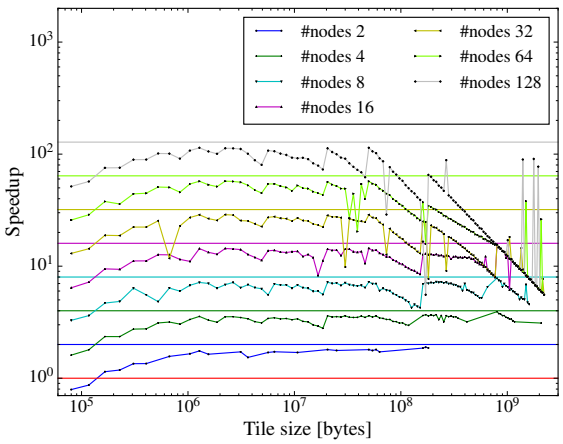
Figure 3.4: We compare the timings and speedups of the example contraction from Eq. (3.13) for 2^x nodes with $x = 1 \dots 7$. The reference timings and speedups are given as red horizontal lines. For the speedups we have furthermore given the ideal speedup as a horizontal line in the same color as the calculation series on a given number of cores.



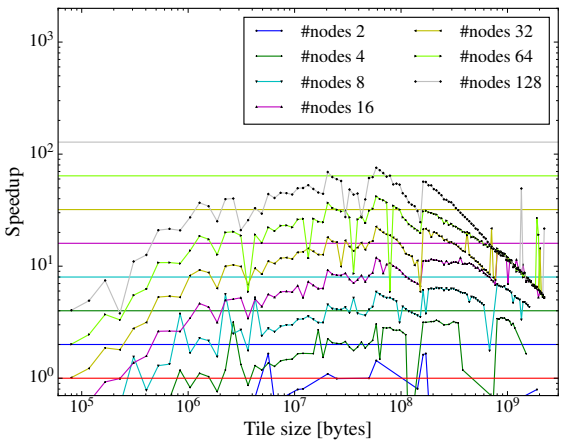
(a) Timings of the contraction carried out with 1 OpenMP thread per node. The local reference contraction with 1 OpenMP thread is given as red horizontal line.



(b) Timings of the contraction carried out with 15 OpenMP threads per node. The local reference contraction with 15 OpenMP threads is given as red horizontal line.



(c) Speedups for the contraction carried out with 1 OpenMP thread per node compared to the reference contraction on a *single node/core*.



(d) Speedups for the contraction carried out with 15 OpenMP threads per node compared to the reference contraction on a *single node*.

bound, to being bound by the transfer rate of the network. The increase in the required time occurs in a region where there are not enough tiles to distribute among all nodes anymore and thus fewer nodes will share the computation. Comparing the flat region to the ideal speedup for the used number of cores for the contraction, shown in Figure 3.4c, shows that almost ideal speedup has been obtained (for all numbers of nodes).

When using 15 OpenMP threads, the situation changes as shown in Figure 3.4b. Here, the `dgemm` and sorting routines benefit from big chunks of data, because a) these routines are in general more efficient when they have to process a lot of data (cache reuse), and b) the threads have to be spawned and shut down many times for small tiles. Thus, there are two competing effects, the more efficient OpenMP parallelization and the number of tiles, which should at least match the number of nodes, resulting in an optimal tile size for a given number of nodes. It is therefore in general less efficient with respect to the performance per core, when OpenMP is used additionally and at least four nodes are needed in order to obtain a speedup for the contraction. On the other hand, when memory distribution is the main goal, it is of advantage that no (extreme) slowdown is associated with the distribution of data (given the right tile size is used).

One way to avoid the decrease in performance loss, which is present for tile sizes from $10^6 - 10^7$ bytes using 15 OpenMP threads (on all numbers of nodes) compared to using one OpenMP thread per node, is to combine a number of preloaded tiles to one larger sub–tensor before calling `dgemm`. Thus, spawning (and shutting down) the treads multiple times will be avoided while using the vendor–optimized `dgemm` for larger chunks of data.

3.3 Conclusion and outlook

In this chapter we developed a strategy for the general handling of tensors in **LSDALTON**. We have furthermore explained the strategy of performing local tensor contractions, given details about how the tiles are obtained from a dense tensor in **LSDALTON** and discussed the distribution of the tiles among the nodes. Furthermore, we have given the general strategy for how tiled tensor operations are performed, and finally we have analyzed the performance of a tiled tensor contraction in detail. Despite the rather good overall performance, we have identified deficiencies when using OpenMP in connection with the current contraction strategy, and given a possible solution. Despite this minor flaw, the tensor framework performs reasonably and its importance in **LSDALTON** may increase, e.g. for avoiding the use of ScaLAPACK, which is sometimes difficult to use.

Chapter 4

Massively parallel CCSD

This section describes work published in the form of a **LSDALTON** *software release [29]*

4.1 Introduction

In the past years much effort has been directed toward making high precision calculations possible for ever larger molecular systems. One way of systematically improving on the precision of an electronic structure calculation is to ascend in the hierarchy of coupled cluster (CC) models, CCSD, CCSD(T), CCSDT, etc. The standard implementation of the CCSD model has a sixth order computational scaling with respect to system size, which severely restricts the application range for the model. To extend the application range standard CCSD implementations were restructured to allow for a parallel execution and today many parallel implementations with greatly extended application range exist [33–36]. In order to be able to treat a wide range of molecules efficiently a flexible conventional CCSD algorithm with increased application range is needed. Even more so, because the CCSD calculation becomes the bottleneck in a CCSD(T) calculation on medium sized molecules if accelerators are used in the (T)-part. In this chapter, we describe an efficient parallel CCSD implementation for a large application range, containing several improvements compared to previous parallel implementations. We also note that such an adaptive CCSD algorithm becomes extremely useful in the divide-expand-consolidate (DEC) framework where CCSD calculations are carried out on fragments of the full molecule.

In Chapter 2, it was shown how the CCSD equations are solved using iterative algorithms (the DIIS and CROP algorithms) and in Chapter 3 it was shown how tensorial quantities may be distributed in order to lower the local memory requirements. Here, we discuss how the most expensive part of an iteration, i.e. the evaluation of the CCSD amplitude equations Eqs. (1.41) and (1.42) for a set of trial parameters, is parallelized making use of the tensors from Chapter 3. In order to evaluate the amplitude equations two fundamentally different strategies have emerged. One strategy is to store the MO integral blocks needed during the evaluation of the amplitude equations on file and to read them whenever required. Even if efficient screening techniques are used and symmetries are exploited the required MO integrals eventually grow very large since the required MOs are of the sizes V^4, V^3O, V^2O^2, VO^3 and O^4, with O and V being the number of occupied and virtual MOs, respectively.

The second strategy avoids the storage of MO integral blocks by recalculating the atomic orbitals (AOs) in each iteration (integral–direct) in batches and evaluating the amplitude equations directly from the AO integral batches. The first strategy is usually faster but it is not very scalable, while the second is scalable but suffers from the recalculation of the AO integrals

in each iteration. Many intermediate approaches have been implemented, which usually avoid to store the largest MO integral blocks V^4 and V^3O by integral–direct techniques and calculate the cheaper terms from MO integral blocks [34–37].

The CCSD vector equations may be expressed as effective CCD equations using a T_1 transformed Hamiltonian [15] as described in Section 1.3.3. The integrals thus become dependent on the cluster parameters and have to be modified in each iteration. Therefore, the T_1 equations are traditionally implemented using an integral–direct formalism. We break with that tradition and formulate a MO–direct approach, where the full MO integrals are stored and T_1 transformed in each iteration. In combination with the AO integral–direct formulation we can set up a switchable algorithm that is highly adaptive to the molecular system. In this way we seek to implement an algorithm that works as efficiently as possible for a broad variety of molecules. This strategy is particularly of advantage for algorithms like the DEC CC [38, 39] technique where a broad variety of differently sized fragment CCSD calculations are run and where the fragment basis $M \gg O + V$.

Among the most successful strategies for the evaluation of the amplitude equations are furthermore a reduction in the overall scaling of the algorithm by reducing the scaling of the most expensive term and avoiding the storage of the V^4 and V^3O integrals. A straight forward implementation of the CCSD vector equations has the computational scaling of V^4O^2, which is the dominant scaling among all the terms of the CCSD vector equations. Saebø et al. [40] and Scuseria *et al.* [41] showed how the computational scaling of this term could be reduced to $\frac{1}{4}V^4O^2$ which has been implemented in an efficient integral density driven loop to scale as $\frac{1}{4}M^4O^2$ by Kobayashi and Rendell [37]. We have introduced a small modification of the AO integral density driven implementation which reduces the scaling to $\frac{1}{4}M^2V^2O^2$ and can be calculated from an AO exchange density distribution only without the need for a coulomb density distribution. Also, the term containing the V^3O integrals is evaluated in an alternative fashion that does not require the V^3O integrals to be stored.

There are different technical possibilities to store the required integral blocks, and parameter vectors on modern computers. While there is a local hard–disk (HD) on most personal computers and nodes on a commodity cluster which usually have the size of a few TB, modern supercomputers feature a file–system (FS) storage, which may provide storage up to several PB. However, HD and FS access is slow compared to memory and network access and if working in the regime of many thousands of nodes even the fastest FSs will not be able to handle the request when a file is accessed by all nodes at the same time. Therefore, we have designed the current implementation to use the parallel distributed memory (PDM) storage provided through the tiled–tensor framework described in Chapter 3.

Previous massively parallel implementations of the coupled cluster equations were written with the assumption that quantities of sizes V^2O^2 can be kept in the memory of a single node. While this requirement still persists for one working array, all the integrals g, amplitudes t and residual vectors Ω of that size can be stored in PDM. The algorithm distributes the memory automatically, depending on the size of the system and the totally available memory in order to retain performance when it is possible. In a future implementation we will remove the necessity for the V^2O^2 working array.

4.2 Evaluation of the CCSD amplitude equations

In this section we will discuss how the different terms of Table 1.2 may be evaluated optimally for a broad variety of molecular system sizes. To facilitate the understanding, the notation is

chosen such that $\mathfrak{p}$ and $\mathfrak{r}$ are used for batched indices, resulting in $N_\mathfrak{p}$ and $N_\mathfrak{r}$ (nonuniform) batches containing $B_\mathfrak{p}$ and $B_\mathfrak{r}$ elements, respectively. The indices $\mathfrak{p}$ and $\mathfrak{r}$ refer generically to an AO or MO basis as discussed in Section 1.3.3.

In Section 4.2.1, we will begin by considering how the equations for the second part of the Ω^{A2} term, can be restructured to reduce the scaling from $\frac{1}{4}M^4O^2$ to $\frac{1}{4}M^2V^2O^2$ in an integral direct implementation and how the resulting algorithm may be used in an MO–direct implementation will be discussed in Section 4.2.2. In Section 4.2.3 we discuss how the storage of the V^3O integral of the singles residual equations may be avoided. In Section 4.3 details about the implementation are discussed, and in Section 4.4 we present numerical results. Finally we conclude this Chapter by giving a summary in Section 4.5.

4.2.1 Treating the sixth order V^4O^2 term in the integral–direct implementation: Ω^{A2}

In this section we describe how the cost of the most expensive term in the CCSD residual equations may be reduced. The scaling of the second term of Ω^{A2} of Table 1.2 (denoted $\Omega^{A2.2}$) involves the contraction of doubles amplitudes with an integral that has four virtual indices. In the limit of a large basis this term therefore becomes the most expensive in the CCSD residual equation. Since this term is symmetric in the contraction pattern, we can use the permutational symmetries of the amplitudes t and integrals $\tilde{g}$ to reduce the cost associated with the most expensive contractions. The computational scaling of the second term in Ω^{B2} of Table 1.2 (denoted $\Omega^{B2.2}$) may be reduced along the same lines as for $\Omega^{A2.2}$ as described in the last part of this section.

Initially we recognize that the $\Omega^{A2.2}$ vector equation contribution may be constructed in two ways

$$\begin{aligned}\Omega^{A2.2}_{aibj} &= \sum_{cd} t^{cd}_{ij}\tilde{g}_{acbd} \\ &= \sum_{\mathfrak{pr}} \Lambda^p_{\mathfrak{p}a}\Lambda^p_{\mathfrak{r}b}\sum_{cd} t^{cd}_{ij}\tilde{g}_{\mathfrak{p}c\mathfrak{r}d} = \sum_{\mathfrak{pr}} \Lambda^p_{\mathfrak{p}a}\Lambda^p_{\mathfrak{r}b}\sigma^{\mathfrak{pr}}_{ij},\end{aligned} \tag{4.1}$$

$$\begin{aligned}\Omega^{A2.2}_{ajbi} &= \sum_{cd} t^{cd}_{ji}\tilde{g}_{acbd} = \sum_{cd} t^{cd}_{ij}\tilde{g}_{bcad} \\ &= \sum_{\mathfrak{pr}} \Lambda^p_{\mathfrak{p}a}\Lambda^p_{\mathfrak{r}b}\sum_{cd} t^{cd}_{ij}\tilde{g}_{\mathfrak{r}c\mathfrak{p}d} = \sum_{\mathfrak{pr}} \Lambda^p_{\mathfrak{p}a}\Lambda^p_{\mathfrak{r}b}\sigma^{\mathfrak{rp}}_{ij},\end{aligned} \tag{4.2}$$

where we have introduced the intermediate $\sigma^{\mathfrak{mn}}_{ij}$

$$\sigma^{\mathfrak{mn}}_{ij} = \sum_{cd} t^{cd}_{ij}\tilde{g}_{\mathfrak{m}c\mathfrak{n}d}. \tag{4.3}$$

In Eq. (4.2) we have exchanged the summation indices c and d and used the symmetry in the amplitudes and integrals. The contribution $\Omega^{A2.2}_{aibj}$ for free indices i, j may therefore be built from $\Omega^{A2.2}_{aibj}$ and $\Omega^{A2.2}_{ajbi}$ with restricted indices $i \leq j$ and thus from $\sigma^{\mathfrak{mn}}_{ij}$ with $i \leq j$ and free $\mathfrak{m}, \mathfrak{n}$.

We will now evaluate $\sigma_{ij}^{\mathfrak{mn}}$ for $i \leq j$ and free $\mathfrak{m}, \mathfrak{n}$ restricting the summation indices to $c \geq d$. First, we rewrite $\sigma_{ij}^{\mathfrak{mn}}$ in Eq. (4.3) as

$$\begin{aligned}\sigma_{ij}^{\mathfrak{mn}} &= \sum_{c>d} t_{ij}^{cd} \tilde{g}_{\mathfrak{m}cnd} + \sum_{c<d} t_{ij}^{cd} \tilde{g}_{\mathfrak{m}cnd} + \sum_{c} t_{ij}^{cc} \tilde{g}_{\mathfrak{m}cnc} \\ &= \sum_{c>d} \left(t_{ij}^{cd} \tilde{g}_{\mathfrak{m}cnd} + t_{ij}^{dc} \tilde{g}_{\mathfrak{m}dnc} \right) + \sum_{c} t_{ij}^{cc} \tilde{g}_{\mathfrak{m}cnc}.\end{aligned} \tag{4.4}$$

Introducing the symmetric and antisymmetric combinations of the amplitudes and integrals

$$t_{ij}^{cd} = \frac{1}{2} \left(t_{ij}^{+cd} + t_{ij}^{-cd} \right), \tag{4.5}$$

$$t_{ij}^{\pm cd} = t_{ij}^{cd} \pm t_{ij}^{dc}, \tag{4.6}$$

$$\tilde{g}_{\mathfrak{m}cnd} = \frac{1}{2} \left(g_{\mathfrak{m}cnd}^{+} + g_{\mathfrak{m}cnd}^{-} \right), \tag{4.7}$$

$$\tilde{g}_{\mathfrak{m}cnd}^{\pm} = \tilde{g}_{\mathfrak{m}cnd} \pm \tilde{g}_{\mathfrak{m}dnc}, \tag{4.8}$$

Eq. (4.4) may be written as

$$\begin{aligned}\sigma_{ij}^{\mathfrak{mn}} =& \sum_{c>d} \left[\frac{1}{2}(t_{ij}^{+cd} + t_{ij}^{-cd}) \frac{1}{2}(\tilde{g}_{\mathfrak{m}cnd}^{+} + \tilde{g}_{\mathfrak{m}cnd}^{-}) + \frac{1}{2}(t_{ij}^{+dc} + t_{ij}^{-dc}) \frac{1}{2}(\tilde{g}_{\mathfrak{m}dnc}^{+} + \tilde{g}_{\mathfrak{m}dnc}^{-}) \right] \\ &+ \sum_{c} \left[\frac{1}{2}(t_{ij}^{+cc} + t_{ij}^{-cc}) \frac{1}{2}(\tilde{g}_{\mathfrak{m}cnc}^{+} + \tilde{g}_{\mathfrak{m}cnc}^{-}) \right].\end{aligned} \tag{4.9}$$

Using (4.6) and (4.8) we obtain

$$\begin{aligned}\sigma_{ij}^{\mathfrak{mn}} =& \sum_{c>d} \left[\frac{1}{2}(t_{ij}^{+cd} + t_{ij}^{-cd}) \frac{1}{2}(\tilde{g}_{\mathfrak{m}cnd}^{+} + \tilde{g}_{\mathfrak{m}cnd}^{-}) + \frac{1}{2}(t_{ij}^{+cd} - t_{ij}^{-cd}) \frac{1}{2}(\tilde{g}_{\mathfrak{m}cnd}^{+} - \tilde{g}_{\mathfrak{m}cnd}^{-}) \right] \\ &+ \sum_{c} \left[\frac{1}{4} t_{ij}^{+cc} \tilde{g}_{\mathfrak{m}cnc}^{+} \right],\end{aligned} \tag{4.10}$$

noting that $t_{ij}^{-cc} = \tilde{g}_{\mathfrak{m}cnc}^{-} = 0$. Only the pure symmetric and antisymmetric combinations, i.e. $t^{+}g^{+}$ and $t^{-}g^{-}$ contribute in the first term of Eq. (4.10). Further, the last term may be included as a diagonal element in the symmetric combination

$$\tilde{t}_{ij}^{+cd} = t_{ij}^{+cd} - \frac{1}{2} \delta_{cd} t_{ij}^{+cd}, \tag{4.11}$$

giving

$$\sigma_{ij}^{\mathfrak{mn}} = \sum_{c \geq d} \left[\frac{1}{2} \tilde{t}_{ij}^{+cd} \tilde{g}_{\mathfrak{m}cnd}^{+} + \frac{1}{2} t_{ij}^{-cd} \tilde{g}_{\mathfrak{m}cnd}^{-} \right] = \frac{1}{2} \left[\sigma_{ij}^{+\mathfrak{mn}} + \sigma_{ij}^{-\mathfrak{mn}} \right], \tag{4.12}$$

where we have introduced

$$\sigma_{ij}^{+\mathfrak{mn}} = \sum_{c \geq d} \tilde{t}_{ij}^{+cd} \tilde{g}_{\mathfrak{m}cnd}^{+}, \tag{4.13}$$

$$\sigma_{ij}^{-\mathfrak{mn}} = \sum_{c \geq d} t_{ij}^{-cd} \tilde{g}_{\mathfrak{m}cnd}^{-}. \tag{4.14}$$

Using Eq. (4.8) we may write

$$\sigma_{ij}^{+\mathfrak{mn}} = \sigma_{ij}^{+\mathfrak{nm}}, \tag{4.15}$$

$$\sigma_{ij}^{-\mathfrak{mn}} = -\sigma_{ij}^{-\mathfrak{nm}}, \tag{4.16}$$

and we may thus obtain

$$\sigma_{ij}^{\mathfrak{nm}} = \frac{1}{2}\left[\sigma_{ij}^{+\mathfrak{mn}} - \sigma_{ij}^{-\mathfrak{mn}}\right]. \tag{4.17}$$

Consequentially, we can construct $\sigma_{ij}^{\mathfrak{mn}}$ with free indices $\mathfrak{m}$, $\mathfrak{n}$ from $\sigma^{\pm}$ with restricted indices $\mathfrak{m} \leq \mathfrak{n}$ using Eqs. (4.12) and (4.17).

We now introduce σ in the CCSD vector equations

$$\Omega_{aibj}^{A2.2} = \sum_{\mathfrak{p}\leq\mathfrak{r}} \Lambda_{\mathfrak{p}a}^{p}\Lambda_{\mathfrak{r}b}^{p}\sigma_{ij}^{\mathfrak{pr}} + \sum_{\mathfrak{p}<\mathfrak{r}} \Lambda_{\mathfrak{r}a}^{p}\Lambda_{\mathfrak{p}b}^{p}\sigma_{ij}^{\mathfrak{rp}}, \tag{4.18}$$

and

$$\Omega_{ajbi}^{A2.2} = \sum_{\mathfrak{p}\leq\mathfrak{r}} \Lambda_{\mathfrak{p}a}^{p}\Lambda_{\mathfrak{r}b}^{p}\sigma_{ij}^{\mathfrak{rp}} + \sum_{\mathfrak{p}<\mathfrak{r}} \Lambda_{\mathfrak{r}a}^{p}\Lambda_{\mathfrak{p}b}^{p}\sigma_{ij}^{\mathfrak{pr}}. \tag{4.19}$$

If $\sigma_{ij}^{\pm\mathfrak{mn}}$ is available for $\mathfrak{m} \leq \mathfrak{n}$ and $i \leq j$ then we may evaluate the individual terms in Eqs. (4.18) and (4.19) giving the contribution for the full $\Omega^{A2.2}$ term.

The dominating computational step for evaluating $\Omega^{A2.2}$ is the construction of $\sigma_{ij}^{\pm\mathfrak{mn}}$ for $i \leq j$ and $\mathfrak{m} \leq \mathfrak{n}$. From Eqs. (4.13) and (4.14) we see that each term requires a $\frac{1}{8}M^2V^2O^2$ transformation step, giving a total scaling of the $\Omega^{A2.2}$ term as $\frac{1}{4}M^2V^2O^2$. The remaining term $\Omega^{A2.1}$ is a conventional four-step transformation without restrictions in the indices and thus fifth order scaling, which will be constructed and integral density driven loop (the integral–direct loop) and added direction to Ω_{aibj} in each iteration.

The $\Omega_{aibj}^{B2.2}$ terms may written as

$$\begin{aligned}\Omega_{aibj}^{B2.2} &= \sum_{kl} t_{kl}^{ab} \sum_{cd} t_{ij}^{cd}\tilde{g}_{kcld} \\ &= \sum_{kl\mathfrak{pr}} t_{kl}^{ab}\Lambda_{\mathfrak{p}k}^{p}\Lambda_{\mathfrak{r}l}^{p}\sigma_{ij}^{\mathfrak{pr}},\end{aligned} \tag{4.20}$$

and may thus be obtained as a byproduct when the $\Omega^{A2.2}$ term is constructed. For the remaining term of Ω^{B2}, intermediates from the previous four-step transformation can be reused and the O^4 integral can be added with index restrictions $i \leq j$ to a transformed version of $\sigma_{ij}^{\mathfrak{pr}}$

$$\Omega_{aibj}^{B2} = \sum_{kl} t_{kl}^{ab}(\tilde{g}_{kilj} + \sigma_{ij}^{kl}). \tag{4.21}$$

4.2.2 Treating the sixth order V^4O^2 term in the MO–direct implementation: Ω^{A2}

If the MO integrals g_{pqrs} are stored, the memory requirement may be lowered by using the symmetry of the indices. Technically it is simplest to use the restrictions of the paired indices

$p \leq q$ and $r \leq s$ when storing the integrals in PDM. However, storing the integrals in this way prohibits the strategy used for the AO integral–direct implementation given in Section 4.2.1. Since it is still possible to use the index restrictions $i \leq j$ of the doubles amplitudes t_{ij}^{cd} when evaluating the $\Omega^{A2.2}$ term using Eqs. (4.1) and (4.2), the MO direct implementation has a scaling of $\frac{1}{2}V^4O^2$. Despite the formally higher prefactor of the $\Omega^{A2.2}$ term using the MO–direct algorithm compared to the AO integral–direct algorithm, the MO-direct implementation is in practice faster for the system sizes it is designed for because the overhead in evaluating the AO integrals is avoided. This will be demonstrated in Section 4.4.

4.2.3 Treatment of the Ω^{E2} contributions and the singles vector equations

In order to avoid storing the V^3O integrals, it the T_1 transformed (AO– and MO–direct) algorithms, we can exploit the similarities in the construction of the Ω^{E2}, the Ω^{A1} and the Ω^{B1} terms. The Ω^{E2} term may be expressed as

$$\begin{aligned}
\Omega_{aibj}^{E2} &= \sum_c t_{ij}^{ac} \left[{}^I\tilde{F}_{bc} - \sum_{\mathfrak{s}} \Lambda_{\mathfrak{s}c}^{h} \sum_{dkl} u_{kl}^{bd} \tilde{g}_{ldk\mathfrak{s}} \right] - \sum_k t_{ik}^{ab} \left[{}^I\tilde{F}_{kj} - \sum_{\mathfrak{p}} \Lambda_{\mathfrak{p}k}^{p} \sum_{cdl} u_{lj}^{cd} \tilde{g}_{\mathfrak{p}dlc} \right] \\
&= \sum_c t_{ij}^{ac} \left[F_{bc} - \sum_{\mathfrak{s}} \Lambda_{\mathfrak{s}c}^{h} H_{b\mathfrak{s}} \right] - \sum_k t_{ik}^{ab} \left[F_{kj} + \sum_{\mathfrak{p}} \Lambda_{\mathfrak{p}k}^{p} G_{\mathfrak{p}j} \right] \\
&= \sum_c t_{ij}^{ac} \left[F_{bc} - H_{bc} \right] - \sum_k t_{ik}^{ab} \left[F_{kj} + G_{kj} \right],
\end{aligned} \tag{4.22}$$

where the intermediates

$$H_{b\mathfrak{s}} = \sum_{dlk} u_{kl}^{bd} \tilde{g}_{ldk\mathfrak{s}} = \sum_{\mathfrak{r}\mathfrak{p}k} u_{k\mathfrak{p}}^{b\mathfrak{r}} \tilde{g}_{\mathfrak{p}\mathfrak{r}k\mathfrak{s}}, \tag{4.23}$$

$$G_{\mathfrak{p}j} = \sum_{cdl} u_{lj}^{cd} \tilde{g}_{\mathfrak{p}dlc} = \sum_{c\mathfrak{r}l} u_{lj}^{c\mathfrak{r}} \tilde{g}_{\mathfrak{p}\mathfrak{r}lc}, \tag{4.24}$$

can be constructed from partially transformed integrals in the batched integral loop for both, the integral– and MO–direct algorithms. The last transformation step as well as the last fifth order contractions are then carried out outside the integral loop. However, the construction of the Ω^{E2} from Eq. (4.22) scales as M^3VO in the construction of the H intermediate in Eq. (4.23) and as M^2VO^2 in the construction of the G intermediate in Eq. (4.24), compared to its evaluation in the MO basis where these parts scale as V^3O^2 and V^2O^3. Despite being considerably more expensive, than in the MO basis, constructing the intermediates scales only as $\mathcal{O}(N^5)$ whereas several other terms in the CCSD algorithm scale as $\mathcal{O}(N^6)$, where N is a measure for the system size. Therefore, we consider the increased scaling through the construction of the H and G intermediates a small price to pay for avoiding the storage of the V^3O integral. Now we explain how the storage of this integral is avoided.

The V^3O integral occurs only in the Ω^{A1}. Due to the structural similarity of this term with the Ω^{E2} we may use the previously constructed intermediate H to obtain the Ω^{A1} contribution. Similarly, we can use the intermediate G for the construction of the Ω^{B1} contribution. A simple exchange of dummy variables shows that $H_{b\mathfrak{s}}$ and $G_{\mathfrak{p}j}$ of Eqs. (4.23) and (4.24) can be used for

constructing both Ω^{A1} and Ω^{B1} in an MVO scaling step

$$\begin{aligned}\Omega^{A1}_{ai} &= 2\sum_{\mathfrak{p}} \Lambda^{p}_{\mathfrak{p}a} \sum_{cdk} u^{cd}_{ki} \tilde{g}_{\mathfrak{p}dkc}, = 2\sum_{\mathfrak{q}} \Lambda^{p}_{\mathfrak{p}a} G_{\mathfrak{p}i}, \\ \Omega^{B1}_{ai} &= -2\sum_{\mathfrak{s}} \Lambda^{h}_{\mathfrak{s}i} \sum_{ckl} u^{ac}_{kl} \tilde{g}_{lck\mathfrak{s}}, = -2\sum_{\mathfrak{s}} \Lambda^{h}_{\mathfrak{s}i} H_{a\mathfrak{s}}.\end{aligned} \tag{4.25}$$

If Ω^{A1} and Ω^{B1} is constructed in the MO basis this requires a V^3O^2 and a V^2O^3 scaling operation. However, the major advantage of constructing the Ω^{A1} and Ω^{B1} contributions in terms of the $H_{b\mathfrak{s}}$ and $G_{\mathfrak{p}j}$ intermediates is that the construction and storage of the V^3O integral may be avoided.

The remaining two sixth order terms Ω^{C2} and Ω^{D2} of Table 1.2 may be evaluated in the MO basis making extensive use of PDM techniques. Therefore, we chose to discuss the construction of the Ω^{C2} and Ω^{D2} in the implementation Section 4.3 rather than in the theoretical discussion.

4.3 Implementation

In the previous section we have given the explicit equations used in the CCSD implementation. In this section, we will detail the implementation of the CCSD amplitude equations and how memory distribution is accomplished. Furthermore, we will discuss, how the AO integral–direct formalism and the MO–direct formalism gave rise to (currently) five different levels of memory distribution. In Section 4.3.1 we will focus on the implementation of the remaining sixth order terms of the CCSD vector equations, i.e. the remaining contraction in the Ω^{B2} term and the Ω^{C2} and Ω^{D2} terms. In Section 4.3.2 we will discuss the difference of the five schemes for evaluating the CCSD vector function.

4.3.1 Work distribution of the Ω^{B2}, Ω^{C2} and Ω^{D2} terms

For the evaluation of the Ω^{B2}, Ω^{C2} and Ω^{D2} terms, when all intermediates may be stored locally on a node, work distribution is achieved by striped matrix multiplications, where the tensors entering these terms are unfolded as matrices. The matrix stripes are chosen such that each node has approximately an equal amount of work, by combining pairs of indices $m_{\mathrm{t}} = V \cdot O$ from $[ai]$, $n_{\mathrm{t}} = V \cdot O$ from $[bj]$, $g_{\mathrm{t}} = V \cdot O$ from $[ck]$, $h_{\mathrm{t}} = V \cdot O$ from $[dl]$, and letting each node work on a stripe of length $m_{\mathrm{p}} = \frac{m_{\mathrm{t}}}{n_{\mathrm{nodes}}}$, where n_{nodes} is the number of nodes (m_{p} is augmented by 1 for the first $\mathrm{mod}\,(m_{\mathrm{t}},\ n_{\mathrm{nodes}})$ nodes, if $\mathrm{mod}\,(m_{\mathrm{t}},\ n_{\mathrm{nodes}}) \neq 0$). On each node, we may then write the carried out contraction in the (unfolded) matrix form exemplified for the Ω^{D2} term as

$$\Omega^{D2}_{m_{\mathrm{p}} n_{\mathrm{t}}} = \frac{1}{2}\sum_{g_{\mathrm{t}}} \left(\tilde{L}_{m_{\mathrm{p}} g_{\mathrm{t}}} + \frac{1}{2}\sum_{h_{\mathrm{t}}} u_{m_{\mathrm{p}} h_{\mathrm{t}}} \tilde{L}_{h_{\mathrm{t}} g_{\mathrm{t}}} \right) u_{g_{\mathrm{t}} n_{\mathrm{t}}}. \tag{4.26}$$

Another possibility is to compute all of these terms from parallel distributed tensors in tiles allowing for streamlined preloading of tiles and computationally overlapping contractions of these tiles using the tiled tensor framework described in Chapter 3. Using the capital indices $A, B, \ldots I, J, \ldots$ for a segment of the whole index range, and letting the small letters be defined

Table 4.1: Summary of the currently implemented schemes in the CCSD algorithm.

scheme	–direct	(main) local storage requirements	tensor contractions	`MPI_ACCUMULATE`
5	MO	$1/4M^4 + 7 \cdot V^2O^2$	Eq. (4.26)	No
4	MO	$1/(4 \cdot n_{\text{nodes}})M^4$ $+7 \cdot V^2O^2$	Eq. (4.26)	Yes
3	AO	$7 \cdot V^2O^2$	Eq. (4.26)	No
2	AO	$4 \cdot V^2O^2$	Eq. (4.26)	Yes
1	AO	$1 \cdot V^2O^2$	Eq. (4.27)	Yes

in the segments $a \in A, b \in B, \ldots i \in I, j \in J, \ldots$ we can write the same contraction as

$$\Omega^{D2}_{AIBJ} = \frac{1}{2}\sum_{CK}\left(\tilde{L}_{AIKC} + \frac{1}{2}\sum_{DL} u^{AD}_{IL}\tilde{L}_{LDKC}\right)u^{BC}_{JK}, \tag{4.27}$$

$$\Omega^{D2}_{aibj} = \frac{1}{2}\sum_{ck}\left(\tilde{L}_{aikc} + \frac{1}{2}\sum_{dl} u^{ad}_{il}\tilde{L}_{ldkc}\right)u^{bc}_{jk}, \tag{4.28}$$

where Eq. (4.27) in contrast to Eq. (4.26) does not require any quantity of the size V^2O^2 to be stored in local memory. On the other hand, if $\tilde{L}_{hg}$ can be stored locally for Eq. (4.26), the work will be well distributed, and no communication is needed. Therefore, Eq. (4.27) is only invoked, if a local storage is prohibited, and only if necessary, the code will switch to use Eq. (4.27).

4.3.2 The five schemes for evaluating the CCSD vector equations

The two strategies discussed previously, i.e. the AO integral–direct strategy, and the MO–direct strategy are implemented in an adaptive hierarchy in the degree of memory distribution. The quantities which are worth distributing are the integrals $\tilde{g}$, the doubles amplitudes t and the doubles residual vectors Ω of sizes V^2O^2. It is necessary to store at least three amplitude and residual vectors of the previous iterations for the CROP solver, and therefore the CROP solver is implemented completely in terms of parallel distributed tensors, cf. Chapters 2 and 3. Therefore, we do not need to take these vectors into account in the current section. An overview over the implemented CCSD schemes is given in Table 4.1

Schemes **5** and **4** use the MO–direct strategy, with the difference that in scheme **5** all computational nodes have the full g_{pqrs} (with the index restrictions discussed in Section 4.2.2) available. In contrast, scheme **4** takes advantage of the parallel distributed tensors introduced in Chapter 3 to distribute the g_{pqrs} among all computational nodes and therefore remote updates are necessary during the transformation to the MO basis. During the iterations, to setup the residual vector each node will only work on the locally stored parts of the g_{pqrs} integral and finally, in order to obtain the residual contribution, the intermediates are added by calling `MPI_REDUCE` (i.e. carrying out the summations over $N_\mathfrak{p}$ and $N_\mathfrak{r}$). For all the sixth order contractions outside the MO–integral loop we use Eq. (4.26) for the work distribution.

Schemes **3**, **2** and **1** are based on the AO–integral direct strategy and therefore, besides the necessary work space, the most memory intensive quantities will be t^{ab}_{ij}, Ω_{aibj} within the residual routine (they are distributed in the solver) and the integrals $\tilde{g}_{iajb}$, $\tilde{g}_{aijb}$ and $\tilde{g}_{abij}$. For scheme **3** all of these quantities are kept in local memory and no communication will be involved for constructing them from a given AO batch, except in the end when the final intermediates will

be obtained by summing over the contributions of all nodes (i.e. summing over all $N_{\mathfrak{p}}$ $N_{\mathfrak{r}}$) by calling MPI_ALLREDUCE. In this way, all the nodes will have the full information necessary for the construction of the Ω^{B2}, Ω^{C2}, Ω^{D2} and Ω^{E2} terms, and the work distribution is obtained by using the strategy given in Eq. (4.26). In scheme **2** only the integrals $\tilde{g}_{iajb}$, $\tilde{g}_{aijb}$ and $\tilde{g}_{abij}$ will be distributed throughout the integral–direct loop, which makes remote updates using MPI_ACCUMULATE necessary (see next subsection and Appendix A). After the integral–direct loop, however, the integrals may be stored locally, since the overall memory requirements are lower, thus we use Eq. (4.26) to obtain work distribution among the nodes. For scheme **1** all the integrals, amplitudes and residual vectors are stored in PDM throughout the evaluation of the vector function. The main memory requirement for scheme **1** is given by one working array, which has the size V^2O^2 due to the way the remote updates are currently implemented. For the construction of the Ω^{B2}, Ω^{C2}, Ω^{D2} and Ω^{E2} terms, the strategy given in Eq. (4.27) is used.

The five schemes, therefore form a hierarchy in their local memory requirements (see Table 4.1) and also they form a hierarchy in computational efficiency, as will be shown in Section 4.4, such that it is simple to choose a scheme for a given molecular system based on the memory requirements. We may therefore choose to use the scheme with the highest memory requirements possible for a given molecular system to obtain reasonable efficiency when solving the CCSD equations for the system. Note that schemes **1** and **2** (**4**) cannot be run on a single node, since they (it) will collapse to scheme **3** (**5**) since no memory distribution is possible.

Remote updates in the integral direct loop

In order to further avoid storage of big quantities throughout the memory–intensive integral direct loop, we (currently) allow for the storage of one V^2O^2 quantity in local memory, and perform remote updates using MPI_ACCUMULATE. This will be exemplified with a pseudo code for the addition of the integral contribution of Ω^{A2}, where N_A, N_I, N_B and N_J are the number of segments in the respective dimensions.

Algorithm 4: Remote update of the integral $\tilde{g}_{aibj}$

```
1: do τ̄ = 1, N_τ
2:   do p̄ = 1,N_p
3:     get g_{p̄q̄τ̄s}
4:     transform to g̃_aibj
5:     do A = 1, N_A
6:       do I = 1, N_I
7:         do B = 1, N_B
8:           do J = 1, N_J
9:             extract g̃_AIBJ
10:            MPI_ACCUMULATE(g̃_AIBJ)
```

An alternative, better way, would be to transform only to segments of the next tile to update and use multiple buffering in order to efficiently have concurrent communication and computation. In a future implementation, this, or a similar technique will remove the need for the V^2O^2 working array in the CCSD residual algorithm.

Table 4.2: Example systems to demonstrate the scalability of the presented CCSD algorithm, n_{nodes} is the number of nodes used in the calculation.

Identifier	System	basis set	M	V	O
Ia	C_2H_5COOH	cc-pVDZ	100	80	20
Ib	C_2H_5COOH	cc-pVTZ	234	214	20
Ic	C_2H_5COOH	cc-pVQZ	455	435	20
II	$C_6H_{13}COOH$	cc-pVQZ	915	879	36
III	$C_{10}H_{21}COOH$	cc-pVTZ	698	646	52

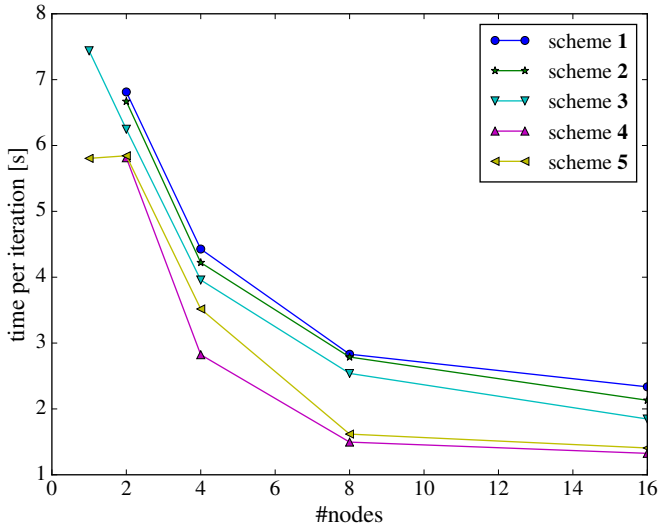
4.4 Numerical results

In this section we report a series of calculations on a set of organic acids of different lengths using Dunning's cc-pVTZ and cc-pVQZ basis sets, where all calculations have been carried out on **Titan** and **Chester** (see page 133), at Oak Ridge National Laboratory, without using the accelerators present on each node. With the calculations presented in this chapter we demonstrate the scalability and adaptivity of the presented algorithm. Furthermore, we demonstrate that despite the fact that the algorithm does not use any local disk or file system storage, and despite the fact that there are only 32GB of main memory per node, it is possible to carry out rather large CCSD calculations. It will also be shown that an algorithm based on `MPI_ACCUMULATE` can achieve reasonable performance if the necessary hard- and software support is present.

The calculations presented here are based on a test set of simple carbonic acids, i.e. propionic acid (**I**), enanthic acid (**II**), and undecylic acid (**III**), and the basis sets cc-pVDZ, cc-pVTZ and cc-pVQZ (see Table 4.2) which allow for systematically increasing the test system and thereby allow for an assessment of the scaling behavior of the algorithm. While the strong scaling behavior of the algorithm may be assessed using any of the shown systems, the weak scaling has not been assessed. The CCSD algorithm consists of a number of steps with different scaling behaviors, where $\frac{1}{4}M^2V^2O^2$ is the dominating one in the limit of large molecules, but for test systems **I** -**III**, other steps may dominate the process, rendering the evaluation of the weak scaling rather cumbersome.

We begin by comparing the AO and MO direct approaches in their different degrees of memory parallelization, schemes $\mathbf{1}-\mathbf{5}$, as introduced in Section 4.3. We have chosen system **Ia**, which was small enough to be run on a single node with schemes **3** and **5**. In Figure 4.1 we compare the average time for one iteration using any of the schemes **1** (blue),**2** (green), **3** (gray), **4** (purple), and **5** (yellow). As expected, the MO–direct approach is faster than the AO–direct approach, and the more distributed the data is, the slower the algorithm, even though the differences are not prominent. That the distributed scheme **4** is faster than the local scheme **5** for the MO–direct algorithm may be a result of the anyways rather short timings. The MO–direct algorithm does not have a speedup, when using 2 nodes instead of 1. This may be attributed to a non–ideal setup concerning the communication using these schemes which becomes a bottleneck for small calculations. The speedup, in general, is not ideal for any of the schemes shown here, since the workload is rather low, considering that there are 8 floating point units per node and up to 16 nodes. In order to be able to better quantify the difference between the schemes, larger systems are necessary, since the computational timings for system **Ia** are too low to be accurate. Since only schemes $\mathbf{1}-\mathbf{3}$ are ported in a reliable way to **Titan**,

Figure 4.1: Comparison of the timings of schemes $\mathbf{1}-\mathbf{5}$ for test system **Ib**, **1** (blue),**2** (green), **3** (gray), **4** (purple), and **5** (yellow)



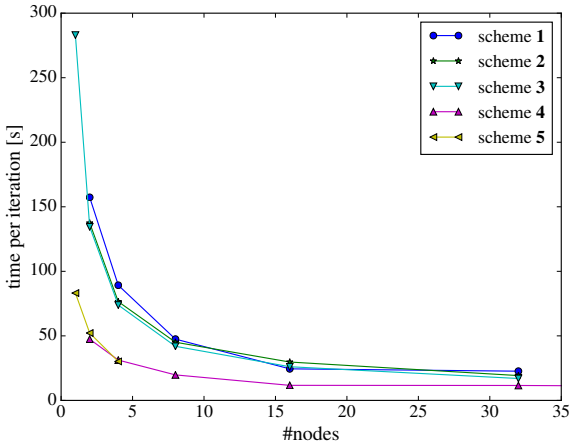
yet, we will reference schemes **4** and **5** only if data were obtainable.

In Figure 4.2 and Figure 4.3 we present timing and speedup data for systems **Ib** and **Ic**, respectively. When comparing the timing for scheme **3** for a single node the time increases by a factor of 38.0 when moving from system **Ia** to **Ib** and 15.2 when moving from system **Ib** to **Ic**. When calculating the respective scaling factors for these steps using $\frac{1}{4}M^2V^2O^2$ we obtain 9.8 and 3.9, respectively. This demonstrates that the algorithm is not dominated by the most expensive term at the considered system sizes, and this is the reason for why schemes **4** and **5** are significantly faster for all the considered systems despite the increased scaling.

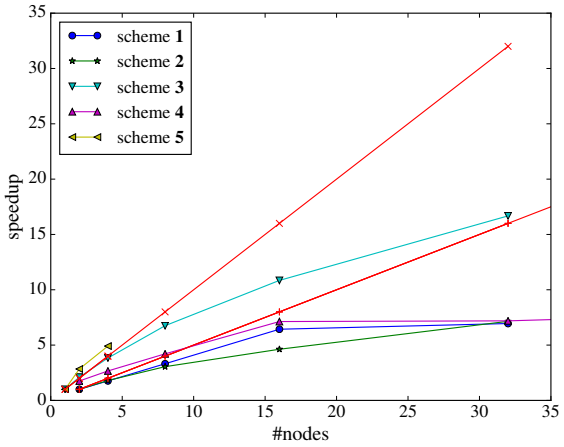
Furthermore, it is reconfirmed from the timings in Figures 4.2a and 4.3a that a) the MO–direct approach is about a factor 2-3 faster than the AO-direct approach, and b) that the memory distribution slows down the code, now with more reliable timings. When calculating the speedup for the individual schemes in Figures 4.2b and 4.3b for systems **Ib** and **Ic**, respectively, we used the lowest number of nodes for a certain scheme as reference. The ideal speedup with respect to that reference is given in red with different markers to make the curves distinguishable. The scaling improves for the larger system, e.g. for scheme **3** and using 32 nodes the speedup is a factor of 16 and 21 for systems **Ib** and **Ic**, respectively. Yet, any of the presented algorithms is rather far from the ideal speedup for large node counts, as both, the work per node decreases, while the associated communication effort increases. For scheme **5** and system **Ib** we observe a super–linear speedup, resulting most probably from the memory distribution in the solver, i.e. there is more memory available during the MO–direct loop and therefore fewer repetitions associated with the calculation of the intermediates.

In Figure 4.4 we have given the timing and the speedup for the largest systems under consideration in this study, i.e. systems **II** and **III**. For these systems, the one (necessary) locally stored tensor occupies about 7.5 and 8.4 GB of local memory (of the 32 GB total on one **Titan** node). We have used 64, 128, 256, 512 and 1024 (16, 32, 64, 128 and 256) nodes for system **II** (**III**), where the timing and scaling develop as expected, when increasing the number of nodes, until time for system **II** (**III**) is increased when using 1024 (256) instead of 512 (128)

Figure 4.2: Comparison of the timings and speedups for schemes **1** (blue),**2** (green), **3** (gray), **4** (purple), and **5** (yellow), and test system **Ib**.

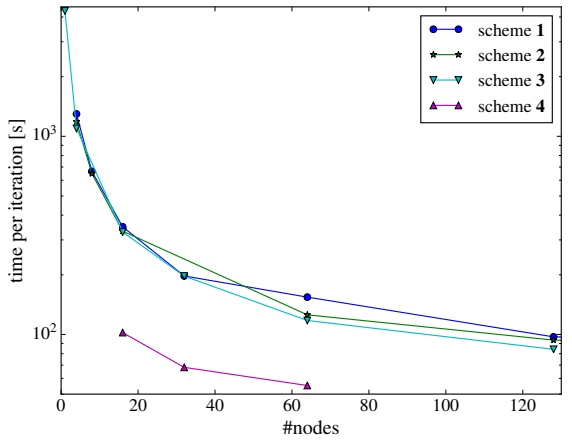


(a) Time per iteration for system **Ib** and schemes **1** − **5**.

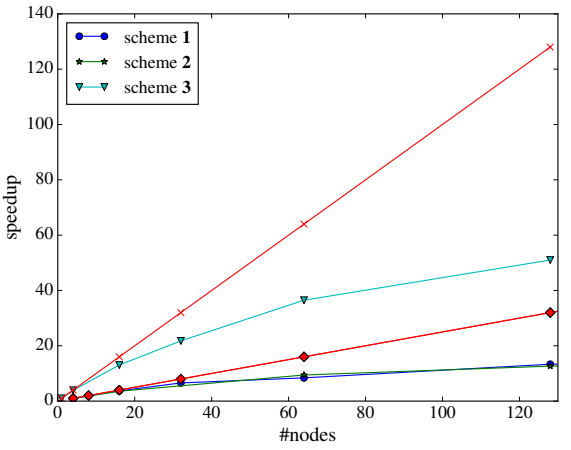


(b) Scaling of the schemes **1** − **5** and system **Ib**. The idealized speedup with 1 (2) node(s) as reference is given in red x (+) to compare with the obtained speedups for schemes **3** and **5** (**1**, **2** and **4**).

Figure 4.3: Comparison of the timings and speedups for schemes **1** (blue),**2** (green), **3** (gray), and **4** (purple, only time) and test system **Ic**.

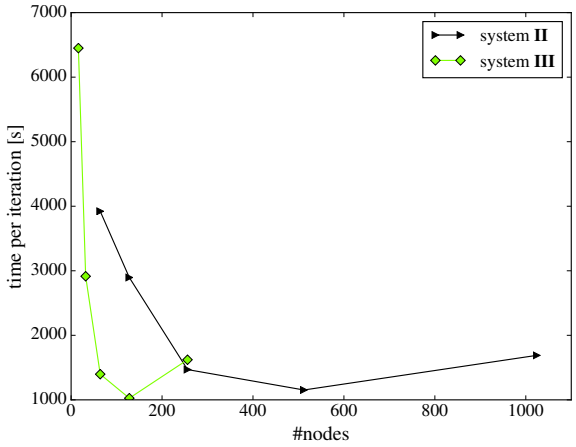


(a) Time per iteration for system **Ic** and schemes **1** − **4**. Note the logarithmic y-axis.

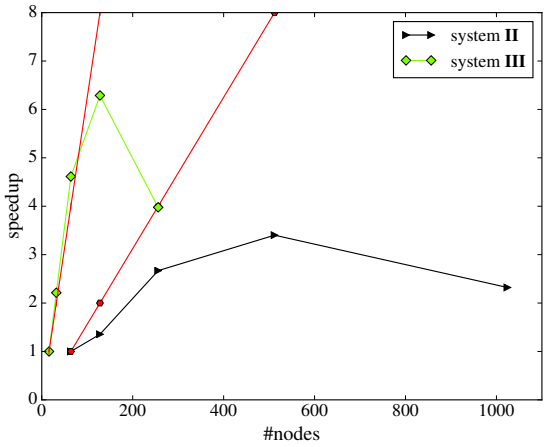


(b) Scaling of the schemes **1** − **3** and system **Ic**. The idealized speedup with 1 (4) node(s) as reference is given in red x (◊) to compare with the obtained speedup for scheme(s) **3** (**1** and **2**). Scheme **4** is omitted for clarity.

Figure 4.4: Comparison of the timings and speedups of scheme **1** for test systems **II** (black) and **III** (yellow), demonstrating the strong scaling behavior of the CCSD algorithm up to 1024 nodes.



(a) Time per iteration for systems **II** and **II** using scheme **2**



(b) Scaling of scheme **1** for systems **II** and **III**. The idealized speedup with 16 (64) node(s) as reference is given in red — (○) to compare with the speedups for systems **II** and **III**.

nodes. System **III** exhibits near perfect scaling, increasing the number of nodes from 16 to 128, indeed, the super linear speedup seen when using 32 and 64 nodes may be ascribed to the surplus of memory available on a single node when distributing the large tensors. The decrease of efficiency beyond 512 (128) nodes for system **II** (**III**) is an artifact of an automatic increase in the number of batches $N_\mathfrak{p}$ and $N_\mathfrak{r}$ which occurs if many more nodes than $N_\mathfrak{p} \cdot N_\mathfrak{r}$ are available. This technique, which is designed for satisfying all nodes with enough work, introduces more overhead, than can be compensated for by the amount of nodes used in the calculation for large node counts. Yet, reducing that splitting will lead to an uneven distribution of work among the nodes, and thus, may not remedy the problem.

4.5 Conclusion and outlook

In this section we have developed a strategy to tackle a broad variety of system sizes when solving the CCSD amplitude equations in an optimal way, with algorithmic schemes that are designed for a minimal memory footprint and absolutely no use of local disks. Because the five schemes are almost hierarchical in the time–to–solution and (reciprocally) in memory use, it is simple to select the scheme, which is the most suitable for the memory situation on a given computer, and thus achieve a short time–to–solution. Because the algorithm is flexible, it is ideally suited to be used in a fragment approach, like the DEC algorithm [38, 39, 42]. We have demonstrated that the algorithm is fairly scalable, and that calculations on molecules with about 900 basis functions are possible with as little as 32 GB of main memory per node. If more memory is available, the application range of the algorithm is drastically increased [43]. We have furthermore identified memory and workload bottlenecks, which are to be removed in a future implementation.

Only recently we could run the larger calculations presented in this section due to the many technical problems we have faced on **Titan**. A combination of a system update and many technical workarounds finally made these large scale calculations possible. As a future perspective we expect, that after removing more technical barriers, the current code should be applicable for even larger molecular systems.

Appendix A

Technical appendix:

A.1 Technical notes for running the current CCSD code with DMAPP and asynchronous progress.

In order to utilize the one–sided capabilities of the **Titan** machine in an optimal way, huge memory pages of 2MB had to be used, furthermore, since **LSDALTON** uses dynamic memory allocation and many allocation/deallocation cycles lead to a fragmentation of memory, a rather large chunk of memory is allocated at application startup and used throughout the calculation (based on an input keyword to **LSDALTON**). Because of the memory demands of a CCSD calculation, and the calculation being rather floating–point intensive, the calculations were run with 8 threads, one per floating–point unit (which is achieved by setting `export OMP_NUM_THREADS=8` and `--cc 0,2,4,6,8,10,12,14,1,3,5,7,9,11,13` to `aprun`) and running with an additional communication thread (`-r 1` to `aprun`).

A note on `MPI_ACCUMULATE`: `MPI_ACCUMULATE` has the downside of not being implemented as a real one–sided routine in most MPI implementations, which has lead application implementors to avoid this routine [36]. While this is true for most commodity clusters, larger clusters nowadays feature Cray's Distributed Memory APPlication (DMAPP) together with MPICH2, which allows for a real one–sided `MPI_ACCUMULATE` use when `export MPICH_RMA_OVER_DMAPP=1` is set at runtime. Furthermore, when used in connection with `export MPICH_NEMESIS_ASYNC_PROGRESS=1` (it is then necessary to use `export MPICH_MAX_THREAD_SAFETY=multiple` at runtime) the application can still achieve reasonable performance as demonstrated in the results section 4.4. In case DMAPP and MPICH2 are available, **LSDALTON** features some optimizations which may be switched on by setting `export LSMPI_ASYNC_PROGRESS=1` at runtime, like a dynamic job distribution scheme taking advantage of advanced MPI3 features.

Chapter 5

Application of the CCSD code: Interaction energies

This chapter summarizes the work of article 3

5.1 Introduction

In biology, functional materials design, and many other disciplines of science, it is often of interest how two molecules A and B interact. The interaction energy ΔE_{AB} is used to characterize the degree of the interaction. While this is a useful measure, it is often difficult to access experimentally and therefore computational methods are often used to predict the interaction energies instead. The interaction energy ΔE_{AB} of two molecules A and B is defined as the energy difference of the combined system AB and the energy of the system, where A and B infinitely far apart. With the standard atom centered basis sets used in most quantum chemistry programs, it is well–known that there is an error associated with the straightforward, uncorrected (UC), calculation of these interaction energies, because the variational freedom of the combined system is larger, than the variational freedom of the individual systems, (usually) leading to interaction energies, which are too negative. In the literature this effect is known as the basis set superposition error (BSSE) [44]. For truncated basis sets, the BSSE is usually corrected for, by using the counterpoise (CP) approach by Boys and Bernardi [45], where the basis of the combined system AB is used to calculate the individual energies of A and B. The difference between the UC and the CP energies approaches zero when the basis set is increased and it vanishes in the limit of a complete basis set (CBS). In the literature it has been extensively discussed to which degree the CP approach is justified [46–58] and other methods for calculating interaction energies have been developed, e.g., the virtual CP (VCP) approach [46, 47], scaled CP corrections [59–61], the semi-empirical geometrical CP approach [62], symmetry-adapted perturbation theory [63, 64], the chemical Hamiltonian approach [65, 66], the strictly monomer-molecular orbital approach [67], and the constrained dimer function approach [68]. Despite this manifold of alternatives, most applications continue to use the CP procedure, and therefore we will not assess these alternatives in detail.

The same number of optimized parameters [43] (SNOOP) approach has recently been developed as an alternative to the UC and CP approaches. The SNOOP scheme was developed with the focus on getting accurate interactions right without the necessity for extrapolation. It is is based on imposing the same number of WF parameters on the system $A + B$ (i.e. the noninteracting monomers) as for the combined system AB. In Ref. 43, calculations were carried

out using the CCSD algorithm of Chapter 4. Here, we give a short summary of the UC, CP and SNOOP methods and reiterate briefly on the results presented in Ref. 43.

In Section 5.2, we summarize the UC, CP and SNOOP approaches for determining intermolecular interaction energies. Section 5.3 contains computational details, and UC, CP, and SNOOP interaction energies at the MP2 and CCSD(T) levels for a selection of test molecules are compared in Section 5.4. Finally, Section 5.5 contains some concluding remarks and perspectives.

5.2 Summary of the UC, CP and SNOOP approaches for calculating interaction energies

For the two isolated monomers A and B with electronic energies E_A and E_B, the electronic interaction energy of the AB dimer, ΔE_{AB}, is defined as the difference between the total electronic energy of AB, E_{AB}, and the total electronic energy of the noninteracting monomers, $E_A + E_B$, i.e.,

$$\Delta E_{AB} = E_{AB} - E_A - E_B, \tag{5.1}$$

where, for simplicity, we have neglected the zero point vibrational energy and the geometry relaxation of the monomer in the interacting system. Thus, the practical determination of ΔE_{AB} involves three separate calculations – one for each of the AB, A, and B systems. If usual atom-centered basis functions (atomic orbitals, AOs) are used on each monomer, obviously, the AB calculation has AOs centered on all atoms of both systems A and B.

When the calculation of E_A (E_B) involves only AOs on monomer A (B) we will refer to the resulting ΔE_{AB} as the *uncorrected* interaction energy. In the following we will use curly brackets to emphasize which basis set was used in each calculation. Thus, the UC interaction energy may be defined as

$$\Delta E_{AB}^{\mathrm{uc}} = E_{AB}^{\{AB\}} - E_A^{\{A\}} - E_B^{\{B\}}. \tag{5.2}$$

Similarly, we may write the interaction energy using the CP approach as

$$\Delta E_{AB}^{\mathrm{cp}} = E_{AB}^{\{AB\}} - E_A^{\{AB\}} - E_B^{\{AB\}}, \tag{5.3}$$

and the BSSE may be defined as the difference between $\Delta E_{AB}^{\mathrm{uc}}$ and $\Delta E_{AB}^{\mathrm{cp}}$. In practice, the CBS result is *usually* somewhere between the $\Delta E_{AB}^{\mathrm{uc}}$ and $\Delta E_{AB}^{\mathrm{cp}}$ energies obtained with truncated basis sets. For variational WFs the energy of the monomers will be lowered in the CP approach compared to the UC approach, and since the reference $E_{AB}^{\{AB\}}$ is the same in both cases, the usual situation is

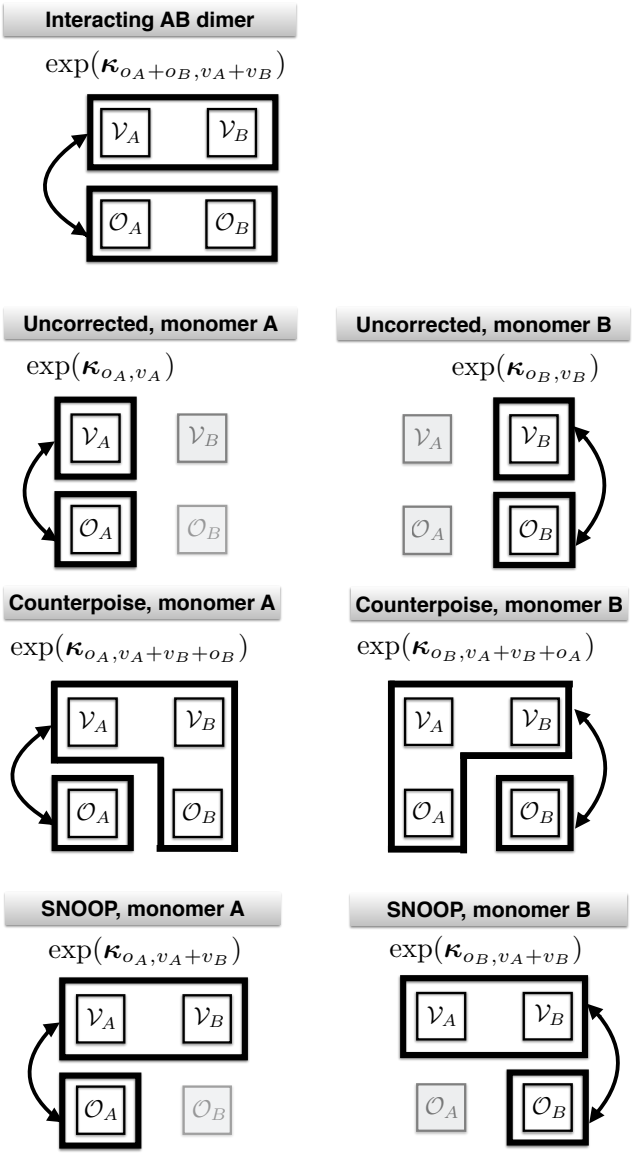
$$\Delta E_{AB}^{\mathrm{cp}} > \Delta E_{AB}^{\mathrm{cbs}} > \Delta E_{AB}^{\mathrm{uc}}, \tag{5.4}$$

which often is the case for non variational WFs, too. An example of this is the MP2/aug-cc-pVTZ interaction energy of the water dimer, for which $\Delta E_{AB}^{\mathrm{uc}} = -8.14$ mE_h and $\Delta E_{AB}^{\mathrm{cp}} = -7.38$ mE_h, while the CBS estimate is $\Delta E_{AB}^{\mathrm{cbs}} = -7.80$ mE_h. Thus, in a sense, the CP approach "over–corrects" for the errors made in the UC approach.

As shown in Eq. (1.31) the total CC energy may be partitioned into a HF part and a correlation part, and thus the interaction energy in Eq. (5.1) may be written as

$$\Delta E_{AB} = \Delta E_{AB,\mathrm{HF}} + \Delta E_{AB,\mathrm{corr}}. \tag{5.5}$$

Figure 5.1: Schematic illustration of the HF optimizations for the interacting AB dimer and the noninteracting $A + B$ monomers using the UC, CP, and SNOOP approaches. The dimensions of the $\boldsymbol{\kappa}$ rotation matrix are indicated as subscripts. $\mathcal{O}_A$ and $\mathcal{V}_A$ ($\mathcal{O}_B$ and $\mathcal{V}_B$) refer to occupied and virtual orbital spaces for monomer A (B). The arrows denote HF orbital optimizations, and grey-shaded orbital spaces are not included in the HF optimizations.



The optimization of the HF WF may be parameterized as in Eq. (1.14) where the number of non–redundant parameters in is OV. We will in the following discussion explicitly distinguish the origins of the orbitals, by giving them the subscript of the system they are associated with, i.e. O_A/V_A (O_B/V_B) denotes the number of occupied/virtual orbitals associated with system A (B), and we may write the dimer HF rotation matrix as $\exp(\boldsymbol{\kappa}_{O_A+O_B,V_A+V_B})$. The total number of parameters in the calculation is therefore

$$P_{AB,\mathrm{HF}} = (O_A + O_B)(V_A + V_B) = O_A(V_A + V_B) + O_B(V_A + V_B). \tag{5.6}$$

Figure 5.1 illustrates the HF optimizations for the interacting AB dimer and for the UC and CP monomer calculations. We use calligraphic letters to denote orbital spaces (e.g. $\mathcal{O}_A$), while capital roman letters (e.g. O_A) are used for denoting the corresponding dimensions. Figure 5.1 shows that the sum of the number of κ parameters in the UC/CP-corrected HF monomer calculations are smaller/larger than the number of parameters in the dimer calculation, i.e.,

$$P^{\mathrm{uc}}_{A+B,\mathrm{HF}} = O_A V_A + O_B V_B, \tag{5.7}$$

and

$$P^{\mathrm{cp}}_{A+B,\mathrm{HF}} = O_A(V_A + V_B + O_B) + O_B(V_A + V_B + O_A). \tag{5.8}$$

Equations (5.7) and (5.8) refer to the number of HF parameters, but similar equations can be derived for the number of CC amplitudes, see Ref. 43. One interpretation of Eqs. (5.7) and (5.8) is that the variational freedom in the UC/CP-corrected monomer calculations is smaller/larger than in the dimer calculation. For a finite basis set the UC/CP monomer energies therefore tend to be too high/low compared to the dimer energy, which explains the commonly observed ordering of interaction energies in Eq. (5.4).

For a better parameter balance between the noninteracting $A + B$ and the interacting AB calculations the SNOOP scheme enforces that the number of optimized parameters in the $A+B$ calculations be equal to the number of optimized parameters in the AB calculation. From Eq. (5.6), we see that this may be achieved by choosing the virtual space of each monomer calculation to be of dimension $V_A + V_B$ as illustrated in Figure 5.1 (bottom). Thus, it follows that

$$P^{\mathrm{snoop}}_{A+B,\mathrm{HF}} = O_A(V_A + V_B) + O_B(V_A + V_B). \tag{5.9}$$

The optimized orbital spaces of the SNOOP monomer HF calculations are then used in subsequent correlated calculations.

It was demonstrated in the Appendix of Ref. 43 that Eqs. (5.7), (5.8), and (5.9) effectively hold also for the CC amplitudes and thus for all parameters entering the CC calculations. We may therefore, in general, write

$$P^{\mathrm{uc}}_{A+B} < P^{\mathrm{snoop}}_{A+B} = P_{AB} < P^{\mathrm{cp}}_{A+B}. \tag{5.10}$$

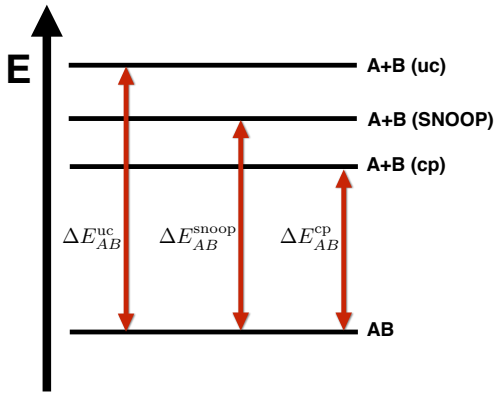
and for strictly variational WFs we thus conclude that

$$E^{\mathrm{uc}}_A + E^{\mathrm{uc}}_B \geq E^{\mathrm{snoop}}_A + E^{\mathrm{snoop}}_B \geq E^{\mathrm{cp}}_A + E^{\mathrm{cp}}_B, \tag{5.11}$$

where the equality signs hold in the CBS limit. For all considered examples in this section, this ordering is also observed for non–variational WFs. Since the same dimer energy E_{AB} is used in the three approaches, it follows from Eq. (5.1) that

$$\Delta E^{\mathrm{uc}}_{AB} \leq \Delta E^{\mathrm{snoop}}_{AB} \leq \Delta E^{\mathrm{cp}}_{AB}. \tag{5.12}$$

Figure 5.2: Energy diagram illustrating the relative positions of UC, SNOOP, and CP noninteracting monomer $A + B$ energies for the situation where $\Delta E_{AB} < 0$.



Equations (5.11) and (5.12) are summarized in Figure 5.2 for the case where $\Delta E_{AB} < 0$. For the common situation in Eq. (5.4), in which the UC interaction energies are too attractive, while the CP-corrected interaction energies are not attractive enough, Figure 5.2 indicates that the SNOOP approach might provide a better balance between the $A + B$ and the AB calculations. A detailed description of the SNOOP algorithm is given in the chart on page 58.

5.3 Computational details

We consider a test set consisting of the 8 molecular dimers in Figure 5.3, i.e., the water dimer, the hydrogen fluoride dimer, the formamide dimer, the methane dimer, the benzene–hydrogen sulfide complex, and the sandwich (S), T-shaped, and parallel displaced (PD) conformations of the benzene dimer. This test set ranges from molecular systems where the interactions are dominated by hydrogen bonds (e.g. the water dimer) to systems where dispersion forces are dominating (e.g. the methane dimer).

All geometries are equilibrium structures, and the monomer geometries are the same for the isolated monomers and for the dimer complexes. The reference interaction energies are presented in Table 5.1. For details on the molecular geometries and reference interaction energies see Ref. 43.

We have carried out MP2 and CCSD(T) calculations using the aug-cc-pVXZ (X=D,T,Q) basis sets [28]. The HF gradient norm was converged to 10^{-6} a.u., and the CCSD residual norm was converged to 10^{-9} a.u. All calculations used the frozen core approximation and were carried out using with the CCSD implementation described in Chapter 4, implemented in the **LSDALTON** program [29]. For the complexes containing benzene the diffuse functions were omitted on hydrogen atoms. The (T)/aug-cc-pVQZ calculations were too large to be carried out for the benzene dimers with our present code (30 occupied and 1278 virtual orbitals).

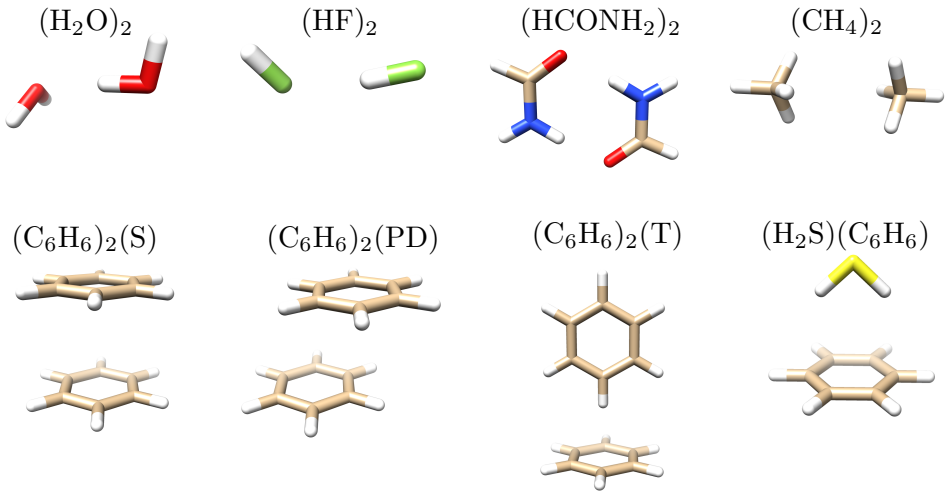
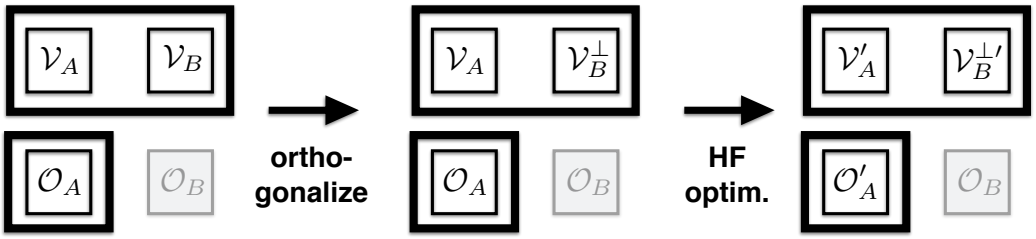


Figure 5.3: Molecular dimers considered in this study.

1. Carry out UC HF optimizations for systems A and B to obtain occupied and virtual spaces $\mathcal{O}_A$ and $\mathcal{V}_A$ for monomer A, and $\mathcal{O}_B$ and $\mathcal{V}_B$ for monomer B.

2. For system A, orthogonalize $\mathcal{V}_B$ orbitals against the $\mathcal{O}_A$ and $\mathcal{V}_A$ orbitals to generate orthogonalized $\mathcal{V}_B^\perp$ orbitals, and carry out HF optimization in the $\{\mathcal{O}_A, \mathcal{V}_A, \mathcal{V}_B^\perp\}$ space to calculate $E_{A,\mathrm{HF}}^{\mathrm{snoop}}$:



3. Carry out the correlated calculation for system A using optimized occupied $\mathcal{O}'_A$ and virtual $\mathcal{V}'_A \cup \mathcal{V}_B^{\perp\prime}$ orbitals to determine $E_{A,\mathrm{corr}}^{\mathrm{snoop}}$.

4. Calculate the SNOOP CC energy for monomer A:

$$E_A^{\mathrm{snoop}} = E_{A,\mathrm{HF}}^{\mathrm{snoop}} + E_{A,\mathrm{corr}}^{\mathrm{snoop}}. \tag{5.13}$$

5. Carry out the equivalents of steps 2, 3, and 4 for system B.

6. Carry out conventional AB dimer calculation (HF+correlation) in the total orbital space to determine the dimer energy E_{AB}.

7. Calculate the SNOOP interaction energy:

$$\Delta E_{AB}^{\mathrm{snoop}} = E_{AB} - E_A^{\mathrm{snoop}} - E_B^{\mathrm{snoop}}. \tag{5.14}$$

Table 5.1: Estimated MP2 and CCSD(T) CBS interaction energies (mE_h) for the molecular dimers in Figure 5.3.

System	MP2	CCSD(T)
$(H_2O)_2$	-7.80	-7.91
$(HF)_2$	-7.15	-7.35
$(HCONH_2)_2$	-25.1	-25.4
$(CH_4)_2$	-0.80	-0.86
$(C_6H_6)_2$ (S)	-5.50	-2.63
$(C_6H_6)_2$ (PD)	-7.82	-4.25
$(C_6H_6)_2$ (T)	-5.77	-4.29
$(H_2S)(C_6H_6)$	-5.76	-4.51

5.4 Results

5.4.1 Comparison of the UC, CP-corrected, and SNOOP schemes

For a given basis set, we consider the relative error of the interaction energy, Δ,

$$\Delta = \frac{\Delta E^{\text{method}} - \Delta E^{\text{cbs}}}{|\Delta E^{\text{cbs}}|} \cdot 100\%, \tag{5.15}$$

where ΔE^{method} is either the UC (Eq. (5.2)), the CP-corrected (Eq. (5.3)), or the SNOOP (Eq. (5.14)) interaction energy, and ΔE^{cbs} is the interaction energy at the estimated CBS limit (Table 5.1). For the MP2 and CCSD(T) models, $\Delta E^{\text{cbs}} < 0$ for all molecules in Figure 5.3, and a negative (positive) Δ thus implies that the calculated interaction energy is too attractive (not attractive enough). We also consider the mean error, $\bar{\Delta}$, the absolute mean error, $\bar{\Delta}_{\text{abs}}$, the standard deviation, σ, and the maximum absolute error, Δ_{max}. The statistical error measures are presented in Tables 5.2 and 5.3, respectively, while the relative MP2 and CCSD(T) errors for the individual molecular systems are given in Tables 5.4 and 5.5, respectively. The normal distribution of the MP2 errors is given in Figure 5.4.

With the exception of some of the CP σ values, the statistical measures in Tables 5.2 and 5.3 are significantly smaller for the SNOOP scheme than for the two other schemes for both the MP2 and CCSD(T) models and regardless of the basis set, see also Figure 5.4. These findings confirm that it is important to use a balanced number of parameters in the monomer and dimer calculations. The relatively small σ values for the CP scheme in Tables 5.2 and 5.3 show that the CP errors are quite systematic (see Section 5.4.2).

Tables 5.4 and 5.5 collectively show that the UC approach yields interactions energies that are too attractive ($\Delta < 0$) for all dimers of the test set, while the corresponding CP values are not attractive enough ($\Delta > 0$) in agreement with Eq. (5.4). The SNOOP errors for the individual molecules are in all cases – with the exception of $(C_6H_6)_2$ (S)/aug-cc-pVTZ – smaller than the corresponding CP errors. Smaller errors are in some cases observed for the UC approach, but this approach suffers from large errors for, e.g., the benzene and methane dimers, illustrating that the UC interaction energies are quite unreliable. In accordance with the monomer energy ordering in Figure 5.2 the SNOOP scheme always yields errors that are sandwiched between the UC and CP-corrected errors, which is why the SNOOP in general leads to smaller errors than the UC and CP approaches. These findings in connection with Figure 5.2 also explain why an

Table 5.2: Statistical errors for relative MP2 interaction energies (%) for the set of test molecules.

Measure	Basis	Uncorrected	CP	SNOOP
$\bar{\Delta}$	aug-cc-pVDZ	-30.0	16.6	4.4
	aug-cc-pVTZ	-13.1	6.6	1.2
	aug-cc-pVQZ	-5.2	2.7	1.1
$\bar{\Delta}_{\mathrm{abs}}$	aug-cc-pVDZ	30.0	16.6	4.5
	aug-cc-pVTZ	13.1	6.6	1.8
	aug-cc-pVQZ	5.2	2.7	1.1
σ	aug-cc-pVDZ	30.4	4.3	4.2
	aug-cc-pVTZ	9.5	1.1	2.0
	aug-cc-pVQZ	3.0	0.6	0.7
Δ_{max}	aug-cc-pVDZ	83.8	21.7	11.3
	aug-cc-pVTZ	25.1	8.0	3.4
	aug-cc-pVQZ	8.6	3.6	2.4

Table 5.3: Statistical errors for relative CCSD(T) interaction energies (%) for the set of test molecules.

Measure	Basis	Uncorrected	CP	SNOOP
$\bar{\Delta}$	aug-cc-pVDZ	-40.2	17.7	2.1
	aug-cc-pVTZ	-17.7	6.5	0.1
$\bar{\Delta}_{\mathrm{abs}}$	aug-cc-pVDZ	40.2	17.7	7.1
	aug-cc-pVTZ	17.7	6.5	3.1
σ	aug-cc-pVDZ	38.9	4.4	8.3
	aug-cc-pVTZ	16.4	1.8	3.8
Δ_{max}	aug-cc-pVDZ	85.1	25.0	13.6
	aug-cc-pVTZ	46.8	10.4	5.4

empirical scaling of the CP correction in many cases gives improved results over the standard CP approach [59–61]. Both positive and negative SNOOP errors are observed. We elaborate on this issue in Section 5.4.2.

5.4.2 Convergence with basis set and extrapolation

In Ref. 43 it was discussed why the SNOOP scheme is not suitable for basis set extrapolations, e.g., the X^{-3} extrapolation scheme [69] of Halkier *et al.* Here we reiterate on the numerical results, which support that SNOOP interaction energies obtained using a given basis are of similar quality as those determined by basis set extrapolation of CP results obtained at a similar computational cost.

To analyze the components of the interaction energy in greater detail, we separate the total interaction energy errors into HF and correlation contributions. For brevity, we limit the analysis to the water and the PD benzene dimers as representative examples of molecular complexes dominated by hydrogen bonding and dispersion interactions, respectively. Furthermore, we consider only MP2 results since the CCSD(T) results are similar.

The results are presented in Table 5.7. We recognize that the correlation errors dominate over the HF errors in most cases, and the general superiority of the SNOOP scheme over the UC and CP approaches is caused primarily by the comparatively small errors in the correlation

Table 5.4: Relative errors Δ (%) of UC, CP-corrected, and SNOOP MP2 interaction energies for the test molecules.

System	Basis	Uncorrected	CP	SNOOP
$(H_2O)_2$	aug-cc-pVDZ	-5.3	12.2	6.6
	aug-cc-pVTZ	-4.3	5.5	2.6
	aug-cc-pVQZ	-2.9	2.0	1.1
$(HF)_2$	aug-cc-pVDZ	-3.1	11.4	7.0
	aug-cc-pVTZ	-4.5	5.8	2.7
	aug-cc-pVQZ	-3.7	1.8	0.7
$(HCONH_2)_2$	aug-cc-pVDZ	-1.9	12.0	7.2
	aug-cc-pVTZ	-2.6	5.2	3.4
	aug-cc-pVQZ	-1.4	2.3	1.8
$(H_2S)(C_6H_6)$	aug-cc-pVDZ	-5.2	19.8	11.3
	aug-cc-pVTZ	-6.0	8.0	3.4
	aug-cc-pVQZ	-2.1	3.6	2.4
$(C_6H_6)_2$ (S)	aug-cc-pVDZ	-47.4	18.0	2.6
	aug-cc-pVTZ	-25.1	6.9	-0.4
	aug-cc-pVQZ	-8.3	3.1	0.5
$(C_6H_6)_2$ (T)	aug-cc-pVDZ	-46.2	21.4	1.1
	aug-cc-pVTZ	-20.1	7.9	-0.1
	aug-cc-pVQZ	-7.2	3.4	1.4
$(C_6H_6)_2$ (PD)	aug-cc-pVDZ	-46.8	16.3	-0.3
	aug-cc-pVTZ	-20.9	6.5	-1.2
	aug-cc-pVQZ	-7.2	2.9	0.9
$(CH_4)_2$	aug-cc-pVDZ	-83.8	21.7	-0.2
	aug-cc-pVTZ	-21.1	6.9	-0.5
	aug-cc-pVQZ	-8.6	2.7	0.4

Figure 5.4: Normal distributions of the MP2 errors for UC, CP-corrected, and SNOOP interaction energies for the aug-cc-pVDZ (bottom), aug-cc-pVTZ (middle), and aug-cc-pVQZ (top) basis sets.

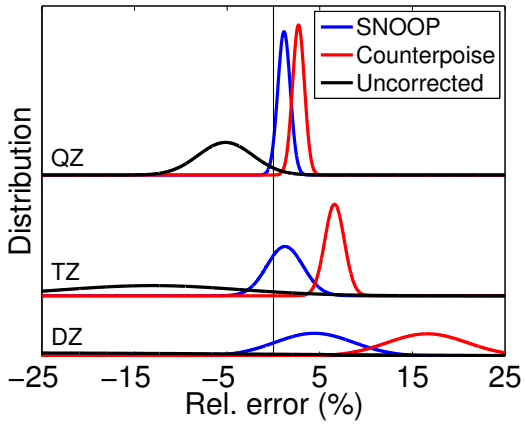


Table 5.5: Relative errors Δ (%) of UC, CP-corrected, and SNOOP CCSD(T) interaction energies for the test molecules.

System	Basis	Uncorrected	CP	SNOOP
$(H_2O)_2$	aug-cc-pVDZ	-4.8	13.9	7.8
	aug-cc-pVTZ	-3.9	5.8	2.9
	aug-cc-pVQZ	-1.9	2.1	1.4
$(HF)_2$	aug-cc-pVDZ	-3.0	13.2	8.1
	aug-cc-pVTZ	-4.3	6.4	3.0
	aug-cc-pVQZ	-2.8	2.1	1.1
$(HCONH_2)_2$	aug-cc-pVDZ	-2.3	12.2	7.1
	aug-cc-pVTZ	-3.3	4.2	2.4
	aug-cc-pVQZ	-1.9	1.0	0.6
$(H_2S)(C_6H_6)$	aug-cc-pVDZ	-8.5	25.0	13.6
	aug-cc-pVTZ	-7.1	10.4	4.5
	aug-cc-pVQZ	-0.9	4.8	3.6
$(C_6H_6)_2$ (S)	aug-cc-pVDZ	-85.1	20.3	-7.0
	aug-cc-pVTZ	-46.8	5.1	-5.4
$(C_6H_6)_2$ (T)	aug-cc-pVDZ	-58.1	17.5	-3.2
	aug-cc-pVTZ	-22.4	7.1	-1.5
$(C_6H_6)_2$ (PD)	aug-cc-pVDZ	-83.6	18.5	-9.6
	aug-cc-pVTZ	-35.6	6.8	-5.1
$(CH_4)_2$	aug-cc-pVDZ	-76.4	20.6	-0.4
	aug-cc-pVTZ	-18.2	6.2	-0.3
	aug-cc-pVQZ	-5.4	2.6	1.0

Table 5.6: Statistical errors (%) for SNOOP/aug-cc-pVTZ and extrapolated DZ-TZ CP interaction energies where the latter were obtained using the X^{-3} extrapolation scheme [69].

Model	Scheme	$\bar{\Delta}$	$\bar{\Delta}_{\mathrm{abs}}$	σ	Δ_{max}
MP2	SNOOP (TZ)	1.2	1.8	2.0	3.4
	CP (DZ-TZ)	2.1	2.1	1.3	3.7
CCSD(T)	SNOOP (TZ)	0.1	3.1	3.8	5.4
	CP (DZ-TZ)	1.7	1.8	1.2	3.5

part of the SNOOP interaction energy. In addition, the extraordinarily small SNOOP error for the PD benzene dimer using the aug-cc-pVDZ basis is seen to result in part from a fortuitous cancellation of a negative HF error and a positive correlation error.

From Table 5.7 it is clear that the SNOOP interaction energies oscillate slightly when the basis set is increased, as illustrated by the convergence of the HF contribution for $(H_2O)_2$ and the correlation contribution for $(C_6H_6)_2$ (PD). On the other hand, the CP interaction energies display a stable convergence with basis set for the HF as well as the correlation contribution. CP interaction energies obtained with basis sets of increasing cardinal numbers can therefore by used to extrapolate the interaction energy to the CBS limit [69], while such extrapolations would be quite unstable for the SNOOP scheme.

It is explained in detail in Ref [43], why the SNOOP scheme is not suited for the extrapolation with the basis set, and we will not go into that level of detail here. We will rather continue with pursuing the very relevant question of whether the SNOOP interaction energy calculated with a given basis set like aug-cc-pVTZ is closer to the CBS value than the corresponding CP interaction energy obtained by extrapolating aug-cc-pVDZ and aug-cc-pVTZ results (henceforth referred to as DZ-TZ extrapolation). These two approaches have roughly the same computational cost, since an aug-cc-pVDZ calculation is relatively inexpensive compared to an aug-cc-pVTZ calculation, and since the SNOOP and CP calculations using the aug-cc-pVTZ basis cost roughly the same.

Table 5.6 shows that the SNOOP/aug-cc-pVTZ interaction energies are of similar quality as the extrapolated CP interaction energies. The SNOOP approach is therefore an attractive alternative to extrapolating CP interaction energies. For application purposes it may be advantageous to use both approaches and compare the results to validate the interaction energy of interest. For example, a large difference between SNOOP/TZ results and extrapolated CP/DZ-TZ results would be a strong indication that it is necessary to increase the basis set to obtain a reliable interaction energy.

Finally, we note that most of the reference data in Table 5.1 are based on TZ-QZ extrapolations of CP interaction energies. However, a generalization of the results in Table 5.6 to higher cardinal numbers might indicate that the TZ-QZ extrapolated CP interaction energies in fact contain errors of the same magnitude as the SNOOP/aug-cc-pVQZ results. It is thus possible that the SNOOP/aug-cc-pVQZ errors in Tables 5.4 and 5.5 in reality reflect the accuracy of the reference data and not the inherent accuracy.

Table 5.7: Relative interaction energy errors Δ (%) for UC, CP-corrected, and SNOOP MP2 interaction energies partitioned into HF and correlation contributions for the water dimer and the PD benzene dimer.

System	Method	Basis	Hartree–Fock	Correlation	Total
$(H_2O)_2$	Uncorrected	aug-cc-pVDZ	-4.6	-0.7	-5.3
		aug-cc-pVTZ	-0.8	-3.5	-4.3
		aug-cc-pVQZ	-0.6	-2.2	-2.9
	CP	aug-cc-pVDZ	0.5	11.7	12.2
		aug-cc-pVTZ	0.8	4.7	5.5
		aug-cc-pVQZ	0.0	2.0	2.0
	SNOOP	aug-cc-pVDZ	-0.7	7.3	6.6
		aug-cc-pVTZ	0.5	2.2	2.6
		aug-cc-pVQZ	-0.1	1.2	1.1
$(C_6H_6)_2$ (PD)	Uncorrected	aug-cc-pVDZ	-20.0	-26.8	-46.8
		aug-cc-pVTZ	-4.6	-16.3	-20.9
		aug-cc-pVQZ	-1.2	-6.0	-7.2
	CP	aug-cc-pVDZ	-0.1	16.4	16.3
		aug-cc-pVTZ	0.0	6.4	6.5
		aug-cc-pVQZ	0.0	2.9	2.9
	SNOOP	aug-cc-pVDZ	-5.8	5.5	-0.3
		aug-cc-pVTZ	-1.1	-0.1	-1.2
		aug-cc-pVQZ	-0.1	1.0	0.9

5.5 Conclusion and perspectives

We have presented the calculations, which were used to back up the development of the SNOOP scheme for the calculation of intermolecular interaction energies in Ref. [43]. All these calculations employed the algorithm described in Chapter 4

Numerical results for the aug-cc-pVXZ (X=D,T,Q) basis sets show that the SNOOP scheme which, per construction, is sandwiched between the UC and CP approaches, outperforms both of them for the calculation of MP2 and CCSD(T) interaction energies for a selection of test molecules, indicating that the parameter balance is indeed very important for practical calculations.

Furthermore, SNOOP interaction energies calculated using a given basis are of similar quality as those determined by basis set extrapolation of CP results obtained at a similar computational cost. The SNOOP approach is therefore an attractive alternative to extrapolating CP interaction energies. For application purposes, it may be particularly advantageous to use both approaches and compare the results in order to validate the interaction energy of interest.

Part II

The Divide-Expand-Consolidate method revisited

Introduction

Highly accurate calculations of molecular energies and properties have been feasible for many years using the systematically improvable coupled cluster (CC) hierarchy of methods – MP2 (second order Møller-Plesset perturbation theory) [11], CCSD (coupled-cluster with singles and doubles excitations) [12], CCSD(T) (coupled-cluster with single and double excitations and an approximate treatment of triple excitations) [13], CCSDT (coupled-cluster with singles, doubles and triples excitations) [14], etc. In conventional implementations the computational scaling of this hierarchy of models is N^5, N^6, N^7, N^8, etc. where N is a measure of the size of the system. While the CC hierarchy can routinely be used to calculate molecular energies and properties within accuracies that challenge modern experiments, the CC methods have in general been restricted to relatively small molecules due to their inherent scaling wall. With the growing capabilities of experimental methods and advances in life and material sciences to address ever more complex chemical systems, the need for highly accurate electronic structure calculations for large molecules is growing rapidly.

The grotesque scaling problems of the CC methods is a direct consequence of the nonlocal nature of the conventionally used canonical Hartree-Fock (HF) orbitals which was realized early [70], and it was recognized that the use of local orbitals would allow for an accurate linear–scaling description of the electronic correlation effects. Local correlation methods were pioneered by Pulay [70] and an early prominent contribution is the local coupled cluster (LCC) method of Werner and coworkers [71–74]. Many other local CC methods have been proposed, e.g. the atomic orbital-based CC, the natural linear scaling approach, the cluster-in-a-molecule approach, the divide-and-conquer approach, the fragment molecular orbital approach and the incremental scheme. In this existing plethora of local CC methods [75–97] *ad hoc* approximations have often been introduced, for example by assigning fixed virtual correlating orbital spaces to local occupied orbitals, or by a physical fragmentation of the molecule. These approximations make the precision of a local correlated calculation, compared to a standard calculation, unclear.

In the standard CC methods the precision of a calculation is determined by the residual norm threshold for the cluster amplitude equations (see Part I). The precision is thus defined prior to a calculation. This *a priori* knowledge of the precision is an important feature and one of the reasons for the success of CC calculations on small molecular systems. In the recently introduced divide-expand-consolidate (DEC) [38,39,98] local CC method the precision is similarly defined prior to a calculation. In the DEC method a calculation on a full molecular system is divided into a sequence of single fragment and pair fragment calculations referencing small parts of the orbital space of the full molecular system and the precision is defined by the fragment optimization threshold (FOT) that is imposed on the fragment energy calculations. The FOT identifies the orbital spaces where the cluster amplitude equations have to be solved to give the fragment energies to the desired precision. The FOT based error control at the single fragment level ultimately leads to an error in the total correlation energy, which is determined

by the FOT. The recently developed local orthogonal orbitals for both the occupied and virtual orbital spaces [99–101] constitute the foundation for the Divide-Expand-Consolidate DEC–CC method. In this part of the present work, all quantities are therefore given in this set of local orbitals, unless stated otherwise.

We will begin this second part with Chapter 6, where we will describe how the orbital spaces for the single fragment calculations may be determined in MP2, CCSD and CCSD(T) calculations to obtain the final correlation energy to within a predefined precision. In Chapter 7, we will refactor the DEC algorithm for massively parallel execution and demonstrate the overall scalability of the algorithm for MP2 and CCSD(T) calculations and in Chapter 8 we will investigate the possibility of using the pair–natural orbital approach in a DEC–CCSD context in order to reduce the cost of the overall calculation.

Chapter 6

A locality analysis of the DEC–CCSD(T) method

This chapter summarizes the work of articles 4 and 5

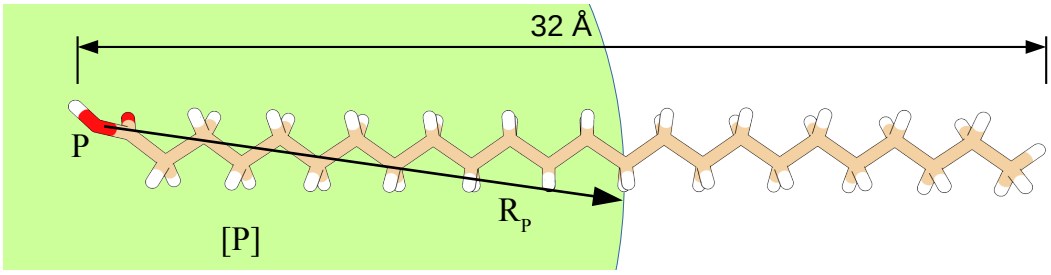
Introduction

DEC is a local correlation method where the inherent locality of the electron correlation problem is efficiently exploited to express the correlated wave function calculation on the full molecular system in terms of many small and independent fragment calculations utilizing subsets of the total orbital space. Importantly, the local orbital spaces employed in the fragment calculations may be determined in a black box manner during the calculation to ensure that the calculated properties are determined to a predefined precision compared to a conventional calculation. A DEC calculation may be described both in terms of an occupied and a virtual partitioning of the correlation energy. Using these different partitioning schemes, it is possible to validate the calculated correlation energy using internal consistency checks and to exploit locality when evaluating molecular properties [38, 102].

We have previously carried out a locality analysis for the MP2 and CCSD equations in terms of local orbitals using the occupied partitioning scheme [39] which demonstrates how orbital spaces may be selected for evaluating the single fragment energies. Here, we simplify and generalize this analysis to include the virtual space partitioning scheme. The connection between the single fragment energy contributions and the selected orbital spaces is also examined. For MP2, orbital space extensions arise solely due to a non–vanishing Fock matrix coupling mechanism, while for CCSD orbital space extensions may additionally occur via a mechanism involving long–range interactions between charge distributions, and a mechanism describing long–range polarization effects.

In Section 6.1 we summarize the DEC algorithm, and in Section 6.2 we describe how orbital spaces propagate in an iterative algorithm when the MP2 and CCSD amplitude equations are solved and identify the structure of the orbital spaces that have to be used for evaluating the single fragment energies. In Section 6.3 we use this information to develop the practical algorithm that may be used to determine the single fragment energies to the FOT precision.

Figure 6.1: All numerical illustrations in Sections 6.1 and 6.2 refer to this simple test system (lignoceric acid, 6-31G basis), unless stated otherwise. The atomic site P under consideration is the protonated oxygen atom, and the radius R_P defines which orbitals are included in the $[P] = [\underline{P}] \cup [\overline{P}]$ space.



6.1 The DEC algorithm summarized

In this subsection we shortly summarize the DEC–CC method with emphasis on giving the background that is required for determining the orbital spaces where the cluster amplitude equations have to be solved to give the single fragment energies to a predefined tolerance, the FOT.

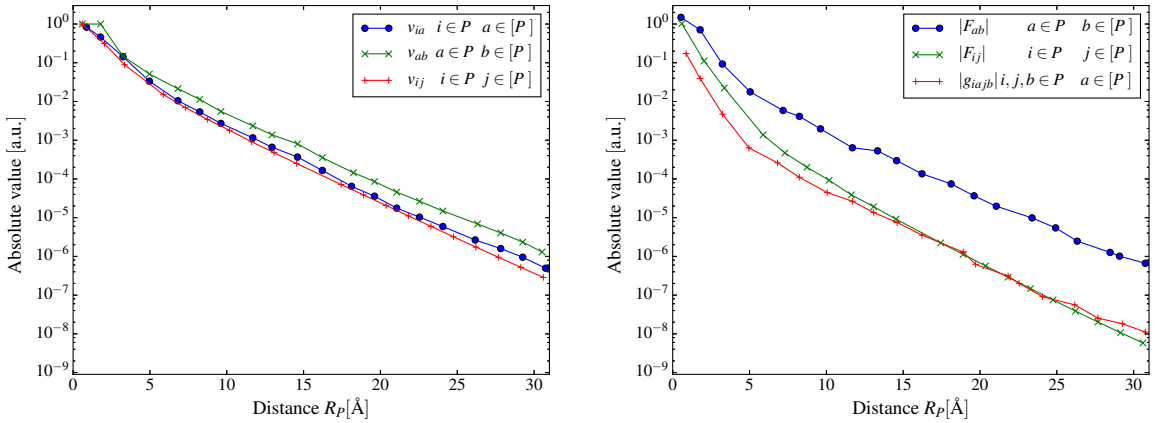
6.1.1 Molecular integrals in a local orbital basis

The CC correlation energy for a closed–shell system may be expressed as given in Eq. (1.33), where, for a set of local HF orbitals, each orbital may be assigned to a site $P, Q, \ldots$. The generic term "site" may stand for, e.g., an orbital site, an atomic site, or a molecular site, in the following analysis, where the locality arguments will hold, irrespective of the choice of site. In practice, however, atomic sites have proven to be a solid choice. The set of occupied (virtual) orbitals assigned to site P will be denoted $\underline{P}$ ($\overline{P}$), and occasionally we will use P as $P = \underline{P} \cup \overline{P}$, where the meaning of P will be clear from the context. For local orbitals a charge distribution $\omega_{pq} = \phi_p \phi_q$ is negligible if the MOs ϕ_p and ϕ_q are well separated in space. Therefore, only when ω_{ia} and ω_{jb} are both non–negligible the integral g_{iajb} is non–negligible. Thus, if ϕ_i and ϕ_j are both assigned to site P ($i, j \in \underline{P}$) then g_{iajb} is non–negligible and contributes to the energy only if $a, b \in [\overline{P}]$ where the bracket denotes the set of orbitals assigned to sites spatially close to site P (including P itself). For compactness we will collectively refer to the occupied $[\underline{P}]$ and virtual $[\overline{P}]$ orbital spaces as $[P] = [\underline{P}] \cup [\overline{P}]$.

When using the locality of a MO basis, it is important that the MOs are represented by a subset of the underlying AOs in order to obtain a method that is local in all respects. How the truncation of the AO basis may be obtained when a set of local MOs is known is not the focus of this analysis. For completeness, however, the practical procedure for the truncation of the AO basis for a given fragment is summarized in Appendix B.1.

In this section and in Section 6.2 we present some illustrative numerical calculations, which refer to the simple test system in Figure 6.1 where the HF orbitals have been localized using the second power of the second moment localization function [100] and all CC equations have been converged to a residual norm of 10^{-9} a.u. To illustrate the locality arguments in the current and following discussions, we have plotted the decays of the absolute overlap (integrated over all space), $v_{pq} = \int |\phi_p||\phi_q| dV$ (for $p \in P$ and $q \in [P]$), as v_{ia}, v_{ab} and v_{ij}, in Figure 6.2a. The absolute overlap integrals decay rapidly with radius R_P as a manifestation of the locality of the charge distributions ω_{pq}. The Fock matrix elements F_{pq} (for $p \in P$ and $q \in [P]$) and the two-electron integrals, g_{iajb} (for $i, j, b \in P$ and $a \in [P]$) are governed by charge distributions

Figure 6.2: Distance decays for absolute overlap $v_{pq} = \int |\phi_p||\phi_q| dV$, two-electron integral g_{iajb}, and Fock matrix elements for the test system in Figure 6.1. The points shown are the maximum values of consecutive 1.6 Å intervals.



(a) Absolute overlap integrals v_{ia}, v_{ab} and v_{ij} as a measure for the locality.

(b) The Fock matrix elements F_{pq} and two–electron integrals g_{iajb} follow the decay behavior of v_{pq}.

and therefore follow the same decay behavior as illustrated in Figure 6.2b.

6.1.2 Local formulation of the correlation energy expression

We now discuss how the locality of the orbitals may lead to a local description of the molecular system. Replacing the summations in Eq. (1.33) by summations over sites and pairs of sites we may write the correlation energy without introducing any approximations as

$$E^x_{corr} = \sum_P [E^x_P + \sum_{Q<P} E^x_{PQ}]. \tag{6.1}$$

where x may, in a DEC context, either refer to the occupied ($x = o$) or the virtual ($x = v$) partitioning scheme, which lead to alternative local descriptions of the correlation energy. In the occupied partitioning scheme, the occupied single fragment energy E^o_P for site P may be written as

$$E^{o,f}_P = \sum_{i,j \in \underline{P}} \sum_{a,b} \tau^{ab}_{ij} L_{iajb}, \tag{6.2}$$

where τ^{ab}_{ij} is defined in Eq. (1.34). Furthermore, using the short hand notation $(\sum_a + \sum_b) \sum_c = \sum_a \sum_c + \sum_b \sum_c$, we may write the occupied pair interaction energy E^o_{PQ} between sites P and Q as

$$E^{o,f}_{PQ} = \left(\sum_{\substack{i \in \underline{P} \\ j \in \underline{Q}}} + \sum_{\substack{j \in \underline{P} \\ i \in \underline{Q}}} \right) \sum_{a,b} \tau^{ab}_{ij} L_{iajb}. \tag{6.3}$$

Similarly, the virtual single fragment energy E^v_P and pair interaction energy E^v_{PQ} may be calculated according to

$$E^{v,f}_P = \sum_{a,b \in \overline{P}} \sum_{i,j} \tau^{ab}_{ij} L_{iajb}, \tag{6.4}$$

and

$$E_{PQ}^{v,f} = \left(\sum_{\substack{a\in\overline{P}\\ b\in\overline{Q}}} + \sum_{\substack{b\in\overline{P}\\ a\in\overline{Q}}} \right) \sum_{i,j} \tau_{ij}^{ab} L_{iajb}, \tag{6.5}$$

where, for both the occupied and virtual partitioning schemes, amplitudes for the full molecular system are required to evaluate the expressions of Eqs. (6.2)-(6.5). Therefore, also the amplitude equations have to be solved for the full molecular system and no cost–reduction has been obtained yet. Also, *no approximations* have been introduced in the correlation energy expression in Eq. (6.1) if Eqs. (6.2)-(6.5) are used. For the sake of completeness we note that for MP2 calculations there exists a third partitioning scheme, called Lagrangian partitioning $\mathcal{L}_{corr}^{MP2}$ [102]. It is practically a mixture of the occupied and virtual partitioning schemes and will therefore not add a new perspective to this discussion, but we will use it in Chapter 7, where correlation density matrices are calculated.

Following the locality considerations from Section 6.1.1, the integrals g_{iajb} may be neglected (to the given precision) when the indices i and a (j and b) refer to spatially well separated orbitals, and thus a truncation in the summations of Eqs. (6.2)-(6.5) may be introduced. Using this restriction in the summations of the local occupied single fragment energy E_P^o and local pair interaction energy E_{PQ}^o, we obtain

$$E_P^o = \sum_{i,j\in\underline{P}} \sum_{a,b\in[\overline{P}]} \tau_{ij}^{ab} L_{iajb}, \tag{6.6}$$

and

$$E_{PQ}^o = \left(\sum_{\substack{i\in\underline{P}\\ j\in\underline{Q}}} + \sum_{\substack{j\in\underline{P}\\ i\in\underline{Q}}} \right) \sum_{a,b\in[\overline{P}]\cup[\overline{Q}]} \tau_{ij}^{ab} L_{iajb}. \tag{6.7}$$

For the virtual single fragment energy E_P^v and pair interaction energy E_{PQ}^v we obtain

$$E_P^v = \sum_{a,b\in\overline{P}} \sum_{i,j\in[\underline{P}]} \tau_{ij}^{ab} L_{iajb}, \tag{6.8}$$

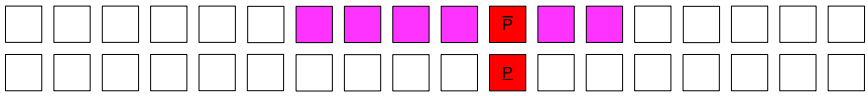
and

$$E_{PQ}^v = \left(\sum_{\substack{a\in\overline{P}\\ b\in\overline{Q}}} + \sum_{\substack{b\in\overline{P}\\ a\in\overline{Q}}} \right) \sum_{i,j\in[\underline{P}]\cup[\underline{Q}]} \tau_{ij}^{ab} L_{iajb}, \tag{6.9}$$

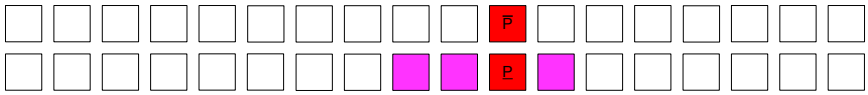
where all quantities entering the expressions of Eqs. (6.6)-(6.9) may be evaluated in a local region around the involved sites P and Q and the whole algorithm asymptotically scales as $\mathcal{O}(N^2)$. If all orbitals are included in $[\underline{P}]$ and $[\overline{P}]$ for all sites P then Eqs. (6.6)-(6.9) and Eqs. (6.2)-(6.5) are identical.

In order to arrive at a truly linear scaling algorithm, the second summation in Eq. (6.1) has to be truncated. Since the pair fragment contributions E_{PQ}^x decay rapidly with the distance between the sites P and Q as R_{PQ}^{-6} (dispersion effects), a screening technique for discarding pair fragment contributions (to the given precision) can be used, as will be detailed in Section 6.3.2.

Figure 6.3: Schematic illustration where the upper and lower rows of squares symbolize the sets of virtual and occupied orbitals assigned to a center, respectively. The orbitals assigned to site P are shown as red squares. The space outside P where the integrals L_{iajb} are non-negligible is shown in magenta.



(a) Schematic drawing of $S^{o,P}_{\mathrm{EOS}}$



(b) Schematic drawing of $S^{v,P}_{\mathrm{EOS}}$

The evaluation of the occupied single fragment energy E^o_P in Eq. (6.6) requires the amplitudes of the occupied energy orbital space (EOS). To indicate that a four-index quantity, X^{ab}_{ij}, is restricted to the occupied EOS, we write,

$$X^{ab}_{ij} \in S^{o,P}_{\mathrm{EOS}}, \tag{6.10}$$

where $S^{o,P}_{\mathrm{EOS}}$ refers to a collection of four orbital indices with $i,j \in \underline{P}$ and $a,b \in [\overline{P}]$. We will use the following compact notation for $S^{o,P}_{\mathrm{EOS}}$,

$$S^{o,P}_{\mathrm{EOS}} \coloneqq \underline{P} \times \underline{P} \times [\overline{P}] \times [\overline{P}] = \underline{P}^2 \times [\overline{P}]^2. \tag{6.11}$$

Similarly, E^v_P requires the amplitudes of the virtual EOS, which can be written as

$$S^{v,P}_{\mathrm{EOS}} \coloneqq [\underline{P}]^2 \times \overline{P}^2. \tag{6.12}$$

Both spaces $S^{o,P}_{\mathrm{EOS}}$ and $S^{v,P}_{\mathrm{EOS}}$ are depicted schematically in Figure 6.3. We henceforth use the generic $S^{x,P}_{\mathrm{EOS}}$, E^x_P, and E^x_{PQ} to refer to one of the partitioning schemes. In order to obtain the single fragment and pair interaction energies to the predefined precision, the amplitudes of $S^{x,P}_{\mathrm{EOS}}$ have to be known. When the amplitude equations are solved, the EOS amplitudes couple to amplitudes outside of $S^{x,P}_{\mathrm{EOS}}$, and it is therefore necessary to take this coupling into account and identifying the orbital space that is required to calculate E^x_P to a predefined precision. The determination of EOS amplitudes to the requested precision is the subject of Section 6.2.

6.2 Orbital spaces in DEC–CC fragment energy calculations

In this section we will start out by summarizing how the single fragment MP2 energy is affected when the orbital spaces are increased. We will then classify the additional mechanisms, when stepwise moving from the initial MP2 model to CCD, CCSD and finally to CCSD(T). We will furthermore identify the necessary spaces in which the respective (MP2, CCSD, CCSD(T)) equations have to be solved in order to arrive at a final correlation energy that underlies rigorous error control. We will furthermore develop a screening technique, which allow us to reduce the number of pair calculations in a DEC calculation, maintaining error control and finally we will describe a practical black box method for identifying the necessary orbital spaces.

6.2.1 Space extensions in MP2 fragment calculations

The MP2 amplitude equations constitute a set of linear equations with a positive definite coefficient matrix

$$\sum_c t_{ij}^{cb} F_{ac} + \sum_c t_{ij}^{ac} F_{cb} - \sum_k t_{kj}^{ab} F_{ik} - \sum_k t_{ik}^{ab} F_{kj} = -g_{aibj}, \tag{6.13}$$

If a canonical HF basis is used the Fock matrix is diagonal and Eq. (6.13) is solved in one iteration. If a basis of local orbitals is used the Fock matrix is diagonally dominant and Eqs. (6.13) may be solved in a few iterations using standard iterative algorithms, such as the conjugate gradient or the conjugate residual methods [26]. The diagonal dominance of the Fock matrix is important for the identification of the local orbital spaces that are necessary for obtaining the amplitudes required to evaluate the single fragment and pair interaction energies. We will now examine how local orbital spaces propagate when single fragment energies are determined during an iterative procedure where the MP2 amplitude equations are solved; in particular, we will consider the propagation when the conjugate residual algorithm is used.

Iterative solution of the MP2 amplitude equations

When solving Eqs. (6.13) using the conjugate residual algorithm, the amplitudes of the $(n+1)$'th iteration are determined from the amplitudes of the n'th iteration

$$t_{ij,n+1}^{ab} = t_{ij,n}^{ab} + \alpha R_{ij,n}^{ab}, \tag{6.14}$$

where the residual $R_{ij,n}^{ab}$ of the n'th iteration

$$R_{ij,n}^{ab} = -g_{aibj} - \sum_c t_{ij,n}^{cb} F_{ac} - \sum_c t_{ij,n}^{ac} F_{cb} + \sum_k t_{kj,n}^{ab} F_{ik} + \sum_k t_{ik,n}^{ab} F_{jk}, \tag{6.15}$$

is used as a search direction. In the conjugate residual method, a line search is performed along the residual direction by optimizing the α parameter (see Chapter 2). The line search affects the convergence rate of the algorithm, but it does not influence how new orbital spaces are introduced in the iterative algorithm. For simplicity, we therefore set $\alpha = 1$ in the rest of this analysis.

In the n'th iteration the energy for single fragment P is

$$E_{P,n}^x = \sum_{S_{\mathrm{EOS}}^{x,P}} t_{ij,n}^{ab} L_{iajb}, \tag{6.16}$$

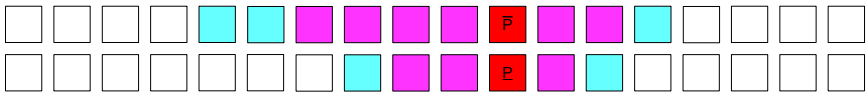
where the set of amplitudes entering this expression $t \in S_{\mathrm{EOS}}^{x,P}$ will henceforth be referred to as EOS amplitudes. The energy change in iteration n, $\Delta E_{P,n}^x$, is given by,

$$\begin{aligned} \Delta E_{P,n}^x &= E_{P,n}^x - E_{P,n-1}^x, \\ &= \sum_{S_{\mathrm{EOS}}^{x,P}} R_{ij,n-1}^{ab} L_{iajb}. \end{aligned} \tag{6.17}$$

where $E_{P,0}^x \coloneqq 0$. Using Eqs. (6.14) and (6.16) the converged single fragment energy may be written as

$$E_P^x = \sum_{n=1}^{n_{it}} \Delta E_{P,n}^x, \tag{6.18}$$

Figure 6.4: Iteration 2: The S_1^P space is symbolized by the squares in red and magenta. The residual is generated in the S_2^P space (all colored boxes) where the cyan squares symbolize the space extension $S_2^P \setminus S_1^P$ in iteration 2. This space interacts directly with the orbitals of the EOS.



where n_{it} is the total number of iterations in the iterative scheme and $E_P^x = E_{P,n_{it}}^x$. The residual $R_n \in S_{\mathrm{EOS}}^{x,P}$ approaches zero with increasing n and $\Delta E_{P,n}^x$ therefore becomes negligible. We now apply the iterative procedure described in Eq. (6.15) and analyze the propagation of the orbital space throughout the iterative procedure to identify the target orbital space required for determining E_P^x to the predefined precision.

Propagation of orbital spaces during the iterative procedure

The EOS amplitudes are defined in $S_{\mathrm{EOS}}^{o,P}$ and $S_{\mathrm{EOS}}^{v,P}$ spaces for the occupied and virtual partitioning schemes, respectively, see Figure 6.3. However, it is convenient to extend these spaces slightly to put the analysis for the occupied and virtual partitioning schemes on an equal footing. We therefore introduce the target space, S_1^P,

$$S_1^P = [\underline{P}]^2 \times [\overline{P}]^2, \qquad (S_{\mathrm{EOS}}^{o,P} \cup S_{\mathrm{EOS}}^{v,P}) \subset S_1^P. \tag{6.19}$$

To simplify the analysis below we collectively consider the EOS amplitudes for the occupied and virtual partitioning schemes to be defined in the same target space S_1^P.

In the first iteration of the iterative procedure described in Eq. (6.15) all amplitudes are set to zero,

$$t_{ij,1}^{ab} = 0, \tag{6.20}$$

and therefore $E_{P,1}^x = \Delta E_{P,1}^x = 0$. The single fragment energy is defined by the EOS amplitudes, and we therefore restrict the residual of the first iteration to the target space S_1^P. Using Eq. (6.15) we thus obtain,

$$R_{ij,1}^{ab} = -g_{aibj} \in S_1^P. \tag{6.21}$$

Using Eqs. (6.14) and (6.20), the amplitudes of the second iteration become

$$t_{ij,2}^{ab} = R_{ij,1}^{ab} \in S_1^P. \tag{6.22}$$

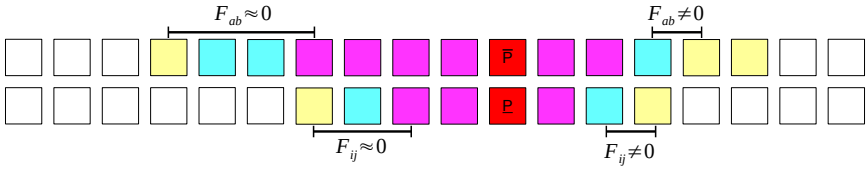
The energy difference between the first and second iterations is thus,

$$\Delta E_{P,2}^x = \sum_{S_{\mathrm{EOS}}^{x,P}} R_{ij,1}^{ab} L_{iajb}, \tag{6.23}$$

where we have used Eq. (6.17). The EOS amplitudes in Eq. (6.22) relax through the iterations and we now analyze the coupling mechanism leading to this relaxation.

Figure 6.2 shows that the distance decay of the two-electron integrals and the Fock matrix elements are very similar. For simplicity and without loss of generality we therefore use the same extents $[\underline{P}]$ and $[\overline{P}]$ to describe non-negligible two-electron integrals and Fock matrix elements in the following analysis. A Fock matrix element F_{ij} (F_{ab}) where $i \in [\underline{P}]$ ($a \in [\overline{P}]$) will thus be non-negligible only if $j \in [\underline{P}]_2$ ($b \in [\overline{P}]_2$), where $[\underline{P}]_2$ ($[\overline{P}]_2$) is a spatial extension of $[\underline{P}]$ ($[\overline{P}]$).

Figure 6.5: Iteration 3: The residual is generated in the S_3^P space (all colored boxes) where the orbitals of sites with light yellow squares ($S_3^P \setminus S_2^P$) symbolize the orbitals which interact directly with the orbitals of the $S_2^P \setminus S_1^P$ space (cyan) through non-negligible Fock matrix elements, but not directly with the orbitals of the S_1^P space (red and magenta).



In the second iteration the amplitudes in S_1^P are coupled directly to sites in the proximity of $[P]$ through summations with non-negligible Fock matrix elements in the residual of Eq. (6.15). For example, the following terms of Eq. (6.15) for the second iteration illustrate space extensions of the occupied and virtual spaces, respectively,

$$\sum_k t_{ik,2}^{ab} F_{jk} \qquad t_{ik,2}^{ab} \in S_1^P; \quad j \in [\underline{P}]_2, \tag{6.24a}$$

$$\sum_c t_{ij,2}^{cb} F_{ca} \qquad t_{ij,2}^{cb} \in S_1^P; \quad a \in [\overline{P}]_2. \tag{6.24b}$$

The residual of the second iteration is thus given by,

$$R_{ij,2}^{ab} = -g_{aibj} - \sum_c t_{ij,2}^{cb} F_{ca} - \sum_c t_{ij,2}^{ac} F_{cb} + \sum_k t_{kj,2}^{ab} F_{ik} + \sum_k t_{ik,2}^{ab} F_{jk}; \quad R_2 \in S_2^P, t_2 \in S_1^P. \tag{6.25}$$

The space extension mechanism caused by non-negligible Fock matrix elements exemplified in Eqs. (6.24) and (6.25) will henceforth be referred to as mechanism 1.

We now generalize the notation above and introduce the space S_n^P,

$$S_n^P = [\underline{P}]_n^2 \times [\overline{P}]_n^2, \tag{6.26}$$

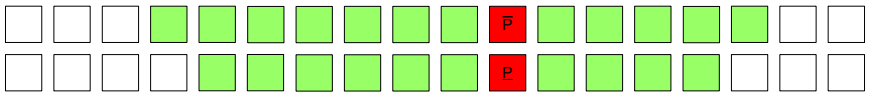
where $[\underline{P}]_n$ ($[\overline{P}]_n$) is the set of occupied (virtual) orbitals which have non-negligible Fock matrix elements with at least one of the orbitals in $[\underline{P}]_{n-1}$ ($[\overline{P}]_{n-1}$), and where $[\underline{P}]_1 := [\underline{P}]$ ($[\overline{P}]_1 := [\overline{P}]_1$).

The amplitudes of the third iteration $t_{ij,3}^{ab}$ will be generated in S_2^P according to Eq. (6.14) giving the residual $R_{ij,3}^{ab} \in S_3^P$, such that more, yet untouched components, of the environment of P enters the residual through the coupling mechanism described in Eq. (6.24). The new space components will not be coupled to S_1^P directly as the Fock matrix elements between these spaces are negligible, see Figure 6.5. The amplitudes in the $S_3^P \setminus S_2^P$ space thus interact directly with the amplitudes of the $S_2^P \setminus S_1^P$ space which in turn couples with the amplitudes in the S_1^P space. Hence, the relaxation effects of the $S_3^P \setminus S_2^P$ space on the EOS amplitudes will be small and indirect, and these effects will therefore be referred to as secondary coupling effects.

For the fourth and following iterations these mechanisms are preserved, i.e., additional couplings are introduced in the already considered space and new spaces are introduced which indirectly couple to the $S_{\text{EOS}}^{x,P}$ space through other spaces. The effects on the EOS amplitudes from the newly introduced spaces thus become more indirect and smaller in each iteration.

The above developments may be summarized as follows. To evaluate the single fragment energies we need the amplitudes of the $S_{\text{EOS}}^{x,P}$ space that was introduced in iteration 1. The EOS amplitudes are a subset of the amplitudes of S_1^P, which has the same extent as $S_{\text{EOS}}^{x,P}$. In

Figure 6.6: All the orbitals that were found to be important for the determination of the EOS amplitudes to a certain precision are collected in an effective coupling space S_{EFF} shown as green squares. The effective coupling environment of P is denoted $[P]_{\mathrm{EFF}}$.



the second iteration direct coupling to the outside space $S_2^P \setminus S_1^P$ is introduced which affects the EOS amplitudes significantly. The $S_2^P \setminus S_1^P$ space is therefore important for the evaluation of the single fragment energy. The effects of the additional spaces introduced in the following iterations affect the EOS amplitudes only indirectly and are referred to as secondary, tertiary, etc. effects. The requested precision will consequently define how much of the environment of P has to be considered. When evaluating the single fragment energies, we may introduce an effective coupling space S_{EFF}^P where the MP2 equations are solved to give the EOS amplitudes to the requested precision (the FOT), see Figure 6.6. Since S_{EFF}^P is the space where the amplitude equations have to be solved, it will also be referred to as the amplitude orbital space (AOS). When calculating the single fragment energy, a determination of the EOS is impractical, and we thus simply redefine $[P] \coloneqq [P]_{\mathrm{EFF}}$ (compare Figs. 6.5 and 6.6), i.e., the $[\underline{P}]$ and $[\overline{P}]$ spaces in Eqs. (6.6) and (6.8) are effectively replaced by $[\underline{P}]_{\mathrm{EFF}}$ and $[\overline{P}]_{\mathrm{EFF}}$.

6.2.2 Space extension in CCSD fragment calculations

The CCSD amplitude equations may be viewed as a nonlinear extension of the MP2 amplitude equations where the additional terms are of higher order in a Møller-Plesset perturbation analysis of the equations. Thus, the energetically largest contributions leading to a space extension have been considered in Section 6.2.1. The focus of this section is to analyze the space extensions that may be introduced in the residual of Eq. (6.14) by the additional smaller terms of the CCSD amplitude equations. In order to simplify the notation in the following section, the CCSD doubles residual in the T_1 transformed formulation from Eq. (1.54) may be rewritten as

$$ {}^{\mathrm{CCSD}}R^D = \Omega^{A2} + \Omega^{B2} + \Omega^{C2} + \Omega^{D2} + \Omega^{E2}, \tag{6.27}$$

and the CCSD singles residual from Eq. (1.53) then reads

$$ {}^{\mathrm{CCSD}}R^S = \Omega^{A1} + \Omega^{B1} + \Omega^{C1} + \Omega^{D1}, \tag{6.28}$$

where the individual terms are given in Table 1.2 and we recall that the T_1 transformation, denoted by the tilde in Table 1.2, is introduced by using the transformation matrices $\mathbf{\Lambda}^p$ and $\mathbf{\Lambda}^h$ from Eqs. (1.48) and (1.49), instead of the MO transformation matrices $\mathbf{C}$.

When the singles amplitudes are neglected, ${}^{\mathrm{CCSD}}R^D$ becomes the nonlinear CCD residual ${}^{\mathrm{CCD}}R$ (removing the tildes in Table 1.2). The starting point for analyzing orbital space extensions for CCSD fragment energy calculations is an analysis for the CCD fragment energy in Section 6.2.2. In Section 6.2.2 we describe the progression of orbital spaces in CCSD when the effects of singles amplitudes are considered.

Space extension mechanisms in the CCD model

The space extension due to the MP2 part of the residual ${}^{\mathrm{CCD}}R$ was considered in Section 6.2.1 and we will now focus on how the terms of the CCD equations beyond MP2 lead to space extensions in the iterative process. We may use the framework established for MP2 in Section 6.2.1

with the only modification that, when solving the nonlinear CCD equations using the conjugate residual method [26], the residual $R^{ab}_{ij,n}$ defined in Eq. (6.15) has to be replaced by the residual ${}^{\mathrm{CCD}}R^{ab}_{ij,n}$.

The MP2 amplitude equations in Eq. (6.15) correspond to g_{aibj} in R^{A2} and the linear Fock matrix terms of R^{E2} in the CCD amplitude equations. The CCD equations thus contain contain additional terms where two–electron integrals are contracted linearly or quadratically with cluster amplitudes. The quadratic terms do not contribute to the space extension of the residual since all the indices entering these contributions are fixed by the amplitudes of the previous iteration. For example, in the second part of the Ω^{B2}_n term, $\sum_{kl}\sum_{cd} t^{ab}_{kl,n} t^{cd}_{ij,n} g_{kcld}$, the integral indices are fixed to the space of the cluster amplitudes of iteration n and therefore no space extension is introduced in the residual.

For the MP2 analysis in Section 6.2.1 we encountered the space extension mechanism due to non-vanishing Fock matrix elements. We now consider three additional space extension mechanisms for CCD and CCSD, which may be summarized as follows:

1. A propagation due to non–vanishing Fock matrix elements
2. A propagation due to non–vanishing charge distributions in two-electron integrals
3. A propagation due to long-range interactions between charge distributions
4. A propagation due to the T_1 transformed Fock matrix.

Mechanisms 1, 2, and 3 are present for CCD, while mechanism 4 is present only for CCSD and will be discussed in Section 6.2.2. An example of mechanism 2 is the second term in $\Omega^{A2}_{aibj,n}$, i.e.,

$$\Omega^{A2.2}_{aibj,n} = -\sum_{cd} t^{cd}_{ij,n} g_{acbd}. \tag{6.29}$$

At iteration n the amplitudes $t_n \in S^P_{n-1}$, and thus the free indices a, b of the integrals g_{acbd} are fixed by the charge distributions ω_{ac} and ω_{bd} to the dummy indices c, d of the summation. Thus, assuming that the non–vanishing Fock matrix elements and the non–vanishing charge distribution of the integrals have the same extent (see Figure 6.2b), $\Omega^{A2.2}_n \in S^P_n$ is defined in the same space as the MP2 residual of the same iteration. In short, mechanisms 1 and 2 leads to the same space extensions.

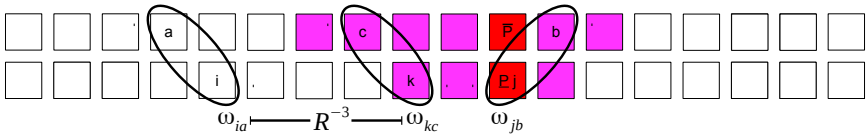
All additional terms of the CCD doubles residual follow mechanism 2 except for one of the terms in Ω^{D2}_n which is denoted as $\Omega^{D2.1}_n$ and given by,

$$\Omega^{D2.1}_{aibj,n} = -\sum_{ck} t^{bc}_{jk,n} g_{aikc}. \tag{6.30}$$

Again, the amplitudes of iteration n are defined in $t_n \in S^P_{n-1}$ and the dummy indices of the summation k, c are therefore restricted to that space. The charge distribution ω_{ai} is thus only bound to the center P by its inherent third power decay with the distance between the charge distributions ω_{ai} and ω_{kc} in the g_{aikc} integral (see Figure 6.7 and Ref. 38). Mechanism 3 may thus lead to a further extent than mechanisms 1 and 2.

In Figure 6.8a we compare the decays of Fock matrices and two-electron integrals which represent the different decay mechanisms as indicated in the figure. The decays are initially similar but at larger distances mechanism 3 decays slower than mechanism 1 and 2. However,

Figure 6.7: Long-range interactions in the $\Omega^{D2.1}_{aibj,n}$ term between the amplitudes $t^{bc}_{jk,n} \in S^P_n$ and the charge distribution outside S^P_n, i.e., $\omega_{ai} \notin [\underline{P}]_n \times [\overline{P}]_n$ (shown here for $n = 1$). The charge distribution ω_{ai} interacts with the charge distribution ω_{kc} via an R^{-3} distance decay in the g_{aikc} integrals [38].



we recall that all space extension mechanisms only lead to a relaxation of the EOS amplitudes which has an indirect effect on the single fragment energy and also that mechanism 3 influences only one of the CCD residual terms. We therefore expect that mechanism 3 has a small effect on the single fragment energy which becomes important only for very high precision.

This analysis is supported by the numerical illustrations in Figure 6.8b where the error decay of the single fragment energy in Figure 6.1 is given for the MP2 model (mechanism 1), a modified CCD model where the term decaying according to mechanism 3 has been removed, and the regular CCD model where mechanisms 1, 2, and 3 are all present. The mechanisms present in the model is given in parenthesis after the model name. For small expansion radii R_P the three models decay similarly although the CCD(1,2) and CCD(1,2,3) errors are smaller than the MP2(1) error in the region from 4-8Å due to a fortuitous error cancellation. However, if a very high precision is requested (below a FOT of 10^{-7} for the present illustration) mechanism 3 dominates and needs to be taken into account. Thus, unless extremely high precision is requested, the orbital spaces determined from a CCD analysis are very similar to those obtained from an MP2 analysis. In general we may redefine the square bracket notation $[P]$ as the orbital space that interacts with P through any of the mechanisms 1, 2, or 3 and find that it is possible to solve the coupled cluster equations to a predefined precision within a restricted effective coupling space S_{EFF}.

Space extension mechanisms in the CCSD model with focus on the effect of singles

In Section 6.2.2 we described the progression of the space extension for the cluster amplitudes in a CCD calculation. In this section we examine the space progression when singles amplitudes are considered and the residual equation is the CCSD residual.

As for the doubles amplitudes, the singles EOS amplitudes are defined by Eqs. (6.6) and (6.8). We find that the space of the singles residual of iteration n has the same extent $[P]_n$ as the doubles residual S^P_n (disregarding mechanism 3). This may also be seen indirectly from the fact that the singles and doubles amplitudes t^a_i and t^{ab}_{ij} are comparable in size and display a similar decay behavior as shown in Figure 6.9. Then, the amplitudes of iteration n have the extent $[P]_{n-1}$. The space extension in the singles residual occurs only through mechanism 2 and is thus accounted for in Section 6.2.2. In order to facilitate the analysis for the doubles residual, we turn around the point of view and identify the coupling contributions from external spaces to the doubles residual ${}^{\mathrm{CCSD}}R^D_n$ within the EOS S^o_{EOS}, i.e., we consider the EOS indices fixed and the summation indices are restricted to S^P_{n-1} by the residual. In the integral contribution $\tilde{g}_{aibj}$ to the doubles residual all indices are T_1 transformed and in iteration n it may be written as

$$\tilde{g}_{aibj} = g_{aibj} + \sum_{k} t^a_{k,n} g_{kibj} + \cdots + \sum_{kc} t^a_{k,n} t^c_{i,n} g_{kcbj} + \cdots + \sum_{kcl} t^a_{k,n} t^c_{i,n} t^b_{l,n} g_{kclj} + \cdots + \sum_{kcld} t^a_{k,n} t^c_{i,n} t^b_{l,n} t^d_{j,n} g_{kcld}, \tag{6.31}$$

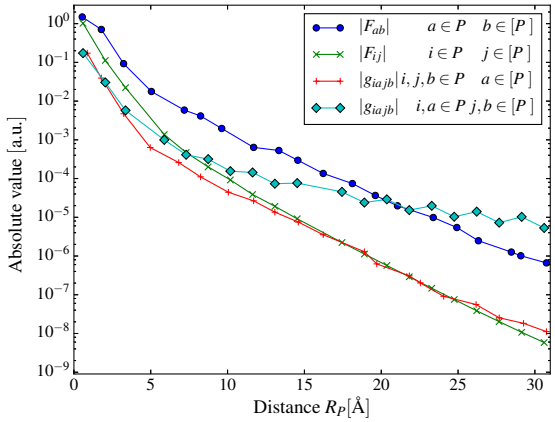
where one example for each linear, quadratic, cubic, and quartic terms in the singles amplitudes is explicitly given. Since we have chosen to look at $\tilde{g}_{aibj} \in S^o_{\mathrm{EOS}}$, we find that terms that are linear in the singles amplitudes will only couple directly to $[\underline{P}]$ through the non-vanishing charge distribution ω_{ki} (or to $[\overline{P}]_2$ through the respective virtual T_1 transformation) and thus expands as mechanism 2. The quadratic coupling terms extending furthest in space couple directly to an environment where a charge distribution (ω_{kc}) is bound to a charge distribution of the EOS (ω_{bj}) by mechanism 3. Cubic and quartic terms in the singles amplitudes may couple directly to $[\overline{P}]_n$ and are bound only loosely to the EOS. From Figure 6.9 we see that terms which are quadratic in the singles amplitudes will introduce a minor modification of the residual in the EOS compared to the terms linear in the cluster amplitudes, and therefore the effect on the energy of these terms will be small (contributions from cubic and quartic terms of Eq. (6.31) will be even smaller). That the singles effects are small in general, is shown in Figure 6.10. The space extension contribution from the singles entering through T_1 transformed integrals will thus be dominated by a charge distribution decay mechanism (mechanism 2).

The Ω_n^{E2}, Ω_n^{C1}, and Ω_n^{D1} terms have Fock matrix contributions constructed from a T_1-transformed density matrix which can be written as [9],

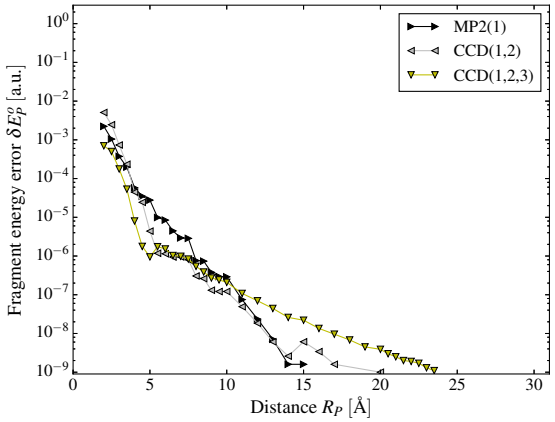
$$\tilde{D}_{\mu\nu} = D_{\mu\nu} + \sum_{ia} C_{\mu i} t_i^a C_{\nu a}, \tag{6.32}$$

where $D_{\mu\nu}$ is the HF density matrix. In a practical DEC calculation for single fragment P the i and a summations in Eq. (6.32) are restricted to the $[P]_{\mathrm{EFF}}$ space, leading to errors in $\tilde{D}_{\mu\nu}$. This error type was denoted mechanism 4 in Section 6.2.2.

Figure 6.8: Decays of Fock-matrix elements, two-electron integrals, and single fragment energies.



(a) Decay of Fock matrix elements (mechanism 1: circles and crosses), charge distribution decay of two-electron integrals (mechanism 2: pluses), and long-range R^{-3} decay of two-electron integrals (mechanism 3: diamonds).



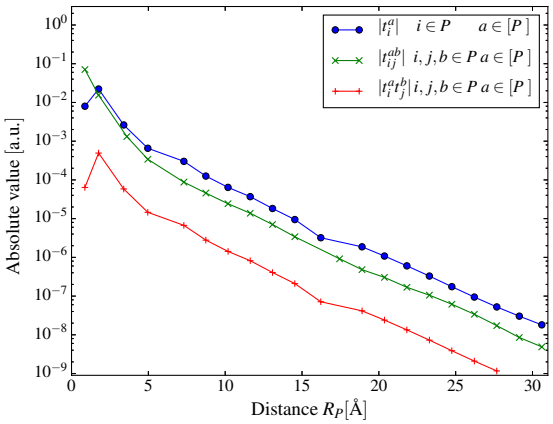
(b) Decay behavior of the (occupied) single fragment energy error $\delta E_P^o = E_P^{o,f} - E_P^o$, and E_P^o is obtained from Eq. (6.6). The error decay is shown for MP2 (mechanism 1), a modified CCD model where long-range terms have been removed (mechanisms 1 and 2), and a regular CCD calculation (mechanisms 1, 2, and 3).

To isolate the errors associated with the T_1-transformed two-electrons integrals from the ones associated with the T_1-transformed density matrix we have defined a modified CCSD(1,2,3) model where the exact T_1-transformed density matrix is used, i.e., all singles amplitudes are included in Eq. (6.32) such that errors of type 4 do not occur. This model is to be compared with the actual DEC–CCSD model where the a and i summations in Eq. (6.32) are restricted to the $[P]_{\mathrm{EFF}}$ space, i.e., errors of type 1, 2, 3, and 4 all occur and this model is therefore denoted CCSD(1,2,3,4).

In Fig. 6.10 we compare the single fragment energy errors for the occupied partitioning scheme for various models. It is seen that the errors of the CCD(1,2,3) and CCSD(1,2,3) models are very similar, indicating that the use of T_1-transformed two-electron integrals in CCSD does not lead to additional space extensions as compared to CCD(1,2,3) in accordance with the analysis above. On the other hand the decay of CCSD(1,2,3,4) model is very slow for high precision (errors below $\approx 10^{-7}$ a.u.), indicating that the truncating errors in the a and i summations in Eq. (6.32) become dominating.

Figure 6.10 thus indicates that mechanism 4 is not important for fragment errors of interest in practical applications (e.g., a FOT of 10^{-4} or 10^{-5} a.u.). However, if a high-precision $\tilde{\mathbf{D}}$ matrix is requested, one may start by carrying out all single fragment calculations and collect the singles amplitudes (t_i^a for $i \in \underline{P}$ and $a \in [\overline{P}]_{\mathrm{EFF}}$ for all single fragments P) in a full molecular singles amplitudes matrix, t_1^{full}. $\tilde{D}_{\mu\nu}$ may then be calculated using t_1^{full} and this improved density matrix is then used in a second round of single (and pair) fragment calculations.

Figure 6.9: Decay of the singles and doubles amplitudes. In order to estimate their respective energy contributions, the singles amplitudes have also been squared according to the correlation energy expression, see Eq. (1.33).



Summary for the space extension in a fragment CCSD calculation

In this section we have in detail analyzed how orbital spaces extend when CCSD amplitude equations are solved to obtain the amplitudes that are required to evaluate to fragment energy within the requested precision. The conclusion is that the orbital spaces used for evaluating fragment energies used for MP2 may only be used for fragment energies at the CCSD for practical purposes. In Figure 6.11, we have given the MP2 and CCSD single fragment errors for

Figure 6.10: Illustrating the effect of mechanisms 1–4 on the single fragment energy error δE^o_P, where for comparison the errors of CCD(1,2,3) (blue) and CCSD without coupling errors (cyan) are displayed. When eliminating mechanism 4 from CCSD(1,2,3,4) (green), the decays of CCSD(1,2,3) (red) and CCD(1,2,3) are almost superimposed. The artificially good CCSD fragment energies result from sign changes in the errors.

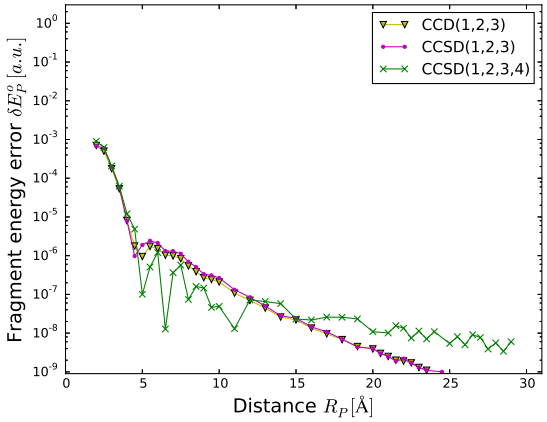
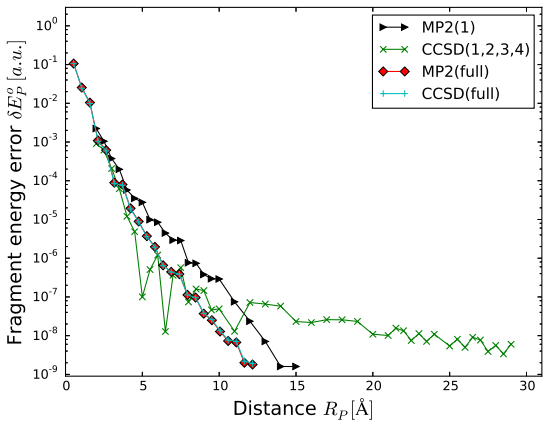


Figure 6.11: Error in the single fragment energy for occupied partitioning scheme as function of expansion radius for MP2 and CCSD. Blue (red): MP2 (CCSD) error when amplitude equations have been solved in the $[P]_{\mathrm{EFF}}$ space (see Figure 6.6) and the virtual summation in Eq. (6.6) is restricted to $[\overline{P}]_{\mathrm{EFF}}$ (practical DEC calculation). Green (cyan): MP2 (CCSD) error when the amplitude equations have been solved in the full orbital space, while the virtual summation in Eq. (6.6) is still restricted to $[\overline{P}]_{\mathrm{EFF}}$ (artificial DEC calculation which removes coupling errors).



the occupied partitioning scheme in Eq. (6.6) as a function of expansion radius in Figure 6.1. We compare a practical DEC calculation where the amplitude equations are solved within the $[P]_{\mathrm{EFF}}$ space (see Figure 6.6) to an artificial DEC calculation where the coupling errors have been removed (the amplitude equations have been solved in the complete orbital space).

Clearly, when the amplitude equations have been solved in the complete orbital space, the MP2 (green) and CCSD (cyan) errors are virtually identical, demonstrating that the errors are determined by the decay of the two-electron integrals in Eq. (6.6). On the other hand, solving the amplitude equations in the restricted $[P]_{\mathrm{EFF}}$ space in general increases the MP2 (blue) and CCSD (red) errors slightly (compared to using the amplitudes converged in the full space) for single fragment errors up to about 10^{-6} a.u. where the long range effects in the doubles residual start to dominate. From about 10^{-7} a.u. precision, special attention has to be addressed to the way singles are treated, e.g. by using the improvement to the T_1 transformed density discussed in Section 6.2.2.

In Figure 6.12, we have given the decays of the single fragment energy errors for a selected fragment of decanoic acid, using the cc-pVDZ basis for the occupied and virtual partitioning schemes. For both partitioning schemes, there is a crossover of the MP2 and CCSD single fragment error, which occurs at 10^{-4} a.u.i (10^{-5} a.u.) for the virtual (occupied) partitioning. This demonstrates that a determination of the fragment sizes at the MP2 level can lead to increased errors at the CCSD level, as will be shown numerically in Section 6.4. For the calculations presented in Figure 6.12, we find that a part of the reason for the early crossover of the MP2 and CCSD error decays is due to mechanism 3, as shown by the lower crossover of the MP2(1) and CCD(1,2) models.

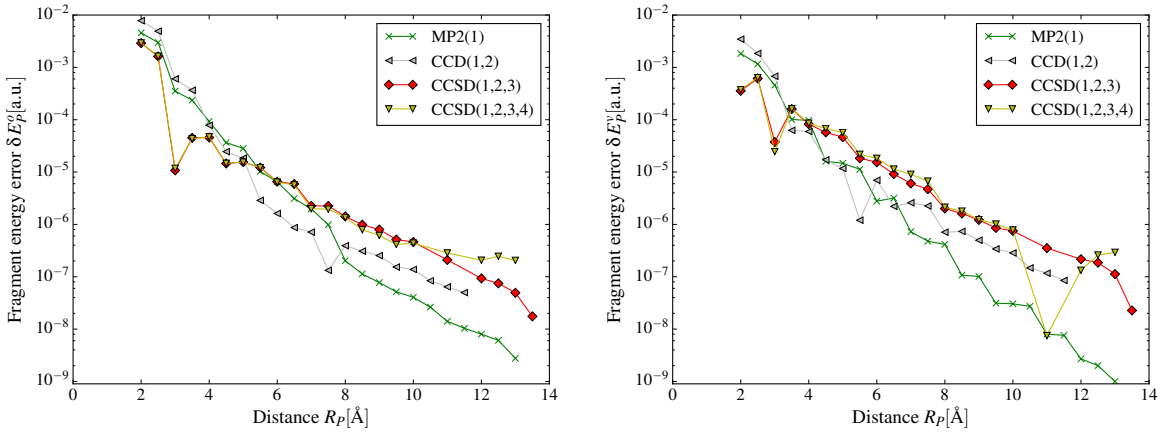
In summary, the theoretical analysis of the present section in combination with Figs. 6.11 and 6.12 demonstrates that for some practical calculations MP2 and CCSD calculations may employ the same local orbital spaces $[P]_{\mathrm{EFF}}$. However, the stronger coupling effects of the CCSD equations (mechanisms 3 and 4) makes it practically impossible to achieve fragment energies more accurately than $\approx 10^{-6}$ a.u., and even at lower accuracies, the precision of a fragment calculation may not be guaranteed. On the other hand, upon lowering the FOT, the precision of the overall calculation is guaranteed to be better. How the fragments may be optimized, to obtain the CCSD single fragment energy with the desired *rigorous* error control, is work in progress.

6.2.3 The DEC–CCSD(T) method

As discussed in the introduction, the contractions of the conventional formulation of the (T) in Eq. (1.57) are limited to the canonical HF basis. Thus, as the local nature of the triples amplitudes cannot be explored within this highly nonlocal basis, locality considerations cannot be used to reduce the overall computational scaling of the (T) energy correction whenever the formulation in Eq. (1.57) is used. On the other hand, the alternative expression in Eq. (1.63) is valid in any basis of optimized HF orbitals, be that a canonical or a local basis, and it despite its increased cost it is therefore beneficial to work with Eq. (1.63). Since the intermediates in Eq. (1.64) are constructed from triples amplitudes and MO integrals only, it is possible to take advantage of the locality of these two quantities when evaluating T_{ij}^{ab} and T_i^a and, in return, $E_{(\mathrm{T})}$ in a local basis. We will now show how Eqs. 1.63 and 1.64 can be used to reduce of the overall computational scaling of the (T) energy correction.

In Eq. (6.1) and Eqs.. (6.2)-(6.5), the CCSD correlation energy is written in such a way that the summations over the two occupied orbitals, i and j, in Eq. (1.32) are replaced by summations over the atomic sites P and pairs of atomic sites P, Q and summations over orbitals i, j assigned to the respective atomic sites. In this section, we will formulate the (T) correction to the CCSD energy in a similar way, and give a short rationale for why the (T) correction can be restricted to the AOS identified at the CCSD level of theory.

Figure 6.12: Error in single fragment energy for the occupied (left) and virtual (right) partitioning schemes as function of expansion radius for MP2(1), CCD(1,2), CCSD(1,2,3), and CCSD(1,2,3,4). Green/gray/red/yellow: absolute MP2(1)/CCD(1,2)/CCSD(1,2,3)/CCSD(1,2,3,4).



(a) Occupied partitioning **(b)** Virtual partitioning

As the (T) correction in Eq. (1.63) contains similar summations over occupied and virtual orbitals, we may likewise substitute these by summations over atomic sites P and pair sites P, Q as in Eq. (6.1)

$$E^{(\mathrm{T})} = \sum_P [E_P^{(\mathrm{T})} + \sum_{Q<P} E_{PQ}^{(\mathrm{T})}], \tag{6.33}$$

where the triples single fragment energy, $E_P^{(\mathrm{T})}$, and triples pair interaction energy, $E_{PQ}^{(\mathrm{T})}$, are given as

$$E_P^{(\mathrm{T})} = E_P^{[4]} + E_P^{[5]}, \tag{6.34a}$$
$$E_{PQ}^{(\mathrm{T})} = E_{PQ}^{[4]} + E_{PQ}^{[5]}. \tag{6.34b}$$

Since the CCSD doubles amplitudes are similarly restricted in space as the two electron integrals (see Figures 6.2b and 6.9) we may restrict the summations in the fourth and fifth order (T) energy contributions may be restricted. The expression for the local (T) fourth order single fragment and pair fragment energy contribution then read

$$E_P^{[4]} = 2 \sum_{ij \in \underline{P}} \sum_{ab \in [\overline{P}]} u_{ij}^{ab} T_{ij}^{ab}, \tag{6.35a}$$
$$E_{PQ}^{[4]} = 2\Big(\sum_{\substack{i \in \underline{P} \\ j \in \underline{Q}}} + \sum_{\substack{i \in \underline{Q} \\ j \in \underline{P}}}\Big) \sum_{ab \in [\overline{P} \cup \overline{Q}]} u_{ij}^{ab} T_{ij}^{ab}, \tag{6.35b}$$

while the fifth-order terms become

$$E_P^{[5]} = 2 \sum_{a \in \overline{P}} \sum_{i \in \underline{P}} t_i^a T_i^a, \tag{6.36a}$$

$$E_{PQ}^{[5]} = 2\Big(\sum_{\substack{a \in \overline{P} \\ i \in \underline{Q}}} + \sum_{\substack{a \in \overline{Q} \\ i \in \underline{P}}} \Big) t_i^a T_i^a, \tag{6.36b}$$

with the T_{ij}^{ab} and T_i^a intermediates given in Eq. (1.64).

Comparing the CCSD and (T) fragment and pair interaction energies, we note how the energies in Eq. (6.33) and in Eq. (6.1) are split in the same way. Furthermore, we note how the fourth order term in Eqs. (6.35) and (6.36) is partitioned in analogy with the occupied partitioning in Eqs. (6.6) and (6.7), that is the summations over occupied orbitals are replaced by summations over the atomic sites and pairs of atomic sites to which the local HF orbitals have been assigned. Also note that a virtual partitioning of the fourth order contribution is possible, but for simplicity we will not go into the details. For the fifth-order term in Eq. (6.36), however, we have chosen a partitioning in which both single excitation indices refer to the same atomic site for the single fragments, while for the pair interaction energies, the indices are assigned to different atomic sites.

In practice, the actual spaces required for the accurate evaluation of the (T) correction will be constrained to the truncated S_{EFF}^P spaces used in the preceding calculation of the CCSD correlation energy due to two primary reasons. First, as the fourth- and fifth-order contributions to the correction are constructed from CCSD singles and/or doubles amplitudes, these will only be satisfactorily described within truncated fragment spaces suitable for the CCSD model. Second, the total (T) correction is known from the literature to be roughly an order of magnitude smaller than the CCSD correlation energy [9], and the errors introduced from reusing the CCSD fragment spaces are thus expected to be minor compared to the intrinsic FOT error. As a result, we will refrain from attempting to improve upon the underlying S_{EFF}^P spaces obtained for the CCSD model when evaluating the (T) correction. Furthermore, we will also use union spaces in the calculation of (T) pair interaction energies. Both of these assumptions will be numerically justified from the DEC-CCSD(T) results in Section 6.4.2. Also, we will demonstrate numerically that the (T) correction to the pair fragment energy contribution E_{PQ} will decay with the distance between centers P and Q as R_{PQ}^{-6}, as a dispersion contributions.

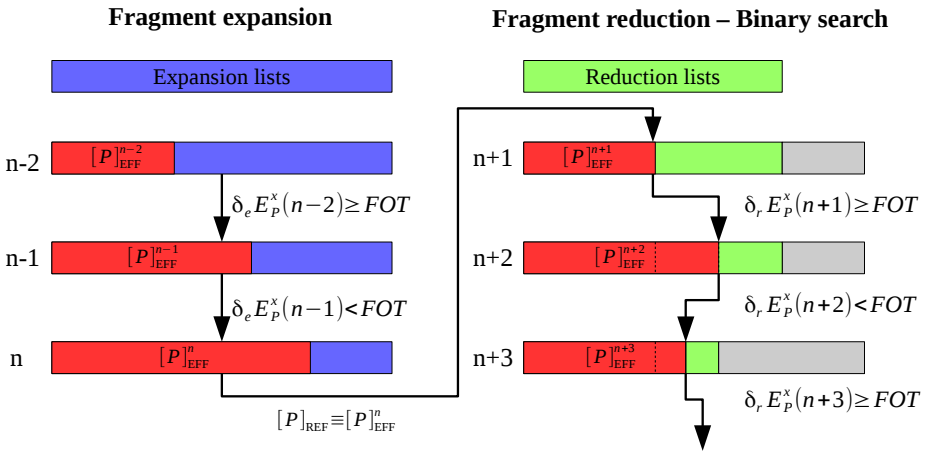
6.3 Implementation of DEC single fragment and pair fragment calculations

In the previous section we have seen how the spaces of single fragment and pair fragments may be chosen in order to maintain rigorous error control in a DEC calculation. In Section 6.3.1 we develop a black box algorithm that determines the single fragment spaces such that fragment energies with the requested precision are obtained. In Section 6.3.2 we describe an energy–based screening technique which ultimately renders the DEC algorithm linear scaling and which removes many of the expensive pair fragment calculations.

6.3.1 The single fragment optimization algorithm

The theoretical analysis in Section 6.2 shows how the single fragment orbital spaces can be chosen in order to maintain rigorous error control in a DEC calculation. In this section we describe

Figure 6.13: The fragment optimization algorithm consists of two steps. In the fragment expansion procedure (left) a reference orbital space is determined based on Eq. (6.37). In the reduction procedure (right) this reference orbital space is fine-tuned using a binary search algorithm where the step direction is based on Eq. (6.40). Red: Orbitals included in the current fragment. Blue: Distance-based list of orbitals. Green: Energy-based list of orbitals. Grey: Discarded orbitals.



a practical implementation for determining the single fragment energies to the predefined FOT precision.

The fragment optimization algorithm for the single fragment energy E_P^x includes two main steps: the fragment expansion procedure where $[P]_{\mathrm{EFF}}$ is increased until the difference between the last two fragment energies is lower than the FOT, and the reduction procedure where the information obtained during the expansion step is utilized to fine-tune $[P]_{\mathrm{EFF}}$ without compromising the FOT precision.

Expansion of the orbital space

In the first step of the fragment optimization algorithm two lists of occupied and virtual orbitals are constructed where the orbitals are ordered according to a measure that estimates their contribution to the single fragment energy E_P^x. Since we use local HF orbitals a simple yet reasonable measure is the distance between the orbital's center of charge and the center P of the fragment. In each expansion step a predefined number of orbitals (*vide infra*) is added to the fragment orbital space $[P]_{\mathrm{EFF}}$. The fragment expansion is illustrated in Figure 6.13 (left). The energy of each expansion step is evaluated at the requested level of theory using Eqs. (6.6) and (6.8), and the energy difference between two subsequent steps i and $i+1$ is calculated,

$$\delta_e E_P^x(i) = |E_P^x(i) - E_P^x(i+1)| \qquad x = o \text{ and } x = v, \tag{6.37}$$

The expansion procedure continues until $\delta_e E_P^x(n-1) < FOT$ and the energy contribution from orbitals not considered in step n is therefore assumed to be below the requested precision. This is a reasonable assumption due to the rapid decay of the single fragment energy with respect to the inclusion of additional orbitals, see, e.g., Figure 6.11. We consider the $[P]^n_{\mathrm{EFF}}$ space of

expansion step n to be the reference space for the reduction procedure and rename it as $[P]_{\mathrm{REF}}$. In this way all the information acquired during the expansion procedure is used in the reduction procedure.

The number of orbitals added in each expansion step needs to be large enough to avoid false convergence and small enough to prevent the orbital space $[P]_{\mathrm{EFF}}$ from becoming unacceptably large. In practice we include $5 \cdot n_{\mathrm{occ}}/n_{\mathrm{atoms}}$ occupied and $5 \cdot n_{\mathrm{virt}}/n_{\mathrm{atoms}}$ virtual orbitals in each expansion step, where n_{atoms} is the number of atoms while n_{occ} and n_{virt} are the numbers of occupied and virtual orbitals in the molecule, respectively. Extensive testing has shown that this choice is a reasonable compromise between accuracy and computational cost.

Reduction of the orbital space

The reduction step defines the final single fragment orbital spaces and thus the pair fragment orbital spaces which are obtained as unions of single fragment orbital spaces (see Appendix B.2). Since the pair fragment calculations dominate the total DEC calculation it is crucial to reduce the single fragment sizes as much as possible within the chosen FOT precision to reduce the cost of the total calculation. To achieve this, a binary search algorithm is used in connection with a priority list of orbitals where the energy of the expanded fragment, $E_P^x(n)$, serves as a reference. Using the energy expression of the occupied partitioning scheme in Eq. (6.6) and the information obtained for the converged fragment space $[P]_{\mathrm{REF}}$, we may define a measure of the energy contribution of a given virtual orbital a,

$$\epsilon_P^a = \left| \sum_{ij \in \underline{P}} \sum_{b \in [\overline{P}]_{\mathrm{REF}}} \tau_{ij}^{ab} L_{iajb} \right|. \tag{6.38}$$

Similarly, based on the energy expression of the virtual partitioning scheme in Eq. (6.8) we define a measure for the energy contribution of an occupied orbital i,

$$\epsilon_P^i = \left| \sum_{ab \in \overline{P}} \sum_{j \in [\underline{P}]_{\mathrm{REF}}} \tau_{ij}^{ab} L_{iajb} \right|. \tag{6.39}$$

Using the measures in Eqs. (6.38) and (6.39) we define priority lists for the occupied and virtual orbitals which are then used to remove the least important orbitals of the $[P]_{\mathrm{REF}}$ orbital space in the reduction procedure, see Figure 6.13 (right). After each step j in the binary search algorithm the energy is evaluated and compared to the energy of the reference single fragment,

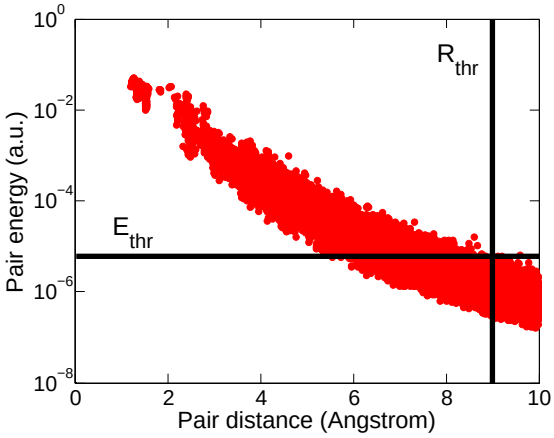
$$\delta_r E_P^x(n+j) = |E_P^x(n+j) - E_P^x(n)|, \qquad x = o \text{ and } x = v, \tag{6.40}$$

where $E_P^x(n)$ is the reference energy from the expansion step. The step is accepted if

$$\delta_r E_P^x(n+j) < \mathrm{FOT}, \qquad x = o \text{ and } x = v, \tag{6.41}$$

and in this case the next step will further decrease the fragment size. On the other hand, if $\delta E_P^x \geq \mathrm{FOT}$ for $x = o$ or $x = v$, the next step will increase the fragment size, see Figure 6.13. This process is repeated until the orbital space $[P]_{\mathrm{EFF}}$ is converged, and the final fragment orbital space is the one that corresponds to the last accepted step of the reduction procedure. We note that relaxation effects are neglected in the energy contribution measures in Eqs. (6.40) and (6.41), and this is the reason why it is necessary to reevaluate the single fragment energy for the occupied and virtual partitioning schemes in each step of the fragment reduction procedure.

Figure 6.14: The pair-energy contributions in an arbitrary DEC calculation are plotted as red dots for each pair-fragment against the distance between the centers. Using an energy–based cutoff E_{thr} (neglecting pairs below the horizontal line), allows for removing more pairs than a (conservatively chosen) real–space cutoff R_{thr}, (neglecting pairs to the right of the vertical line).



Summary and cost reduction considerations

The fragment optimization algorithm in Figure 6.13 is by no means unique but it is an easily programmable approach which ensures that each single fragment energy has a precision corresponding to the input FOT, which ultimately defines the precision of the total calculation.

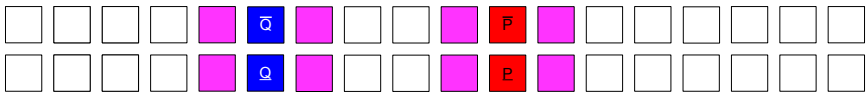
We also note that the fragment optimization may be performed at a different level of theory than that of the target model. For example, for a DEC–CCSD calculation it is possible to determine the fragment orbital spaces by applying the procedure in Figure 6.13 at the MP2 level. We may furthermore reduce the cost of the fragment reduction procedure by considering the procedure converged when only a fraction of the original orbitals is left in the search interval. In practice, the fragment reduction algorithm will return a fragment when the final search interval is 5% of $[P]_{\mathrm{REF}}$.

As described in Section 6.2.2 using MP2 for the fragment optimization is a feasible approach for a typical FOT (e.g. FOT=10^{-4} a.u.). However, for a reliable DEC–CCSD energy, it is required that the CCSD model is also used for the fragment optimization algorithm in Figure 6.13. Due to the sign changes of the CCSD error, a fragment optimization using only the CCSD energy as reference may lead to false convergence, and in that case it may be necessary to check both the MP2 and CCSD energies for convergence. No data to support this option have been obtained yet, as it is currently work in progress to find a thorough procedure for the CCSD fragment optimization. The CCSD fragment optimization is quite expensive, but it may be possible to reduce the cost of the fragment optimization by invoking approximate methods e.g. resolution-of-the-identity (RI) techniques [103], or tensor–hyper–contractions [104], only to determine the AOS.

6.3.2 Screening of pair fragment energy contributions

In the previous sections we discussed in detail how single fragment orbital spaces may be chosen to obtain the single fragment energy to a predefined tolerance for any of the model WFs MP2,

Figure 6.15: An estimation of the pair energy contribution E_{PQ} may be obtained by MP2 calculations where $[P]^{\text{esti},x}_{EFF}$ is chosen such that the nearest neighbors of the sites are included and the calculation is performed in $[P \cup Q]^{\text{esti}}_{EFF}$ and provides the estimate $E^{\text{esti},x}_{PQ}$.



CCSD, or CCSD(T). We dedicate this section to the development of an efficient pair screening method based on energy estimates to reduce the number of pair calculations in DEC.

In a DEC calculation the pair fragment energy contributions decay as R^{-6}_{PQ} (see Figure 6.14), and distant pair energy contributions are therefore negligible. Previously, a real-space cutoff was used to remove distant pair calculations and render the DEC method linear-scaling. In order to preserve error control, this cutoff had to be chosen conservatively and as a result far too many pair-contributions had to be calculated. The real-space cutoff did neither provide a rigorous measure of choosing the necessary contributions to obtain the energy correctly nor did it provide an effective screening method to remove pairs at similar distances but with very different energy contributions (see Figure 6.14). An energy-based cutoff, would therefore allow for a rigorous screening technique that removes many more pairs than a real–space cutoff without any loss of precision. In this subsection we focus on developing an energy–based screening technique, which allows for efficient characterization of the energy contributions of pair fragments and whether a given pair-fragment has to be calculated or may be skipped.

Error of the neglected pairs

In order to rationalize the screening technique, it is convenient to reformulate the DEC correlation energy expression from Eq. (6.1) as

$$E^x_{corr} = \sum_P^N [E^x_P + \frac{1}{2}\sum_{Q\neq P}^N E^x_{PQ}], \tag{6.42}$$

where N is the number of fragments. The errors in the final correlation energy δE_{corr} may then be evaluated as

$$\delta E^x_{corr} = \sum_P^N [\delta E^x_P + \frac{1}{2}\sum_{Q\neq P}^N \delta E^x_{PQ}]. \tag{6.43}$$

Conceptually, the single fragment energy E_P is determined in a DEC calculation such that $\delta E^x_P \leq$ FOT and from the discussions in Section 6.2 and Ref. 38 we know that $\delta E^x_{PQ} \leq$ FOT and that both E^x_{PQ} and δE^x_{PQ} decay as R^{-6}_{PQ}. In order to have a balanced error between the single fragment and pair fragment contributions in Eq. (6.43), we may choose to neglect all contributions E^x_{PQ} such that neglecting them introduces a cumulated error the size of the FOT.

Pair energy estimates for efficient screening

A priori, the contributions of a pair fragment E^x_{PQ} are unknown, but it turns out that order-of-magnitude estimates $E^{\text{esti},x}_{PQ}$ of these may be obtained by MP2 calculations on pair fragments that are composed of the centers P and Q and orbitals on neighboring sites (see Figure 6.15). These

calculations typically recover $80-95\%$ of the pair fragment correlation energy (see Section 6.4) and thus provide a useful estimate of E_{PQ}.

The cutoff in the pair calculations is then obtained by ordering the estimates according to their absolute size, with the largest contribution first, $|E_{P1}^{\text{esti},x}| \leq |E_{P2}^{\text{esti},x}| \leq \cdots \leq |E_{PN}^{\text{esti},x}|$ and exclude all the small pair contributions which have an accumulated estimated contribution of size FOT to the correlation energy,

$$\max_I \left(\frac{1}{2} \sum_{\substack{Q=I \\ Q \neq P}}^{N} E_{PQ}^{\text{esti},x} \right) \leq \text{FOT}, \tag{6.44}$$

where N is the number of single fragments in a DEC calculation. The corresponding I fulfilling the requirement in Eq. (6.44) will be referred to as I_{max}^P. All the pair calculations for the set $\{(P,Q)|1 \leq Q \leq I_{max}^P, Q \neq P\}$ are then carried out, and the total error in a DEC calculation thus becomes a constant times the FOT. Please note that a pair fragment involving the atomic centers P and R may have to be calculated from the perspective of P, but may have been excluded according to Eq. (6.44) from the perspective of R. In any of this case, the pair fragment calculation will have to be carried out and included in the final correlation energy. Note that in practice, we only use the occupied partitioning scheme to obtain the estimates $E_{PQ}^{\text{esti},o}$.

6.4 Numerical illustrations

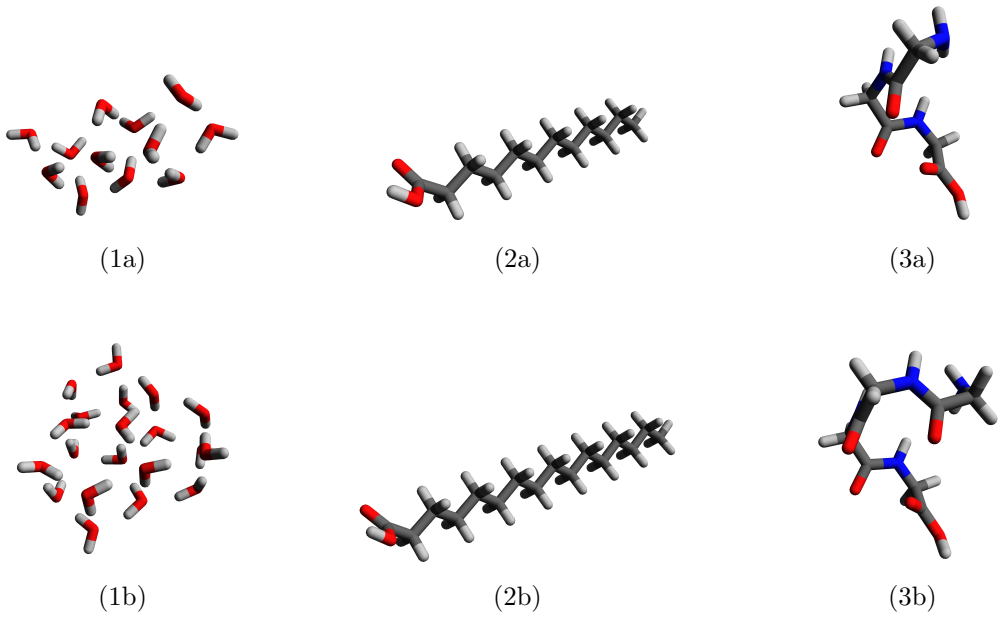
The algorithm devised in Figure 6.13 is a black-box method that operates based on the locality considerations of Section 6.2. In Section 6.4.1 we numerically demonstrate that this algorithm leads to single fragment energies with the requested precision. Then in Section 6.4.2, we demonstrate the overall error control for the DEC–MP2, DEC–CCSD and DEC–CCSD(T) models based on the fragment optimization and pair–screening algorithms of Section 6.3. Finally in Section 6.4.3 we will analyze the pair interaction energy for the DEC–CCSD(T) model. All of the calculations have been performed the set of the molecules presented in Figure 6.16, where calculations involving any molecule labelled with a) used the cc-pVTZ basis set [28], and calculations involving any molecule labelled with b) used the cc-pVDZ basis set [28]. All calculations used the frozen core approximation and were carried out using a local development version of the **LSDALTON** program [29, 105].

6.4.1 Fragment Optimization

When evaluating the precision of single fragment energy calculations we compare each fragment energy, E_P^x (from Eqs (6.6) and (6.8)), obtained from an orbital space determined by the fragment optimization procedure in Figure 6.13 to the single fragment energy obtained in a full molecular calculation, $E_P^{x,f}$ (from Eqs (6.2) and (6.4)). This investigation is therefore limited to calculations for which the standard calculation is feasible at the MP2 and CCSD levels of theory, which is the reason for the limited size of the test systems in Figure 6.16.

In Figure 6.17, we show the distributions of the numerical values of the single fragment energy errors $|\delta E_P^x| = |E_P^x - E_P^{x,f}|$ and the mean values $\chi = \frac{\sum_P |\delta E_P^x|}{N}$ (where N is the total number of fragments considered) separately for the occupied ($x = o$) and virtual ($x = v$) partitioning schemes. The orbital spaces $[P]_{\text{EFF}}$ were optimized at the MP2 level of theory

Figure 6.16: Structures of the systems considered. Two clusters consisting of twelve (1a) and twenty (1b) water molecules; dodecanoic (2a) and hexadecanoic (2b) acid; three (3a) and four (3b) glycine residues in an α-helix structure.



(1a) (2a) (3a)

(1b) (2b) (3b)

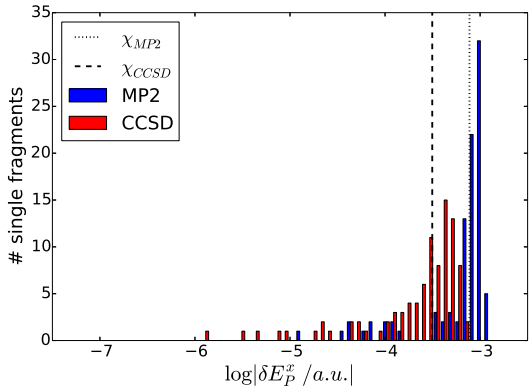
using the algorithm in Figure 6.13 for different FOT values. Single fragment energy errors are reported for all the systems displayed in Figure 6.16.

In general, the fragment optimization procedure leads to errors that are of the requested precision, but errors slightly larger than the FOT may occur, because the reference single fragment energy $E^x_{P,n}$ in the fragment expansion procedure in Figure 6.13 solely relies on the steep decay of the fragment energy with the amount of orbitals included. Therefore converge prematurely if not enough orbitals have been included in a step. For MP2, this can lead to fragment energy errors, which are slightly above the FOT. For high precision CCSD calculations, the single fragment energy converges slower than the corresponding MP2 single fragment energy (see Figure 6.12). As a result of the fact that the fragments shown in Figure 6.17 were optimized at the MP2 level, the CCSD errors may be larger than the corresponding MP2 errors. This may be seen for the occupied partitioning and a FOT of 10^{-5} a.u. in Figure 6.17e. Even more pronounced is the effect for the virtual partitioning, where the separation of the MP2 and CCSD single fragment errors begins already at a FOT of 10^{-4} a.u. in Figure 6.17d. In fact, the fragment with the biggest error in Figure 6.17f, is the one shown in Figure 6.12.

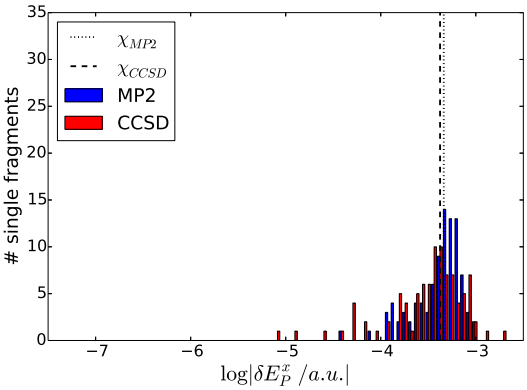
Also, smaller single fragment errors occur, i) because the binary search is not converged to the last possible orbital, ii) the quality of the priority list of orbitals may be imperfect (one orbital may prevent the removal of other less important orbitals), and iii) a cancellation of errors in the CCSD case when the fragment is expanded. The latter error cancellation occurs because the CCSD single fragment energy errors can have either sign while the MP2 errors are always positive.

In Section 6.2 we discussed how CCSD became more long ranged than MP2 at 10^{-7} a.u. However, the data presented in Section 6.2 was based on one single fragment calculation with a

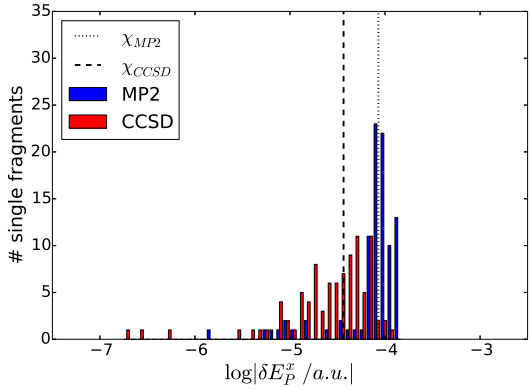
Figure 6.17: Distribution of the absolute single fragment energy errors $|\delta E_P^x|$ in the MP2 (blue) and CCSD (red) for the occupied ($x = o$, left) and virtual ($x = v$, right) partitioning schemes, with respect to calculations in the full orbital spaces and their mean values χ_{MP2} (dotted) and χ_{CCSD} (dashed). The DEC single fragment orbital spaces have been optimized at the MP2 level of theory for a FOT of 10^{-3} a.u. (6.17a and 6.17b), 10^{-4} a.u. (6.17c and 6.17d) and 10^{-5} a.u. (6.17e and 6.17f) for all systems of Figure 6.16.



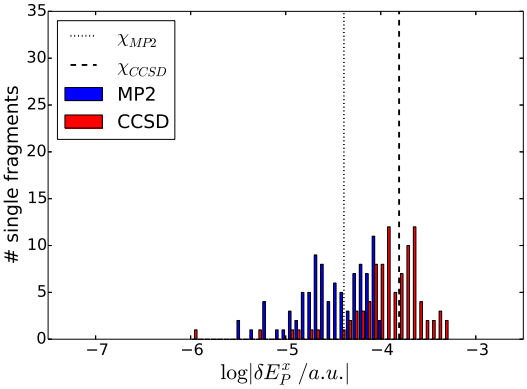
(a) Distribution of single fragment energy errors for a FOT of 10^{-3} a.u. for the occupied partitioning.



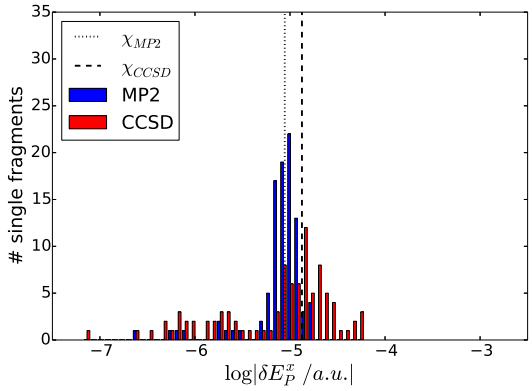
(b) Distribution of single fragment energy errors for a FOT of 10^{-3} a.u. for the virtual partitioning.



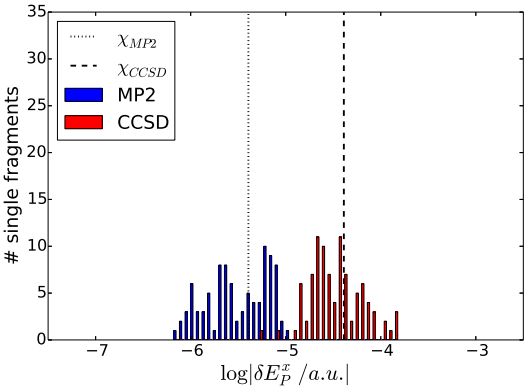
(c) Distribution of single fragment energy errors for a FOT of 10^{-4} a.u. for the occupied partitioning.



(d) Distribution of single fragment energy errors for a FOT of 10^{-4} a.u. for the virtual partitioning.



(e) Distribution of single fragment energy errors for a FOT of 10^{-5} a.u. for the occupied partitioning.



(f) Distribution of single fragment energy errors for a FOT of 10^{-5} a.u. for the virtual partitioning.

6-31G basis set, only. Based on the larger sampling presented in Figure 6.17 it seems reasonable to use MP2 optimized fragments for a CCSD calculation at a FOT of 10^{-5} a.u. for the occupied partitioning scheme, but it is doubtful, whether doing the same for lower FOT values is justified. For the virtual partitioning, the errors in the single fragment energies are decreased when the FOT is decreased, but not by an order of magnitude, as desired. Thus, the FOT may still be used for tuning the precision of a CCSD calculation for both, the occupied and virtual partitioning schemes, in the sense that, if the FOT is lowered by an order of magnitude, the mean absolute error in the fragment calculations is reduced but not systematically enough for the virtual partitioning scheme. This is undesirable, and we therefore strive to find a more reliable way of obtaining single fragments for DEC–CCSD and DEC–CCSD(T) calculations in the future. The virtual partitioning scheme plays a central role for the calculation of molecular properties and gradients [102], and it is therefore of central importance that the single fragment AOS are determined accurately in the fragment optimization procedure for both the occupied and virtual partitioning schemes.

As a last point of this section, we would like to emphasize the importance of the second part of the fragment optimization algorithm in Figure 6.13, the fragment reduction procedure. When comparing the sizes before and after the fragment reduction for all the fragments under consideration we found that the average reduction of the number of occupied/virtual orbitals is 38.3%/38.8%. Thereby, the reduction was at least 10.8%/14.2% and ranging up to 68.8%/67.5% for the occupied/virtual space. In terms of a V^4O^2 scaling CCSD algorithm that corresponds to a mean cost reduction of each single fragment calculation of about 97%, and even more in terms of the time dominating pair fragment calculations.

In conclusion, the numerical data provided in this section support the theoretical analysis in Section 6.2 and shows that the algorithm in Figure 6.13 is a useful practical implementation for determining local orbital spaces, but it has flaws, which are understood, yet not cured. From a practical point of view an important conclusion is that, for the FOTs of practical interest considered here (10^{-3} a.u., 10^{-4} a.u., and 10^{-5} a.u.), the local orbital spaces determined at the MP2 level of theory can also be used for CCSD calculations if only the occupied partitioning scheme is used. This leads to significant savings compared to carrying out the fragment optimization at the CCSD level of theory, particularly for the largest fragment calculation (the nth step in Figure 6.13), but also introduces some ambiguity, as discussed. We do reiterate that for a DEC–CCSD calculation with higher precision, or when molecular properties or gradients are of interest, an improved version of the fragment optimization needs to be carried out at the CCSD level of theory as described in Section 6.3.1.

6.4.2 DEC–MP2, DEC–CCSD and DEC–CCSD(T) energy calculations

In the present section, we analyze the performance currently implemented models of the DEC hierarchy — DEC–MP2, DEC–CCSD, and DEC–CCSD(T) — and how their precision may be tuned via the FOT towards their respective conventional canonical counterparts — MP2, CCSD, and CCSD(T). Proof of concept investigations are performed the set of molecules/systems from Figure 6.16. For all DEC calculations, the fragments were optimized using the procedure described in Section 6.3, and using the occupied partitioning scheme is considered for the reasons discussed in Section 6.4.1.

In the following, we compare DEC–MP2, DEC–CCSD, and DEC–CCSD(T) correlation energies calculated, using FOT values of 10^{-3} a.u., 10^{-4} a.u., and 10^{-5} a.u., to conventional canonical MP2, CCSD, and CCSD(T) results. The total CCSD(T) correlation energy will be

presented in terms of its constituent parts, the fourth- and fifth-order contributions of Eq. (1.56). Table 6.1 gives conventional CC correlation energies for the molecules/systems listed in Figure 6.16. In Table 6.2, we investigate the performance of the DEC–MP2, DEC–CCSD, and DEC–CCSD(T) models by reporting the total error in the DEC-CC correlation energy, i.e., the difference between the DEC-CC correlation energy ($E^{\text{DEC-CC}}$) and the conventional CC correlation energy for the full molecule (E^{CC}),

$$\delta\text{CC} = E^{\text{DEC-CC}} - E^{\text{CC}}, \tag{6.45}$$

as well as the DEC recovery of the full correlation energy,

$$\Delta\text{CC} = \frac{E^{\text{DEC-CC}}}{E^{\text{CC}}}. \tag{6.46}$$

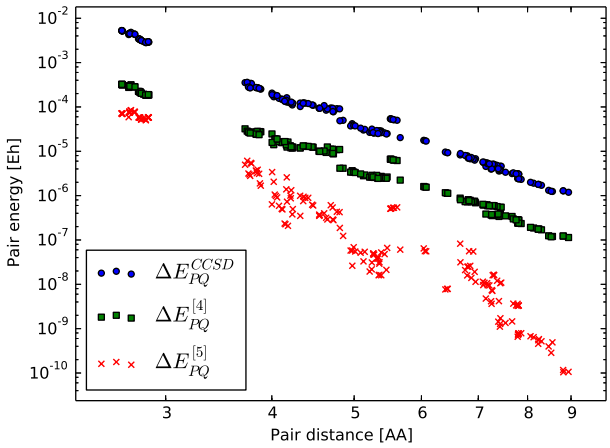
From the results in Table 6.2, we observe a reduction of the total error in the correlation energy (δCC) by roughly an order of magnitude whenever the FOT is reduced by an equal amount. This behavior is observed across all the models and even holds for the (T) energy correction alone, which is consistent with the discussion in Section 6.2, and the data presented in Figure 6.17, as only the occupied partitioning is assessed here. For all of the six test systems in Figure 6.16, the DEC recovery (ΔCC) for a given FOT is independent of the nature of the system in question (homo- or heterogeneous, 1- or 3-dimensional overall structure, dominated by covalent or hydrogen bonds, etc.). At a FOT of 10^{-4} a.u., for example, the DEC–CCSD(T) model succeeds in recovering between 99.83% and 99.96% of the conventional CCSD(T) result. Furthermore, we observe the same pattern for the relative errors in moving from a double-ζ to a triple-ζ basis set.

Upon inspecting the individual entries of Table 6.2, we notice how the CCSD and (T) total errors in the correlation energy are of similar magnitude, indicating that neither of the two dominate the total CCSD(T) error. Phrased differently, no unnecessary efforts have been made at determining any of the two contributions to the total CCSD(T) correlation energy at a higher level of precision than the other. This is found to be true for both loose and tight FOT values. The smaller recoveries for the (T) correction, i.e., larger relative errors, in particular at lower FOT values, naturally arise due to the fact that the (T) correction is more than an order of magnitude smaller than the CCSD correlation energy. The total CCSD errors, however, appear at times to be artificially low, compare, e.g., the MP2 and CCSD errors for systems 1a and 2a at a FOT of 10^{-4} a.u. This occurs because the CCSD single fragment error may have either sign. Finally, we focus on the fourth- and fifth-order contributions to the (T) correction. As mentioned in Section 6.2.3, the fourth-order contribution to the (T) energy correction is typically an order of magnitude larger in size than the fifth-order contribution, and from Table 6.2, we

Table 6.1: Conventional canonical correlation energies (in a.u.) calculated within the cc-pVTZ basis set for systems 1a, 2a, and 3a and the cc-pVDZ basis set for systems 1b, 2b, and 3b.

Model	MP2	CCSD	[4]	[5]	(T)	CCSD(T)
System 1a	−3.22086	−3.27285	−0.10832	0.00313	−0.10519	−3.37803
System 2a	−2.54756	−2.66321	−0.11362	0.00348	−0.11014	−2.77350
System 3a	−2.53798	−2.57194	−0.12294	0.00618	−0.11676	−2.68870
System 1b	−4.19401	−4.34508	−0.08606	0.00381	−0.08225	−4.42733
System 2b	−2.64643	−2.85048	−0.09245	0.00302	−0.08943	−2.93991
System 3b	−2.64774	−2.73911	−0.10103	0.00709	−0.09394	−2.83305

Figure 6.18: CCSD ($\Delta E_{PQ}^{\mathrm{CCSD}}$: blue circles) as well as fourth- and fifth-order (T) ($\Delta E_{PQ}^{[4]}/\Delta E_{PQ}^{[5]}$: green squares and red crosses, respectively) contributions to the CCSD(T) pair interaction energy as a function of the interatomic pair distance for system 1b.



notice how this relationship holds for the total DEC errors as well. For this reason, both the total and relative DEC errors for the (T) correction will be entirely dominated by those for the fourth-order contribution. For the fifth-order contribution, we generally observe smaller total errors, but larger relative errors, than for the fourth-order contribution, due to the fact that the reference fifth-order contributions in Table 6.1 themselves are smaller. However, upon tightening the FOT, the relative fifth-order errors begin to close in at the same level of accuracy as observed for all of the other relative quantities in Table 6.2.

6.4.3 Pair interaction energies, with focus on the CCSD(T) model

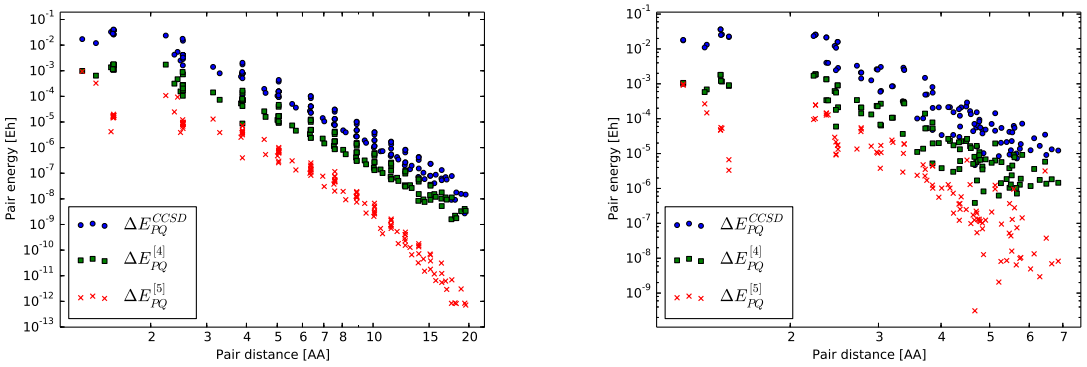
As previously discussed for the MP2 and CCSD models [38, 39], the decay of the pair interaction energies in Eqs. (6.1) and (6.33) with the interatomic distance is an integral part of the DEC method as it allows for a screening of negligible contributions from distant pairs by the procedure described in Section 6.3.2. As mentioned in Section 6.3.2, this screening reduces the scaling of the method from one that depends quadratically on system size to one that depends only linearly. In this section, we provide numerical results that support the theoretical investigations of Section 6.2.3 on the rapid decay of the CCSD(T) triples pair interaction energies in Eqs. (6.35b) and (6.36b) with the interatomic pair distance. In Figure 6.18, we plot the CCSD as well as the fourth- and fifth-order (T) pair interaction energies as a function of interatomic pair distance for system 1b. The results in Figure 6.18 have been obtained by carrying out a CCSD(T) calculation on the full water cluster and subsequently extracting the pair interaction energies using Eqs. (6.3), (6.35b) and (6.36b).

In Figure 6.18, we observe how the CCSD as well as the fourth- and fifth-order (T) contributions all decay rapidly with interatomic pair distance for system 1b. As expected, the fourth-order contributions are roughly an order of magnitude lower than the CCSD energies, while the fifth-order contributions are roughly two orders of magnitude lower in energy. In fact, the fourth-order contributions remain below the corresponding CCSD energies for all pair interaction energies in Figure 6.18, which supports the argument in Section 6.2.3 that the (T) correction has the same distance decay as the CCSD correlation energy. This is further confirmed by similar calculations on systems 2b and 3b shown in Figure 6.19, where the details differ due to the characteristics (i.e. heterogeneity) of the system in question. Thus, the pairs

Table 6.2: DEC-CC (MP2, CCSD, and CCSD(T)) total errors (δCC, in a.u.) and recoveries (ΔCC, in %) with respect to the conventional correlation energies in Table 6.1 for different values of the FOT (in a.u.).

FOT	10^{-3}	10^{-4}	10^{-5}	10^{-3}	10^{-4}	10^{-5}
	System 1a			System 1b		
δMP2	$1.75 \cdot 10^{-2}$	$3.11 \cdot 10^{-3}$	$2.10 \cdot 10^{-4}$	$4.37 \cdot 10^{-2}$	$5.72 \cdot 10^{-3}$	$5.58 \cdot 10^{-4}$
δCCSD	$9.98 \cdot 10^{-3}$	$-5.47 \cdot 10^{-4}$	$-4.58 \cdot 10^{-4}$	$3.57 \cdot 10^{-2}$	$4.84 \cdot 10^{-3}$	$2.87 \cdot 10^{-4}$
δCCSD(T)	$1.70 \cdot 10^{-2}$	$2.42 \cdot 10^{-3}$	$-8.99 \cdot 10^{-5}$	$4.70 \cdot 10^{-2}$	$7.11 \cdot 10^{-3}$	$6.61 \cdot 10^{-4}$
δ(T)	$7.06 \cdot 10^{-3}$	$2.97 \cdot 10^{-3}$	$3.68 \cdot 10^{-4}$	$1.13 \cdot 10^{-2}$	$2.27 \cdot 10^{-3}$	$3.74 \cdot 10^{-4}$
δ[4]	$7.49 \cdot 10^{-3}$	$3.12 \cdot 10^{-3}$	$4.05 \cdot 10^{-4}$	$1.20 \cdot 10^{-2}$	$2.39 \cdot 10^{-3}$	$4.00 \cdot 10^{-4}$
δ[5]	$-4.23 \cdot 10^{-4}$	$-1.54 \cdot 10^{-4}$	$-3.71 \cdot 10^{-5}$	$-7.04 \cdot 10^{-4}$	$-1.18 \cdot 10^{-4}$	$-2.55 \cdot 10^{-5}$
ΔMP2	99.46	99.90	99.99	98.96	99.86	99.99
ΔCCSD	99.70	100.02	100.01	99.18	99.89	99.99
ΔCCSD(T)	99.50	99.93	100.00	98.94	99.84	99.99
Δ(T)	93.29	97.18	99.65	86.29	97.24	99.54
Δ[4]	93.09	97.12	99.63	86.08	97.23	99.54
Δ[5]	86.51	95.10	98.82	81.53	96.91	99.33
	System 2a			System 2b		
δMP2	$4.68 \cdot 10^{-2}$	$4.54 \cdot 10^{-3}$	$5.50 \cdot 10^{-4}$	$4.63 \cdot 10^{-2}$	$6.05 \cdot 10^{-3}$	$6.69 \cdot 10^{-4}$
δCCSD	$2.77 \cdot 10^{-2}$	$6.93 \cdot 10^{-4}$	$-1.37 \cdot 10^{-5}$	$1.76 \cdot 10^{-2}$	$3.40 \cdot 10^{-3}$	$3.79 \cdot 10^{-4}$
δCCSD(T)	$3.78 \cdot 10^{-2}$	$2.59 \cdot 10^{-3}$	$3.49 \cdot 10^{-4}$	$2.90 \cdot 10^{-2}$	$5.11 \cdot 10^{-3}$	$6.29 \cdot 10^{-4}$
δ(T)	$1.01 \cdot 10^{-2}$	$1.89 \cdot 10^{-3}$	$3.63 \cdot 10^{-4}$	$1.15 \cdot 10^{-2}$	$1.71 \cdot 10^{-3}$	$2.49 \cdot 10^{-4}$
δ[4]	$1.04 \cdot 10^{-2}$	$1.96 \cdot 10^{-3}$	$3.66 \cdot 10^{-4}$	$1.20 \cdot 10^{-2}$	$1.77 \cdot 10^{-3}$	$2.55 \cdot 10^{-4}$
δ[5]	$-3.17 \cdot 10^{-4}$	$-6.63 \cdot 10^{-5}$	$-2.87 \cdot 10^{-6}$	$-4.74 \cdot 10^{-4}$	$-6.03 \cdot 10^{-5}$	$-5.14 \cdot 10^{-6}$
ΔMP2	98.16	99.82	99.98	98.25	99.77	99.97
ΔCCSD	98.96	99.97	100.00	99.38	99.88	99.99
ΔCCSD(T)	98.64	99.91	99.99	99.01	99.83	99.98
Δ(T)	90.81	98.28	99.67	87.15	98.09	99.72
Δ[4]	90.81	98.27	99.68	87.06	98.09	99.72
Δ[5]	90.89	98.10	99.92	84.33	98.01	99.83
	System 3a			System 3b		
δMP2	$2.29 \cdot 10^{-2}$	$3.64 \cdot 10^{-3}$	$2.11 \cdot 10^{-4}$	$3.61 \cdot 10^{-2}$	$5.36 \cdot 10^{-3}$	$3.99 \cdot 10^{-4}$
δCCSD	$2.42 \cdot 10^{-4}$	$-9.54 \cdot 10^{-4}$	$-8.81 \cdot 10^{-4}$	$9.33 \cdot 10^{-3}$	$1.16 \cdot 10^{-3}$	$-6.03 \cdot 10^{-4}$
δCCSD(T)	$1.02 \cdot 10^{-2}$	$1.10 \cdot 10^{-3}$	$-5.35 \cdot 10^{-4}$	$2.06 \cdot 10^{-2}$	$3.06 \cdot 10^{-3}$	$-2.81 \cdot 10^{-4}$
δ(T)	$9.92 \cdot 10^{-3}$	$2.05 \cdot 10^{-3}$	$3.46 \cdot 10^{-4}$	$1.13 \cdot 10^{-2}$	$1.89 \cdot 10^{-3}$	$3.22 \cdot 10^{-4}$
δ[4]	$1.09 \cdot 10^{-2}$	$2.26 \cdot 10^{-3}$	$4.15 \cdot 10^{-4}$	$1.26 \cdot 10^{-2}$	$2.20 \cdot 10^{-3}$	$3.90 \cdot 10^{-4}$
δ[5]	$-9.87 \cdot 10^{-4}$	$-2.08 \cdot 10^{-4}$	$-6.98 \cdot 10^{-5}$	$-1.26 \cdot 10^{-3}$	$-3.12 \cdot 10^{-4}$	$-6.76 \cdot 10^{-5}$
ΔMP2	99.10	99.86	99.99	98.64	99.80	99.98
ΔCCSD	99.99	100.04	100.03	99.66	99.96	100.02
ΔCCSD(T)	99.62	99.96	100.02	99.27	99.89	100.01
Δ(T)	91.50	98.24	99.70	87.96	97.99	99.66
Δ[4]	91.13	98.16	99.66	87.56	97.82	99.61
Δ[5]	84.03	96.64	98.87	82.24	95.59	99.05

Figure 6.19: CCSD ($\Delta E_{PQ}^{\mathrm{CCSD}}$: blue circles) as well as fourth- and fifth-order (T) ($\Delta E_{PQ}^{[4]}/\Delta E_{PQ}^{[5]}$: green squares and red crosses, respectively) contributions to the CCSD(T) pair interaction energy as a function of the interatomic pair distance for systems 2b and 3b.



(a) System 2b **(b)** System 3b

which may be screened away in a DEC–CCSD calculation — because their energy contributions are negligible in comparison with the intrinsic DEC error — may also be omitted from a DEC–CCSD(T) calculation without compromising the overall accuracy of the calculation. In fact, since the (T) contribution to the total pair interaction energy is much smaller on average than the corresponding CCSD contribution, one could even consider more elaborate schemes where individual pairs are preordered according to their estimated energy contribution (or interatomic distance) and partitioned into three different levels; at the first level (the smallest energy contributions), the pair interaction energies are neglected altogether, while at the two following levels, they are evaluated at the CCSD and CCSD(T) levels of theory, respectively.

6.5 Summary and outlook

In this section we given a simpler and more general analysis of the locality of the MP2 and CCSD amplitude equations than in Ref. 39 and we have argued for why the (T) correction and the CCSD energy may be obtained from the same space. We have used this knowledge to design an algorithm, which despite its simplicity is able to identify the required fragment spaces to obtain the single fragment energy to within the FOT precision for the occupied partitioning scheme. This was demonstrated numerically for MP2 and CCSD models. Further, it could be demonstrated that the (T) correction may indeed be obtained from the same space as the CCSD energy without compromising the precision of the calculation. For the pair calculations, we have developed an energy–based screening technique to replace the former conservatively chosen real–space cutoff and we have shown that the overall precision of the DEC algorithm is given for all the currently implemented models, DEC–MP2, DEC–CCSD and DEC–CCSD(T), using the current fragment optimization procedure and the pair–screening. Finally we have discussed the decay of the (T) pair interaction energy contributions.

In order to obtain molecular properties and gradients, the virtual partitioning scheme is of central importance, and it is therefore our current research interest to develop a strategy for obtaining the single fragment space for both the occupied and virtual partitioning schemes. We have also given a suggestion for the rigorous determination of these spaces, but no numerical data has been obtained to support this idea at the time of writing.

Appendix B

Details for the identification of single fragment and pair fragment spaces

B.1 Defining locality of molecular orbitals and single fragment extents

Until now, the locality discussion has taken place at the MO level, but the underlying AO basis needs to be truncated i) in order to reduce the cost for the fragment calculations, and ii) to arrive at a truly linear scaling algorithm. It is the purpose of this section is to describe how a set of AOs is selected for a fragment calculation, and how the original MOs are approximated by MOs in the restricted space. The procedure described here is very similar to the one described in Section 6.1 of Ref. 39. However, there are subtle differences, and for completeness the current implementation of the procedure is summarized here.

A localized HF MO ϕ_r^P is assigned to the atomic site P where its atomic Löwdin charge Q^L_{atom} is largest. Even though the bulk of ϕ_r^P is confined to a small volume of space, ϕ_r^P has small expansion coefficients on AOs located some distance away from P. The HF orbital ϕ_r^P,

$$\phi_r^P = \sum_\mu \chi_\mu c^P_{\mu r}, \tag{B.1}$$

may be approximated in the following way

$$\tilde{\phi}_r^P = \sum_{\tilde{\mu}} \chi_{\tilde{\mu}} \tilde{c}^P_{\tilde{\mu} r}, \tag{B.2}$$

where the $\tilde{\mu}$–summation is restricted to AOs which in some sense are neighboring the atomic site P (to be detailed below). The expansion coefficients of $\tilde{\phi}_r^P$ may be determined from a least squares fit, i.e., by minimizing the function $f(\tilde{\mathbf{c}}^P)$,

$$f(\tilde{\mathbf{c}}^P) = \parallel \tilde{\phi}_r^P - \phi_r^P \parallel = \langle \tilde{\phi}_r^P - \phi_r^P | \tilde{\phi}_r^P - \phi_r^P \rangle, \tag{B.3}$$

with respect to the $\tilde{c}$ coefficients. This gives the expansion coefficients

$$\tilde{c}^P_{\tilde{\mu} r} = \sum_{\tilde{\nu}\eta} (\tilde{\mathbf{S}})^{-1}_{\tilde{\mu}\tilde{\nu}} S_{\tilde{\nu}\eta} c^P_{\eta r}, \tag{B.4}$$

where the dimensions of the overlap matrices are defined by the restrictions that are imposed on the AO indices, i.e.,

$$\tilde{S}_{\tilde{\mu}\tilde{\nu}} = \langle \chi_{\tilde{\mu}} | \chi_{\tilde{\nu}} \rangle, \tag{B.5}$$

$$S_{\tilde{\nu}\eta} = \langle \chi_{\tilde{\nu}} | \chi_\eta \rangle. \tag{B.6}$$

We now describe how the $\tilde{\mu}$-summation in Eq. (B.2) may be restricted to exclude the tail region. For each MO ϕ_r^P, a priority list of AOs may be generated by quantifying the importance of each AO χ_μ according to its Löwdin charge Q_μ^L. Going through this list we include AOs until the sum of Löwdin charges is larger than a given threshold δ,

$$1 - \sum_{\tilde{\mu}} Q_{\tilde{\mu}}^L < \delta, \tag{B.7}$$

where δ is a small prefixed number. This procedure defines a set of AOs for each MO ϕ_r^P, which we denote the orbital extent $\{\phi_r^P\}$. The union of orbital extents for all MOs in the effective orbital space $[P]_{\text{EFF}}$ is denoted the *single fragment extent* $\{P\}$. The $\{P\}$ space defines the set of AOs used to describe the MOs in single fragment P, i.e., it defines the restriction on the $\tilde{\mu}$-summation in Eq. (B.2). Thus, all MOs in the fragment are fitted using the same set of AOs to ensure a uniform description. We note that a screening of atomic centers in accordance with Eq. (B.7) was used by Boughton and Pulay [106] for the occupied HF orbitals as a completeness criteria for the assignment of local excitation spaces.

In practice we use $\delta = 0.05$ to define the orbital extents. This might appear to be a very crude value; however, the effect of approximating the MOs is minor because they are fitted using the *union* of all orbital extents. In particular, for the MOs close to site P (large single fragment energy contributions), the fitting procedure has virtually no effect, while it slightly modfies the MOs far from P (small single fragment energy contributions). The fitting procedure therefore has a very minor effect on the single fragment energy. Furthermore, each step of the fragment expansion in Figure 6.13 not only includes new energy contributions and new coupling effects by including more MOs, but it also improves the description of the MOs already included in the previous fragment ($\{P\}$ is enlarged). The effect of the approximation in Eq. (B.2) is thus automatically taking into account by the fragment optimization procedure.

B.2 Pair spaces as unions of spaces

Having identified the spaces that are needed to determine the fragments we can use the same strategy to identify the spaces required to evaluate the pair fragment energies. We will start the analysis with noting that the carrier integrals of the pair fragment EOS amplitudes (following the long–range decay mechanism 3) may be orders of magnitudes smaller than the single fragment EOS amplitudes (see Figure 6.8a). During the iterations of finding the MP2 (or CCSD) solutions, the modifications to the carrier integral contributions are minor, since the coupling effects (Eqs. 6.24) are of higher order in a Møller-Plesset perturbation analysis, and we may therefore conclude that the relative coupling effect on the pair fragment EOS amplitudes is the same as on the single fragment EOS amplitudes.

The pair energy expressions in Eqs. (6.7) and (6.9) allow us to identify the pair fragment EOS

$$S_{\text{EOS}}^{o,PQ} = [\overline{P} \cup \overline{Q}]^2 \times \underline{P} \times \underline{Q}, \tag{B.8}$$

$$S_{\text{EOS}}^{v,PQ} = \overline{P} \times \overline{Q} \times [\underline{P} \cup \underline{Q}]^2, \tag{B.9}$$

where $[P \cup Q] = [P] \cup [Q]$ is the space that interacts with $P \cup Q$ through non-vanishing Fock matrix elements. We henceforth use the generic $S^{x,PQ}_{\mathrm{EOS}}$ for the two spaces. Note that the EOS may include long-ranged on-site excitations, e.g. in the $t^{[\overline{P}][\overline{Q}]}_{\underline{PQ}}$ amplitudes, and short-ranged cross-site excitations, e.g. in the amplitudes $t^{[\overline{Q}][\overline{P}]}_{\underline{PQ}}$,$t^{[\overline{P}][\overline{P}]}_{\underline{PQ}}$, $t^{[\overline{Q}][\overline{Q}]}_{\underline{PQ}}$, which are negligible if $[P] \cap [Q] = \emptyset$. We, again, put the analysis for the occupied and virtual partitionings on an equal footing by introducing the space

$$S_1^{P\cup Q} = [\underline{P} \cup \underline{Q}]^2 \times [\overline{P} \cup \overline{Q}]^2, \qquad (S^{o,P\cup Q}_{\mathrm{EOS}} \cup S^{v,P\cup Q}_{\mathrm{EOS}}) \subset S_1^{P\cup Q}. \tag{B.10}$$

As for the fragment energies, the amplitudes of the first iteration are set to zero

$$t^{ab}_{ij,1} = 0, \tag{B.11}$$

trivially leading to $E^x_{PQ,1} = \Delta E^x_{PQ,1} = 0$ and the residual $R^{ab}_{ij,1} = -g_{aibj}$ is restricted to the EOS $S^{x,PQ}_{\mathrm{EOS}}$. Using Eq. (6.14) we find the amplitudes of the second iteration as

$$t^{ab}_{ij,2} = R^{ab}_{ij,1} \in S^{x,PQ}_{\mathrm{EOS}}, \tag{B.12}$$

which enter the (MP2) residual equation

$$R^{ab}_{ij,2} = -g_{aibj} - \sum_c t^{cb}_{ij,2} F_{ca} - \sum_c t^{ac}_{ij,2} F_{cb} + \sum_k t^{ab}_{kj,2} F_{ik} + \sum_k t^{ab}_{ik,2} F_{jk}; \quad R \in S_2^{P\cup Q}, t \in S_1^{P\cup Q}. \tag{B.13}$$

Space extensions and couplings are again introduced by the mechanisms described in Eqs. (6.24), where the coupling occurs either around center P or around center Q. Using that $[\underline{P}] \times \underline{Q} \subset [\underline{P} \cup \underline{Q}]^2 \supset \underline{P} \times [\underline{Q}]$ we may capture the most extended space analogous to the fragment case (see Eq. (6.26)) by introducing

$$S_n^{P\cup Q} = [\underline{P} \cup \underline{Q}]^2_n \times [\overline{P} \cup \overline{Q}]^2_n, \tag{B.14}$$

where Eqs. (B.14) includes the space extensions for any iteration n as Eq. (B.14) for the single fragment. We initially noted that the carrier integrals of the pair fragment EOS amplitudes were smaller than for the single fragment case, and since it is reasonable to assume that the coupling effects have a similar relative effect on these carrier integrals in both cases, this implies that the pair fragment calculations can be performed in the union of the effective coupling spaces $[P]_{\mathrm{EFF}}$ and $[Q]_{\mathrm{EFF}}$ for fragments P and Q to obtain the pair fragment energy to the predefined precision. Indeed, the error in the pair fragment calculations, for well–separated pairs, is orders of magnitude lower as for the single fragments and decreases with the distance between the atomic centers under consideration as R^{-6}_{PQ} [38].

Chapter 7

Parallelization of the DEC algorithm

This chapter summarizes the work of articles 1

In this chapter, we present an algorithm which combines the local correlation philosophy of the DEC scheme with a massively parallel implementation to enable calculations on large molecular systems to be carried out on modern super computer architectures. The presented algorithm employs three levels of parallelization: (1) the different fragment contributions can be calculated independently (*coarse–grained parallelization*); (2) each individual DEC fragment calculation can be parallelized over several nodes (*medium–grained parallelization*), (3) each node employs OpenMP parallelization over the cores on the node (*fine–grained parallelization*). These three levels of parallelization are combined to yield a dynamic job scheduling scheme where efficient parallelization across thousands of nodes is automatically ensured. The algorithm scales linearly with system size, exhibits near perfect parallel scalability, removes memory bottlenecks, and does not involve any I/O. We avoid the use of I/O to follow the recommendations of the supercomputing community. In short, the presented DEC algorithm removes both time and memory bottlenecks present in conventional CC implementations.

The performance of the algorithm is tested in terms of MP2 energy calculations on two systems containing 528 and 1056 atoms (4278 and 8556 basis functions) using 2945 and 5890 nodes, DEC–MP2 density calculations on a system with 611 basis functions, using 256, 512 and 1024 nodes, and a large–scale, low–precision DEC–CCSD(T) calculation on insulin (4433 basis functions) using 9344 nodes.

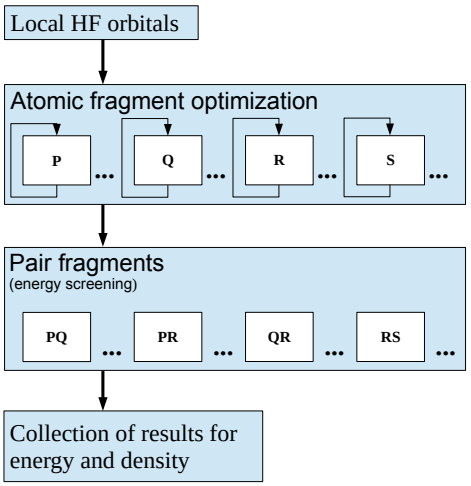
In Section 7.1 we summarize some findings from Chapter 6 and describe how the fragment calculations may be organized such that it is possible to utilize millions of cores on current supercomputer systems. In Section 7.2.1, we summarize the necessary work for a fragment calculation in a DEC context, and finally, in Section 7.3, we evaluate the performance of the DEC-CC algorithm before presenting some concluding remarks in Section 7.4.

7.1 Overview of the DEC scheme

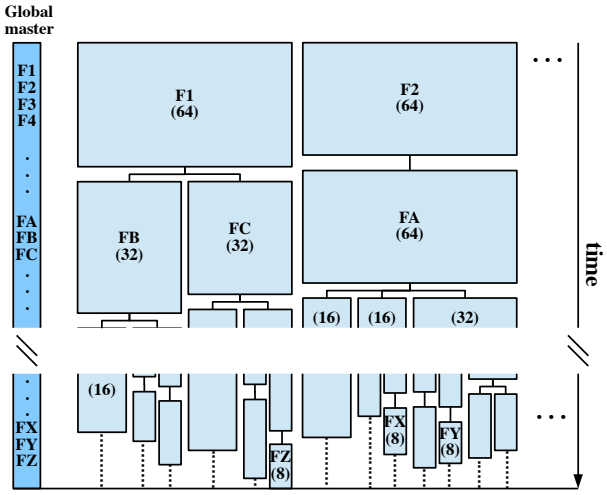
In Figure 7.1a, we give an overview of the DEC scheme. First, a set of local HF orbitals is determined. Second, the orbital spaces entering the single fragment calculations are optimized according to the given FOT as described in Section 6.3. Third, the pair fragment calculations are carried out using unions of orbital spaces from the single fragment calculations and finally, all fragment energy contributions are added up to give the final correlation energy.

In a DEC calculation, the number of single fragments scales linearly with system size, while the number of pair fragments formally scales quadratically with system size. In Section 6.3.2,

Figure 7.1: Overview of the DEC scheme (left) and the parallelization of the time dominating pair fragment calculations (right). Left: First, a set of local orbitals is obtained, then the fragment spaces are generated using the procedure described in Section 6.3, then the pair fragment (and single fragment) calculations are carried out at the desired level of theory, while the collection of results occurs concurrently. Right: In the pair fragment calculations, all the fragment sizes are known from the fragment optimization. The fragment jobs F1, F2, ... are then ordered according to size in descending order on the global master process. A set of computing slots is used to execute the jobs. The number of nodes in each slot is written in a parenthesis. Whenever a slot finishes its job, it asks for a new job from the master. In case the new job is too small to satisfy the efficiency criterion in Eq. (7.1), the slot divides itself into two new independent slots.



(a) Overview of the DEC scheme



(b) Parallelization of the DEC scheme.

we have described an algorithm which enables a prescreening of the pair fragment energies. The number of pair fragments to consider therefore scales only linearly with system size. It is clear from Figure 7.1a that the single fragment calculations as well as the pair fragment calculations may be run independently on different nodes of a large super computer. In Section 7.2.1, we detail how the individual fragment calculations may also be parallelized.

In Figure 7.1a, it is further indicated that the MP2 correlation electron density can be determined using the DEC–MP2 scheme. This evaluation involves the Lagrangian correlation energy as described in Ref. 102, and will not be discussed here.

7.2 Parallelization of the DEC scheme

Let us now discuss how the initially mentioned parallelization levels are combined to yield an efficient dynamic scheme for DEC-CC calculations. In Section 7.2.1, we start with giving some general remarks about the medium– and fine–grained levels of parallelization without going into the details which they are model specific. In Section 7.2.2, we describe the model–independent coarse–grained level of parallelization which is inherent in the DEC method.

7.2.1 The medium–grained level of parallelization in a DEC calculation

The parallelization at the fragment level in a DEC–CC calculation is the same as for conventional CC calculations, and it is beyond the scope of this Section to provide a detailed description of the implementation. It has been discussed in Ref. 42 how an MP2 calculation may be parallelized on the fragment level in order to be conservative with the memory consumption and to obtain the required intermediates for the energy and density. It has been discussed in Part I how a CCSD calculation may be parallelized for the execution numerous nodes, and, in fact, the algorithm described in Chapter 4 is used in a DEC context. It is also beyond the scope of this Section to describe how the (T) correction to the CCSD energy may be parallelized. For the conceptual understanding, it may suffice to say that the kernel of all these models consists of local data sorting and contractions mediated by `dgemm`s, while large quantities may or may not be stored in PDM and loaded using one–sided MPI routines (see Figure 7.2 and Chapter 3).

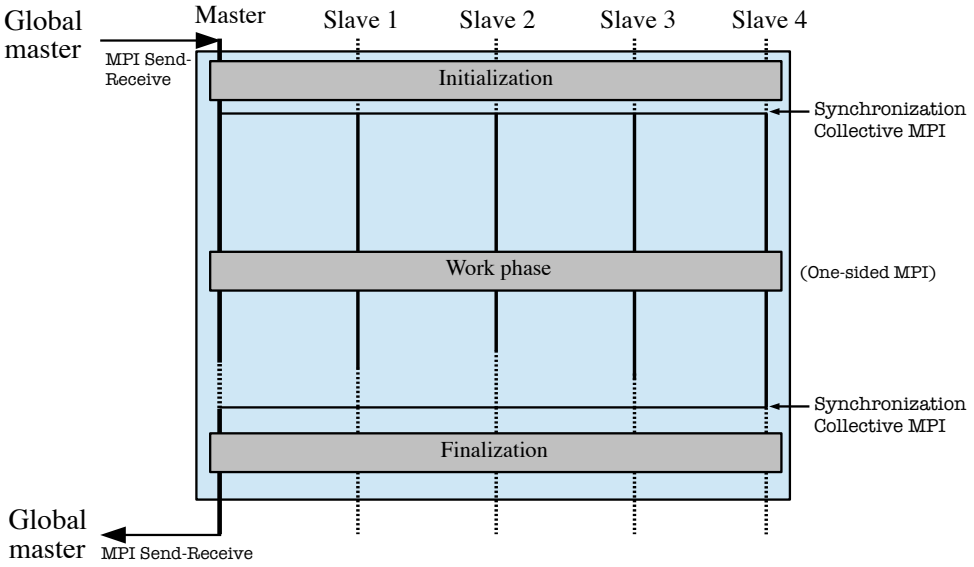
The medium level of parallelization occurs on a subset of all available nodes, referred to as a *slot*. One node in the slot is the *local master* while the rest of the nodes in the slot are the *local slaves*. A fragment calculation is initialized with the local master asking the *global master* node for a job, which involves the communication of two integers in an MPI send–receive style within the *global* MPI group of nodes. Upon receiving the job identifier, the local master sets up the fragment, and wakes up the slaves for the heavy work (Initialization, in Figure 7.2). For all models, there exists a loop with a number of tasks to do (n_{tasks}), which may be distributed among the nodes within a slot. After finishing their assigned chunk of work, the data is collected on the local master, and the local slaves return to their waiting position (Work phase, in Figure 7.2). The local master calculates the (pair) fragment energy (and collects necessary intermediates for the calculation of the density, if requested) and contacts the global master to pass along the fragment data and ask for a new job (Finalization, in Figure 7.2).

In the work phase, a reasonably efficient parallelization is obtained only if the number of tasks (n_{tasks}) is larger than some integer $M_{\text{model}} > 1$ times the number of nodes in the slot (n_{nodes}). The integer M_{model} is a (predefined) heuristic number and model dependent, but in general we may write the efficiency criterion as

$$n_{\text{tasks}} > n_{\text{nodes}} \cdot M_{\text{model}}, \qquad M_{\text{model}} > 1. \tag{7.1}$$

Let us finally comment on the solution of the amplitude equations in Eq. (1.30) for the individual fragment calculations. The locality analysis in Chapter 6 demonstrates that the amplitudes needed to determine the single fragment energies for atomic site P — i.e., E_P^o and E_P^v — may be determined by solving amplitude equations in a local orbital fragment space (see Figure 6.6). The AOS for a pair fragment PQ is obtained as the union of AOSs for single fragments P and Q (see Appendix B.2). For MP2 and the (T) correction, we may avoid solving Eqs. (1.37) and (1.65) using an iterative algorithm, by transforming the orbital space of a given fragment calculation to a fragment–canonical (or pseudo–canonical) basis, in which the fragment Fock matrix assumes a diagonal form. In this basis, the MP2 doubles and (T) triples amplitudes may be calculated using Eqs. (1.39) and (1.58) directly, and then upon transformation to the local basis, the fragment energies may be evaluated. However, also when solving the CCSD equations iteratively, it is beneficial to transform Eq. (1.30) to a fragment–canonical basis, as explained in Chapter 2. When a fragment CCSD calculation has converged, the CCSD amplitudes are transformed to the local basis, where the EOS amplitudes can be extracted and the fragment energy evaluated.

Figure 7.2: Illustration of a fragment job calculated using a slot with 5 nodes (1 local master and 4 local slaves). A job identifier is sent from the global master to the local master, which then performs the calculation together with the local slaves and sends back the calculated fragment data to the global master. The initial and final steps together with a non-ideal job distribution in the work phase lead to a *local loss* of efficiency illustrated with dotted lines.



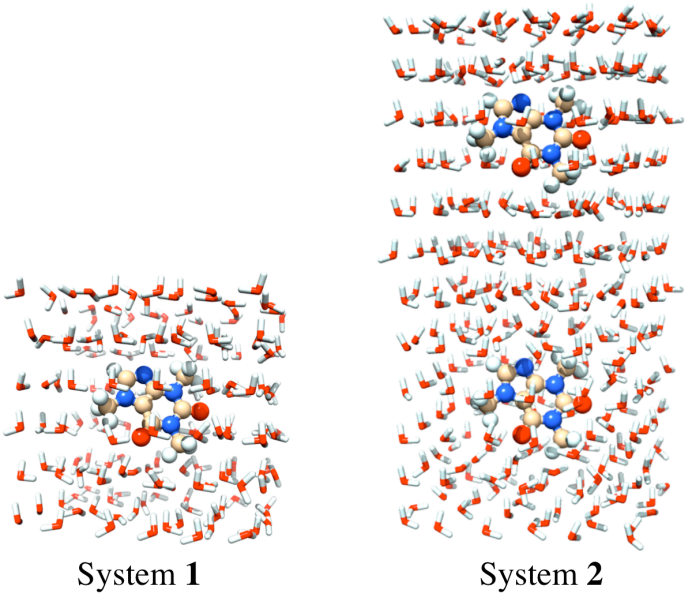
7.2.2 The coarse–grained level of parallelization in a DEC calculation

The coarse–grained level of parallelism in the DEC algorithm is a master–slave approach, where one *global master* node has many associated *slots* (groups of slave nodes) over which independent chunks of work, i.e. fragment calculations, may be distributed. First, the global master orders the fragment calculations according to size with the largest fragments first. This gives the ordering F1, F2, ..., FA, FB, ..., FY, FZ shown in Figure 7.1b. The computational nodes are grouped into slots (of, e.g., 64 nodes) and the master distributes the first fragment calculations in the list to these slots. When a slot finishes its fragment calculation, it sends a message to the master node to receive a new fragment, say, fragment FB. If Eq. (7.1) is not satisfied for fragment FB (i.e., $n_{\text{tasks,FB}} \leq 64M$), then the slot divides itself into two new slots (of, e.g., size 32 each). One of the new slots will execute fragment FB, while the other new slot will ask the master for a new fragment (fragment FC). This scheme is continued until all fragments are done. In this manner, the slots automatically divide such that Eq. (7.1) is satisfied for all fragments to ensure efficient medium–grained parallelization.

It is apparent from Figure 7.1b that there will also be a *global loss* of efficiency at the end of the calculation as shown by the dotted vertical lines. This happens when a given slot has finished a fragment job and all remaining fragment jobs have already been distributed by the global master. The nodes of the finished slot will then have to wait until all other slots are done with their respective jobs.

It remains to point out the weaknesses of this type of job distribution. One major problem is that the global loss may be dominated by a single fragment calculation, if it is very big in comparison to the other fragments. The other main weakness is observed in the limit of very small fragment jobs, where a lot of local masters will request a job at the same time (after all the

Figure 7.3: Molecular systems used for the MP2 energy and density study.



groups are split down to one node each), since then there will be a communication bottleneck in the job distribution as the global master cannot satisfy the requests. It is a current research goal to avoid the second bottleneck by a more dynamic job distribution based on one–sided MPI communication.

In summary, the scheme in Figure 7.1b allows for the resources required for the individual fragment calculations to be adjusted to the actual needs and at a minimal loss of efficiency. In Section 7.3, we discuss the efficiency of DEC-CC calculations on large molecular systems. In particular, we analyze the connection between local and global losses.

7.3 Results

In order to analyze the DEC algorithm developed in this Chapter, we will present individual calculations which have been performed at different levels of theory and on different molecules. For all presented calculations, a development version of the **LSDALTON** [29] program was used.

7.3.1 DEC–MP2 energy and density calculations

The set of pair calculations in Figure 7.1a is the time-dominating step in a DEC–MP2 calculation, and we therefore focus only on this step here. We note that, since the fragment optimizations in Figure 7.1a involve only energy calculations, it is necessary to repeat single fragment calculations along with the pair calculations when also the MP2 density is of interest. Thus, the list of fragments in Figure 7.1b contains both optimized single fragments and pair fragments listed according to size. In the calculations presented, we have used a FOT of 10^{-4} a.u. and instead of the energy–based pair screening procedure described in Section 6.3.2, a distance cutoff of 10 Å was used.

These calculations were carried out on **Titan**, running with two MPI processes per **Titan** node, i.e., 1 MPI process per NUMA node, which then employs OpenMP parallelization over 8 cores (see page 133). Thus, n_{nodes} in Eq. (7.1) refers to the number of NUMA nodes, and for the remainder of the MP2 subsection, the "number of nodes" will implicitly refer to the number of NUMA nodes (which is twice the number of actual **Titan** nodes). We report calculations using 5890 nodes (47120 cores) and 11780 nodes (94240 cores). The accelerators present on **Titan** were not employed.

The systems under investigation are shown in Figure 7.3. System **1** ("nanospresso") contains a caffeine molecule surrounded by 168 water molecules in a box with dimensions $\approx 1.5\text{nm} \times 1.5\text{nm} \times 1.2\text{nm}$, giving a total of 528 atoms (4278 basis functions using the cc-pVDZ basis set). System **2** ("nanospresso doppio") corresponds to two units of system **1** and thus contains 1056 atoms (8556 basis functions). Both geometries were allowed to relax in a force field using the default settings of the Avogadro program (v. 1.0.3) [107].

Time-to-solution

The most important property of a large-scale massively parallel code is that it provides a short time-to-solution (TTS). In our case, this is the time it takes from the first fragment calculation in Figure 7.1b is started to the last fragment calculation is finished (measured by the global master). Table 7.1 gives the time-to-solution for systems **1** and **2** for 5890 and 11780 nodes.

By doubling the number of nodes, the TTS would ideally be halved. In practice, the TTS is decreased by a factor 0.56 (0.93/1.66) for system **1** and by a factor 0.53 (2.37/4.49) for system **2**. Thus, the scheme in Figure 7.1b performs very well with respect to parallel scaling, although some deviation from the ideal behavior is observed.

System **2** is twice as large as system **1** measured by the number of atoms. In the presented calculations, we omitted all pair fragments separated by more than 10 Å. Assuming that all atomic sites P are involved in the same number of pair fragments, this leads to linearly scaling number of pair fragments. However, in practice atoms at the edges of the water box will form fewer pairs than atoms at the center of structure. Since the relative amount of "edge atoms" is larger for **1** than for **2**, the number of pairs to consider for **2** will be slightly larger than twice the number of pairs for system **1**. In practice there are 7136 fragments for system **1** and 16151 ($= 2.3 \times 7136$) fragments for system **2**. Ideally one would thus expect that, for a fixed number of nodes, the TTS is 2.3 times larger for **2** than for **1**. From Table 7.1 it is seen that the actual TTSs for **2** are 2.7 (=4.49/1.66) and 2.5 (2.37/0.93) times larger for 5890 and 11780 nodes, respectively. When increasing the system size, the TTS is thus roughly scaled by the relative increase in the number of fragments.

The key points of this chapter have now been demonstrated: to a very good approximation (1) the scheme in Figure 7.1b allows for the TTS to be decreased in proportion to the number of nodes applied, and (2) the TTS scales linearly with system size. We now analyze the reasons for deviations from ideal behavior.

Local and global losses

Two sources of loss are present in the MPI parallelization of the DEC–MP2 scheme compared to a calculation on a single node (employing OpenMP, but not MPI parallelization): the local loss of the individual fragment calculations illustrated in Figure 7.2 and the global loss at the end of the calculation shown in Figure 7.1b. (Furthermore, there are losses associated with the OpenMP parallelization of each MPI process, which will be discussed in the following section).

Table 7.1: Timings for systems **1** and **2** for different number of nodes. All calculations were carried out using $M = 3$ in Eq. (7.1).

System	#fragments	#nodes[a]	TTS[b](hours)
1	7136	5890	1.66
1	7136	11780	0.93
2	16151	5890	4.49
2	16151	11780	2.37

[a] Number of NUMA nodes. Each NUMA node runs one MPI process which employs OpenMP parallelization over 8 cores.
[b] Time-to-solution for all fragment calculations.

Table 7.2: MPI-associated loss compared to a calculation on a single node (employing OpenMP but not MPI parallelization) for systems **1** and **2** for different number of nodes.

System	#nodes	Local loss[a](%)	Global loss[b](%)	Total loss[c](%)
1	5890	18.4	5.2	23.5
1	11780	17.4	14.6	32.0
2	5890	26.6	1.9	28.4
2	11780	27.0	5.3	32.3

[a] Percentage loss within the fragment slots (Figure 7.2).
[b] Percentage loss associated with idle time at the end of the calculation and MPI communication between global master and local slaves (Figure 7.1b).
[c] Sum of local and global losses.

In Table 7.2, we present the local and global loss for the calculations in Table 7.1. Consider first the calculations for system **1**. Clearly, when the number of nodes is increased, the local loss stays relatively constant, while the global loss increases. This happens because the individual fragment calculations (Figure 7.2) are virtually unchanged, while the global distribution of jobs (Figure 7.1b) inevitably gets less homogenous when the same number of fragments are distributed among twice as many nodes. The same feature is observed for system **2**.

Table 7.2 also reveals the unpleasant feature that the local loss increases when the system size increases. A closer inspection of the individual steps shows that the difference in local loss between system **1** and **2** stems from the fragment initialization (see Figure 7.2), where the fragment data is extracted from data for the full molecule (e.g., the local Fock matrix $\mathbf{F}'_{\text{local}}$ is extracted from the full molecular Fock matrix $\mathbf{F}$). In fact, the fragment initialization for system **2** on average takes four times as long as the similar step for system **1**, which is consistent with the fact that the matrix dimensions for system **2** are twice as large as for **1**.

Clearly there is room for improvements to reduce the local and global losses. However, the main point remains that the massively parallel DEC–MP2 scheme scales very well both with respect to the number of nodes and with respect to system size.

Floating point operations and OpenMP parallelization

We have measured the number of floating points operations (FLOP) for the calculations in Table 7.1 using the PAPI library [108]. The results are given in Table 7.3. As expected the ratio between the total number of petaFLOP for the two systems (486/191=2.5) corresponds roughly to the ratio between the number of fragments (16151/7136 =2.3, see Table 7.1).

Table 7.3 also contains the average number of gigaFLOP per second (GFLOP/s) per core. The ideal value is 2.2GHz $\times$ 4 operations/core = 8.8 GFLOP/s per core (obtained if all cores perform only floating operations during the entire calculation). We are thus roughly an order of magnitude away from the ideal value. This happens (i) due to the local and global loss effects discussed in Section 7.3.1 (MPI parallelization), (ii) because not all operations involved are floating point operations, and (iii) because the OpenMP parallelization is not ideal. To analyze the two latter effects, we report timings and GFLOP/s per core for three selected fragment calculations of different sizes using 1 and 8 OpenMP threads (Table 7.4). In Table 7.4, the time associated with the dotted lines in Figure 7.2 (the local loss) have been subtracted from the effective times when calculating the average GFLOP/s per core. Thus, the timings in Table 7.4 solely describe losses of types (ii) and (iii).

Considering first the calculations using a single OpenMP thread in Table 7.4 (loss type (ii) above), it is seen that about 2-5 GFLOP/s per core is observed for the selected fragments. These numbers deviate from the ideal value of 8.8 GFLOP/s per core because several steps in the MP2 algorithm do not use floating point operations. Such steps include the many data sortings, which ensure that the data is consecutive in memory before the matrix multiplications are performed, and the extraction of fragment data from full molecule data. Table 7.4 also reveals that the GFLOP/s per core is lower for larger fragments. The primary reason for this is the many skinny matrix multiplications, where the summation index runs only over a small set of indices, while the total array is quite large. Such matrix multiplications are not handled as efficiently by `dgemm` as is the case for square matrices. This effect is more pronounced for larger fragments.

Regarding the OpenMP parallelization (loss type (iii) above) the timings in Table 7.4 show a speed-up of about 2.5 for eight OpenMP threads compared to a single OpenMP thread, which is quite far from the ideal speed-up of 8. One reason for this is the node configuration on **Titan**, since there are only 4 floating point units per NUMA node (see page 133) and thus the ideal speedup for a FLOP heavy code is closer to 4 on this system using all cores. In fact, using more than 4 OpenMP threads with a FLOP heavy code may even lead to an efficiency reduction, as the 8 integer cores compete for the 4 floating point units. The effect of this has not been assessed in this study. Other reasons include that the data sorting (at the time running the calculation) and the extraction of fragment data not being OpenMP parallelized, several OpenMP loops that need to be started up and shut down again when calling a series of `dgemm`s, and `dgemm`s that are not OpenMP parallelized as efficiently for the often encountered skinny matrices as for square matrices. For these reasons the GFLOP/s per core is significantly higher for a single OpenMP thread than for 8 OpenMP threads.

The DEC–MP2 energy and density

Although the main focus in this chapter is on the performance of the massively parallel DEC–MP2 scheme in Figure 7.1b, let us briefly discuss the calculated MP2 correlation energy and correlation density for the molecules in Figure 7.3.

As mentioned above, the calculations in this investigation were carried out using a FOT of

Table 7.3: Number of floating point operations for systems **1** and **2** for different number of nodes.

System	#nodes	Tot. petaFLOP[a]	GFLOP/s per core[b]
1	5890	191	0.68
1	11780	191	0.60
2	5890	486	0.64
2	11780	486	0.60

[a] Total number of petaFLOP for all fragment calculations.
[b] Average number of GFLOP per second per core.

Table 7.4: Timings and GFLOP/s per core for three selected fragments using either 1 or 8 OpenMP threads. All calculations were carried out on system **2**.

#occ[a]	#virt[b]	#OpenMP threads	time(s)[c]	GFLOP/s per core[d]
24	121	1	504	5.63
24	121	8	213	1.63
52	162	1	840	3.42
52	162	8	350	1.00
77	351	1	2730	2.10
77	351	8	931	0.77

[a] Number of occupied AOS orbitals in fragment.
[v] Number of virtual AOS orbitals in fragment.
[c] Time measured by local master.
[d] Number of GFLOP per second per core calculated using effective times, where the MPI associated losses in Table 7.2 have been removed (see text).

Table 7.5: Calculated DEC correlation energies (a.u.) and associated error estimates. Also shown is the trace of the correlation density matrix divided by the number of electrons.

System	$E^{\mathrm{o}}_{\mathrm{corr}}$	$E^{\mathrm{v}}_{\mathrm{corr}}$	$\mathcal{L}^{MP2}_{\mathrm{corr}}$	$\Delta E^{\mathrm{DEC}}_{1}$	$\%E^{\mathrm{est}}_{\mathrm{corr}}$	$\mathrm{Tr}\boldsymbol{\rho}/N_{\mathrm{el}}$
1	-37.4246	-37.4299	-37.4240	0.0060	99.98	$-4.0 \cdot 10^{-6}$
2	-74.8824	-74.8943	-74.8819	0.0123	99.98	$-4.2 \cdot 10^{-6}$

Table 7.6: DEC-CCSD(T) calculation for insulin using the 6-31G basis and a 10^{-3} a.u. energy error threshold.

Number of **Titan** cores	149504
Time-to-solution (hours)	9.03
Global MPI loss (%)	0.97
Local MPI loss (%)	2.15
Total MPI loss (%)	3.12

10^{-4} a.u., which controls the *relative* rather than the *absolute* error of the calculated DEC–MP2 correlation energy. We therefore expect calculations carried out using the same FOT value to exhibit roughly the same relative (size-intensive) errors. It is very challenging – if at all possible – to carry out a conventional MP2 calculation on systems as large as **1** and **2**, and the DEC fragmentation errors can therefore not be quantified by comparing to a conventional MP2 calculation, as in Section 6. However, the DEC–MP2 error may be estimated evaluated using

$$\Delta E_1^{\mathrm{DEC}} = \max(E^o_{corr}, E^v_{corr}, \mathcal{L}^{MP2}_{corr}) - \min(E^o_{corr}, E^v_{corr}, \mathcal{L}^{MP2}_{corr}), \tag{7.2}$$

including the Lagrangian contribution $\mathcal{L}^{MP2}_{corr}$. The percentage of the correlation energy recovered by the DEC calculation can be estimated as [109]

$$\%E^{\mathrm{est}}_{\mathrm{corr}} = \frac{|\mathcal{L}^{MP2}_{\mathrm{corr}}| - \Delta E_1^{\mathrm{DEC}}}{|\mathcal{L}^{MP2}_{\mathrm{corr}}|} \cdot 100\%. \tag{7.3}$$

In Table 7.5, we present the calculated correlation energy using the three partitioning schemes along with the estimated errors of Eqs. (7.2) and (7.3) for systems **1** and **2**. Since the correlation energy is a size-extensive property, it is roughly doubled when the system size is doubled. The absolute error ($\Delta E_1^{\mathrm{DEC}}$) also scales with system size because the errors of the individual fragment energy contributions add up when the correlation energy is calculated [109]. Since the absolute error is size-extensive, the percentage of the correlation energy ($\%E^{\mathrm{est}}_{\mathrm{corr}}$) recovered by the DEC calculation is a *size–intensive* property. Systems **1** and **2** therefore recover the same percentage of the correlation energy ($\%E^{\mathrm{est}}_{\mathrm{corr}} = 99.98\%$). Another size-intensive error measure given in Table 7.5 is the trace of the calculated MP2 correlation density matrix $\boldsymbol{\rho}$ divided by the number of electrons, $\mathrm{Tr}\boldsymbol{\rho}/N_{\mathrm{el}}$, which is zero in the limit of a full calculation. Also this quantity is roughly the same for systems **1** and **2**.

It is instructive to compare the results in Table 7.5 to those of a recent DEC–MP2 calculation on the insulin molecule using the same FOT value [109]. For insulin we found $\%E^{\mathrm{est}}_{\mathrm{corr}} = 99.95\%$ and $\mathrm{Tr}\boldsymbol{\rho}/N_{\mathrm{el}} = -7.4 \cdot 10^{-6}$. The relative error measures for insulin are thus of similar size as for systems **1** and **2**, confirming that the FOT defines the relative error of the calculation.

7.3.2 Large scale DEC-CCSD(T) insulin calculation

We have further carried out a DEC-CCSD(T) calculation for the insulin molecule using a 6-31G basis (4433 basis functions), the frozen core approximation, a FOT of 10^{-3} a.u., and the pair screening mechanism was deactivated. The results of this calculation are rather meaningless, but, most importantly, we have been able to demonstrate that the DEC code also performs well

at a higher level of theory, despite the higher complexity of the individual fragment calculations. The calculation was performed using 149504 **Titan** cores (without accelerators) where the time determining part took 9 hours. The results are summarized in Table 7.6. From the scaling behavior of the DEC-MP2 calculations (Table 7.1) and the very small global loss in the MPI parallelization of about 1% in the DEC-CCSD(T) calculations (Table 7.6), we see that the coarse–grained (or global) level of parallelization is indeed independent of the deployed level of theory. For the medium–grained (or local) level of MPI parallelization of a single fragment DEC-CC calculation, ensuring load balancing is inherently more difficult for higher levels of theory and it thus adds up to a larger total loss than the coarse–grained parallelism.

7.4 Conclusion

We have presented a massively parallel algorithm for the calculation of DEC–MP2 and DEC–CCSD(T) correlation energies and DEC–MP2 correlation densities of large molecular systems. The resulting algorithm scales linearly with system size, exhibits near perfect parallel scalability, removes memory bottlenecks, and does not involve any I/O. The algorithm is ideally suited for calculations on large supercomputer architectures as demonstrated by several calculations. Most importantly, the algorithm provides a short time-to-solution for large-scale calculations on super computer architectures.

The presented algorithm involves three levels of parallelization. At the coarse–grained level, the different DEC fragment calculations are distributed among a number of slots each containing a given number of nodes. At the medium–grained level, each slot carries out its own set of fragment calculations, where each fragment calculation is parallelized over the nodes within the slot. At the fine–grained level each node employs OpenMP parallelization to utilize all cores on the node. The fragment calculations are distributed to slots in a dynamic fashion where the slot sizes are automatically adapted to the fragment sizes to ensure efficient use of the resources. Naturally, there are some losses associated with the massively parallel DEC scheme compared to a sequential implementation. These losses have been analyzed and possible improvements have been suggested. However, the main point remains that the massively parallel DEC scheme scales very well both with respect to the number of nodes and with respect to system size, and may be run efficiently independent of the wave function model.

In conclusion, the algorithm satisfies the following criteria which are very important from a practical point of view: (1) for a given molecular system, the time-to-solution is roughly halved when the number of nodes is doubled, (2) for a given number of nodes, the time-to-solution is roughly doubled if the system size is doubled, and (3) the fragmentation errors introduced compared to a conventional calculations are controlled by the FOT and may be made arbitrarily small. The short time-to-solution for a single point calculation is particularly important for structure optimizations where many single point calculations are required.

Chapter 8

Exploring the use pair–natural–orbitals in a DEC–CC calculation

This section describes unpublished work

Introduction

In the previous chapters, it has been discussed how the fragment spaces in a DEC calculation may be chosen in order to retain error control, how the DEC algorithm may be set up such that the independent fragment calculations can be run on a large supercomputer, and how this development may lead to a reduction of the time–to–solution of a given problem. It has not been discussed that, despite the reduced cost and time–to–solution of the DEC algorithm, the fragmentation strategy naturally leads to redundancies in the obtained parameters, and therefore a huge (constant) overhead is associated with the DEC algorithm. Also, it has not been discussed that even a fragment calculation may become too large to be treated it with the conventional CC methods we have discussed so far. In this section, we will discuss a strategy, which may be used for tackling both of these problems, by reducing the number of parameters to be determined at the fragment level.

In contrast to DEC, some other approaches, like the DLPNO-CCSD(T) [110] approach, obtain the WF parameters by solving one equation for the full molecule, instead of many small fragment equations [97, 110–112]. While this usually makes the code difficult to parallelize over many nodes, the pre–factor of a large–scale calculation becomes rather low, in fact so low that there is virtually no crossover with a canonical CCSD(T) algorithm [110].

On the other hand, the DEC approach has the advantage that it may be combined with many of the strategies developed for obtaining the solution to the conventional problem, by applying the strategies at the fragment level. Here, we will investigate the possibility of using a similar approach as the one presented in Ref. 110 in order to reduce the cost for the individual fragment calculations, by tailoring the fragment calculation such that only the important amplitudes have to be calculated.

8.1 Errors and redundancies in a DEC–CC calculation

In Chapter 6 it has been discussed how the fragments in a DEC calculation may be obtained in a way that the error may be controlled by a single input parameter. By fragmenting the molecule, and screening pair–fragment contributions, it was possible to arrive at a linear–scaling

algorithm, where each fragment calculation is a conventional CC calculation on a fraction of the full molecular orbital space, and therefore all the amplitudes in the AOS (see Section 6.2.1) had to be obtained. Here, we will discuss the individual error sources in a fragment calculation in order to identify the (sub) sets of amplitudes (of the AOS), which are necessary in order to obtain the fragment energy to within the predefined precision. For simplicity, we focus on the occupied partitioning scheme, but for completeness, we note that the following developments may also be used for the virtual partitioning due to the fact that the virtual orbitals are local in space. Also, we focus on a single fragment, but the extension of the following discussion to pair fragments is trivial.

There are three different sources of error associated with the calculation of E_P^o in Eq. (6.6):

- **Error type 1:** The ab summation in Eqs (6.6) is restricted to the $[\overline{P}]$ space, i.e. contributions outside $[\overline{P}]$ are neglected.
- **Error type 2:** The amplitude equation is solved for amplitudes in the AOS ($[\underline{P}] \cup [\overline{P}]$), and therefore the amplitudes cannot couple to amplitudes outside the AOS (error type **2** collectively represents the space extension mechanisms 1-4 of Section 6.2.2).
- **Error type 3:** The underlying MOs are described using approximate MO coefficients in the single fragment extent $\{P\}$ as described in Appendix B.1.

When carrying out a fragment expansion step in the fragment optimization procedure described in Section 6.3, $[\underline{P}]$, $[\overline{P}]$, and $\{P\}$ are all increased (see Figure 6.13), and error types **1-3** are thus all reduced. In practical calculations it is not possible to distinguish these errors, but they may be analyzed for test systems (see Section 8.2). The important point is that, for each single fragment, the expansion procedure proceeds in a black box manner until the error in the single fragment energy (caused by a combination of the three errors) is smaller than the requested FOT precision. Let us reiterate on the nature of the amplitudes in a single fragment calculation, since it is of central importance to this section that the amplitudes of the EOS, are obtained as precisely as possible, as they enter the definition of the single fragment energy in Eq. (6.6). The rest of the amplitudes in the AOS are needed, to describe the coupling (mechanisms 1-4 of Section 6.2.2) of the EOS amplitudes to their environment accurately. One reason for the huge overhead of a DEC calculation, thus stems from the fact that the EOS amplitudes of one single fragment P will enter in the AOS of several other single fragments as coupling amplitudes (and/or the same AOS amplitudes are present in several fragments), where they have to be re–determined. The role of these amplitudes in a single fragment calculation for the fragments P and Q is thus very different, e.g. the EOS amplitudes of fragment P are coupling amplitudes of fragment Q, and it would be desirable to treat these amplitudes differently. Furthermore, an attractive design feature of the fragment space would be to express the fragment calculation in (a) set(s) of orbitals that allow for an as compact as possible representation of the important amplitudes for the given single fragment. How to achieve both is the focus of the next section.

8.2 Fragment reduction using fragment-adapted orbitals

In a first step, let us focus on finding a compact representation for the EOS and postpone the discussion about the coupling amplitudes. Assume that the fragment expansion procedure for single fragment P has been converged for the given FOT to yield a set of occupied AOS orbitals $[\underline{P}]$ and a set of virtual AOS orbitals $[\overline{P}]$, both of which are represented in terms of localized

HF orbitals. In this section we analyze a possibility for reducing the virtual orbital space and thereby illuminate different behaviors of the MP2 and CCSD models in a local correlation context which may ultimately lead to a more efficient DEC code.

The single fragment energy resulting from the fragment expansion procedure $E_P^{o,\mathrm{conv}}$ is invariant with respect to a unitary transformation of the virtual AOS orbitals $[\overline{P}]$. The ultimate goal of this section is to effectively reduce $[\overline{P}]$ without compromising the precision of $E_P^{o,\mathrm{conv}}$.

8.2.1 Virtual fragment-adapted orbitals

The strategy that will be used to reduce the $[\overline{P}]$ orbital space is based on pair natural orbitals (PNOs) and similar constructs, and we therefore briefly summarize the construction of PNOs here. The PNOs for the $(i,j)^{\mathrm{th}}$ occupied orbital pair, where i may equal j, are obtained by diagonalizing the pair-specific virtual density matrix $\boldsymbol{\rho}^{(ij)}$ for pair (i,j) [95,113]

$$\rho_{ab}^{(ij)} = \sum_c (\tilde{t}_{ij}^{ac} t_{ij}^{bc} + \tilde{t}_{ij}^{ca} t_{ij}^{cb}) \tag{8.1}$$

$$\boldsymbol{\rho}^{(ij)} \mathbf{d}^{(ij)} = \boldsymbol{\eta}^{(ij)} \mathbf{d}^{(ij)}, \tag{8.2}$$

where $\mathbf{d}^{(ij)}$ is a unitary matrix, $\boldsymbol{\eta}^{(ij)}$ is a diagonal matrix, and

$$\tilde{t}_{ij}^{ab} = \frac{1}{1+\delta_{ij}} \big(4t_{ij}^{ab} - 2t_{ij}^{ba}\big). \tag{8.3}$$

Usually, the amplitudes used in Eq. (8.1) are obtained from an approximation to CID amplitudes [114]. For both, simplicity and conceptual reasons, we have chosen the amplitudes entering Eq. (8.1) to be the exact MP2 amplitudes, independent of the target CC model. This means, it is more expensive to obtain these amplitudes, which is not of importance in a proof of concept implementation. Note that it is possible to define an analogous $\rho_{ij}^{(ab)}$ if local virtual orbitals are available.

The matrix of eigenvectors $\mathbf{d}^{(ij)}$ defines the transformation from local orbitals $\{\phi\}$ to PNOs $\{\psi^{(ij)}\}$,

$$\psi_{\bar{a}}^{(ij)} = \sum_b \phi_b d_{b\bar{a}}^{(ij)}. \tag{8.4}$$

The PNOs $\{\psi^{(ij)}\}$ are orthogonal among each other, but the PNOs for pair (ij) are in general not orthogonal to the PNOs for pair (kl) (where $k \neq i$ and/or $l \neq j$). Listing the eigenvalues $\{\eta^{(ij)}\}$ in decreasing order defines a list of PNOs ordered according to their importance for the MP2 density, where the largest eigenvalues represent the PNOs with the largest amplitudes t_{ij}^{ab} (and thus the largest correlation energy contributions involving ϕ_i and ϕ_j). More generally, the set of natural orbitals is the set of orbitals which leads to the most compact representation of the wave function according to various measures [115,116]. This compact representation allows for discarding a large set of orbitals (and hence parameters associated with these), based on the eigenvalues in Eq. (8.2), with tunable precision.

In the DEC scheme we group local orbitals by assigning them to atoms as described in Chapter 6. For example, for the determination of E_P^o (E_{PQ}^o) in Eq. (6.6) (Eq. (6.7)) the central orbitals ϕ_i, ϕ_j are the ones where $i,j \in \underline{P}$ ($i \in \underline{P}, j \in \underline{Q}$ and $j \in \underline{P}, i \in \underline{Q}$). For a given single fragment P (pair fragment PQ) we therefore define the DEC EOS equivalents of Eqs. (8.1)-(8.4)

as

$$\rho_{ab}^{(P_v)} = \sum_{ij\in\underline{P}}\sum_{c\in[\overline{P}]} \bar{t}_{ij}^{ac} t_{ij}^{bc} \quad (a,b\in[\overline{P}]) \tag{8.5a}$$

$$\boldsymbol{\rho}^{(P_v)}\mathbf{d}^{(P_v)} = \boldsymbol{\eta}^{(P_v)}\mathbf{d}^{(P_v)} \tag{8.5b}$$

$$\psi_a^{(P_v)} = \sum_{b\in[\overline{P}]} \phi_b d_{ba}^{(P_v)} \quad (a\in[\overline{P}]) \tag{8.5c}$$

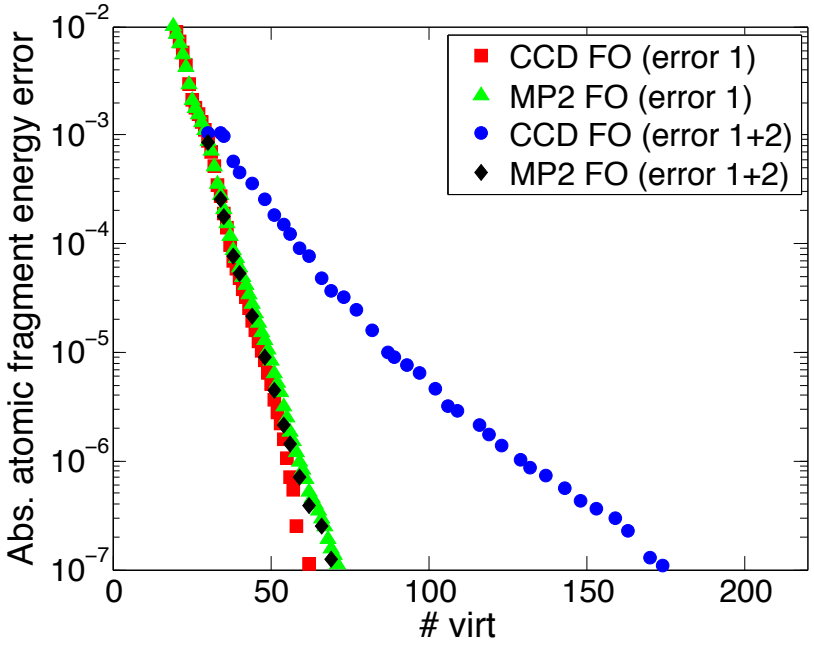
$$\bar{t}_{ij}^{ab} = 4t_{ij}^{ab} - 2t_{ij}^{ba} \tag{8.5d}$$

The (P_v) superscript refers to the virtual orbitals of single fragment P (and pair fragment PQ; to simplify the discussion we, from now on, consider only a single fragment P), and a and b are local orbital indices in the virtual AOS generated by the expansion procedure in Section 6.3. Please note that we may analogously define $\rho_{ij}^{(P_o)} = \sum_{ab\in\overline{P}}\sum_{k\in[\underline{P}]} \bar{t}_{ik}^{ab} t_{jk}^{ab} \quad (i,j\in[\underline{P}])$ for the virtual partitioning (also for the pair fragments). In analogy with the PNOs, the relative importance of the orbitals $\{\psi^{(P_v)}\}$ is dictated by the sizes of the eigenvalues $\{\eta^{(P_v)}\}$. The DEC equivalent of PNOs in Eq. (8.5c) will be denoted virtual *fragment-adapted orbitals (FOs)* because they are specifically adapted to the central occupied orbitals ($ij\in P$) for each fragment. Thus, when using FOs in DEC calculations, the localized HF orbitals merely constitute an efficient basis where the FOs are expanded, rather than representing the actual excitation space, as has been the case in previous DEC calculations. Throughout this section we assume that the FOs are listed in order of decreasing eigenvalues. We note that the pre–factor for $\tilde{t}$ in Eq. (8.3) (included to avoid double counting for $i=j$ in Eq. (8.1)) is not needed in for $\bar{t}$ in Eq. (8.5d), since double counting is not an issue with the density matrix definition in Eq. (8.5a).

In Figure 8.1 we present a test calculation to evaluate the performance of the orbitals defined in Eq. (8.5) (*not* a practical implementation). The test system is decanoic acid, using the cc-pVDZ basis [28], where we look at a single fragment energy of fragment P which we have chosen to be the carbon of the terminal methyl group. For testing purposes, the amplitude equations were first solved in the full molecular orbital space for MP2 in order to obtain the FO basis. Then, in a second step, the MP2 and CCD (as a simplified model for CCSD) equations were solved using the untruncated FO space, and upon transformation to the local orbital space these amplitudes were used in Eq. (6.2) (untruncated). This reference calculation results of course in the correct MP2 and CCD fragment energies. To obtain the green (MP2) and red (CCD) curves in Figure 8.1 (only error type **1**) the FO space was truncated *after* obtaining the amplitudes in the untruncated FO space and *before* the transformation of the amplitudes to the local basis and compared to the correct MP2 and CCD fragment energies. The number of virtual orbitals obtained for a given truncation parameter is given on the x–axis, and the untruncated virtual space includes 220 virtual orbitals. The procedure is the same for obtaining the black (MP2) and blue (CCD) curves (error types **1** and **2**), only that the FO space was truncated *prior to* obtaining the amplitudes.

Figure 8.1 shows that obtaining the (correct) amplitudes in the untruncated FO space and then selecting the amplitudes based on the FO threshold leads to a similar convergence of the single fragment energy error for MP2 and CCD. It is apparent that these amplitudes may be expressed in a similar compact fashion using FOs, as only about 30% of the virtual orbital space is necessary to decrease the fragment error to 10^{-7} a.u. However, when solving the amplitude equations in the truncated space, only the MP2 model performs well. The reason for the slow convergence of the single fragment energy error when solving the CCD equation in the truncated FO space is that many contributions to the CCD residual equations (see Table 1.2)

Figure 8.1: DEC error analysis for MP2 and CCD, The error in the single fragment energy is plotted against the number of virtual FOs selected by various truncation parameters, where the error types discussed in Section 8.1 are present as indicated by the figure legend. In all calculations, all occupied orbitals and AOs were included. The central atom for the single fragment energy under consideration is the terminal $-CH_3$ carbon atom of decanoic acid (cc-pVDZ basis [28]). We note that the horizontal axis goes up to 220 which is the total number of virtual orbitals in the test molecule.



are discarded, i.e., orbitals, which are connected to sizable amplitudes in the surrounding of the EOS amplitudes, but only to negligible EOS amplitudes. This occurs because the FOs are designed only to describe the EOS amplitudes as compactly as possible and not the sizable amplitudes to which the EOS amplitudes couple strongly through the CCD residual equations. Neglecting these coupling amplitudes results in an increased error in the fragment energy for the CCD model.

This demonstrates that a straight forward application of the FOs for CCD and CCSD is not possible as the EOS amplitudes couple to other amplitudes of the AOS to a greater extent than in MP2. In the following subsection we will discuss, how to reduce the overall number of parameters to be determined in a fragment calculation while taking the coupling effects into account. Specifically, we will investigate how we can combine the FO and PNO approaches in order to be able to reduce the number of parameters and still incorporate the important coupling amplitudes.

8.2.2 The combined FO/PNO approach

In this section we will focus on the combining the FO and PNO approaches and explicitly derive the T_1 transformed CCSD equations for FOs/PNOs. The PNO's are constructed using the definition of the pair orbital density in Eq. (8.1) and solving for the eigenvalues in Eq. (8.2). The transformation of a set of local orbitals to a set of pair natural orbitals is then given by Eq. (8.4) and the matrix elements of the transformation matrix may be written as $d_{a\bar{b}}^{(ij)} = \langle \phi_a | \psi_{\bar{b}}^{(ij)} \rangle$. To simplify the notation we drop the symbol ϕ (ψ) for the local (FO/PNO) basis and rather use the orbital index, where a bar is added when referring to an orbital from the FO/PNO basis,

and the space the basis is adapted to, as superscript. An index without bar refers to an orbital from the local basis. Thus, the set of local orbitals and FOs/PNOs adapted to $P_v/i,j$ are written as $\{a\}$ and $\{\bar{a}^{(P_v)}\}/\{\bar{a}^{(ij)}\}$. The PNOs $\{\bar{a}^{(ij)}\}$ form an orthogonal set (implication of diagonalizing $\boldsymbol{\rho}^{(ij)}$) but they are in general not orthogonal to another set of PNOs $\{\bar{a}^{(kl)}\}$ ($i \neq k$ and/or $j \neq l$). However, all the virtual spaces discussed here (local orbitals, FOs, PNOs) are orthogonal to the occupied space, which is used as a common space. Using these different sets of orbitals, it is possible to extract the largest EOS amplitudes by means of the FO space, as described in Section 8.2.1, which directly targets the amplitudes entering the single fragment energy expression in Eq. (6.6). The largest coupling amplitudes may be extracted using the PNOs, and thus, when solving the amplitude equations their effect may be taken into account.

Using the implications of the diagonalization of $\boldsymbol{\rho}^{(ij)}$ it is fairly straightforward to derive the T_1 transformed PNO–CCSD equations, using Table 1.2 as a starting point. Since the matrices $\mathbf{d}^{(ij)}$ are obtained by solving the eigenvalue equation Eq. (8.2) they are unitary

$$(\mathbf{d}^{(ij)})^\dagger \mathbf{d}^{(ij)} = \mathbf{1}. \tag{8.6}$$

We may then write the orbital transformation relations as

$$|\bar{a}^{(ij)}\rangle = \sum_a |a\rangle (\mathbf{d}^{(ij)})_{a\bar{a}^{(ij)}} = \sum_a |a\rangle d^{(ij)}_{a\bar{a}^{(ij)}}, \tag{8.7}$$

$$\langle\bar{a}^{(ij)}| = \sum_a (\mathbf{d}^{(ij)^\dagger})_{\bar{a}^{(ij)}a} \langle a| = \sum_a (d^{(ij)}_{a\bar{a}^{(ij)}})^* \langle a|, \tag{8.8}$$

$$|a\rangle = \sum_{\bar{a}^{(ij)}} |\bar{a}^{(ij)}\rangle (\mathbf{d}^{(ij)^\dagger})_{\bar{a}^{(ij)}a} = \sum_{\bar{a}^{(ij)}} |\bar{a}^{(ij)}\rangle (d^{(ij)}_{a\bar{a}^{(ij)}})^*, \tag{8.9}$$

$$\langle a| = \sum_{\bar{a}^{(ij)}} (\mathbf{d}^{(ij)}_{a\bar{a}^{(ij)}})^* \langle\bar{a}^{(ij)}| = \sum_{\bar{a}^{(ij)}} d^{(ij)}_{a\bar{a}^{(ij)}} \langle\bar{a}^{(ij)}|, \tag{8.10}$$

where the complex conjugation may be dropped when working with real orbitals. The contravariant virtual indices of the amplitudes are transformed as the bras in the upper equations, as

$$t_{ij}^{\bar{a}^{(ij)}b} = \sum_a (d^{(ij)}_{a\bar{a}^{(ij)}})^* t_{ij}^{ab}, \tag{8.11}$$

$$t_{ij}^{ab} = \sum_{\bar{a}^{(ij)}} d^{(ij)}_{a\bar{a}^{(ij)}} t_{ij}^{\bar{a}^{(ij)}b}. \tag{8.12}$$

Here we have used Eq. (8.6). This relation can then be inserted in each of the basis sets defined by i and j whenever we evaluate the overlap matrices.

The overlap between sets of PNOs There are different kinds of overlap matrices which become necessary when we split a fragment into spaces of different importance. We decided to treat the EOS in a special way, since the EOS defines the target amplitudes with $i,j \in \underline{P}$ and virtual indices $a,b \in \overline{P}$. In order to reduce the amount of amplitudes within the EOS we have defined the FO basis $\{a^{(P_v)}\}$ in Eq. (8.5) where a compact representation of the EOS amplitudes is obtained. For the coupling spaces we have decided to use the PNO basis $\{a^{(ij)}\}$, thus, overlap matrices between PNO spaces for (ij) and (kl) are needed, and also the overlap of a PNO space with the FO space. When we derive the CCSD equations we will omit this distinction and only

use a PNO overlap matrix, as replacing a PNO space with the FO space, in the equations, is a trivial matter.

The EOS is expressed in a set of FOs and thus the overlap matrix element between the FOs and PNOs may be written as

$$S^{(P_v),(ij)}_{\bar{a}^{(P_v)}\bar{b}^{(ij)}} = \langle \bar{a}^{(P_v)} | \bar{b}^{(ij)} \rangle = \sum_{ab} (d^{(P_v)}_{a\bar{a}^{(P_v)}})^* \langle a|b\rangle d^{(ij)}_{b\bar{b}^{(ij)}} = \sum_{a} (d^{(P_v)}_{a\bar{a}^{(P_v)}})^* d^{(ij)}_{a\bar{b}^{(ij)}}, \tag{8.13}$$

where we have used the orthonormality of the local basis. This equation degenerates to the matrix element $(d^{(ij)}_{a\bar{b}^{(ij)}})^*$ if the local basis is used for the EOS, which may be a suitable choice, as we will show later. This equation can be written as $\mathbf{S}^{(P_v),(ij)} = (\mathbf{d}^{(P_v)})^\dagger \mathbf{d}^{(ij)}$. The exact same can be derived for the overlap between two PNO spaces, giving $\mathbf{S}^{(ij),(kl)} = (\mathbf{d}^{(ij)})^\dagger \mathbf{d}^{(kl)}$, analogous to the previous expression. As these expressions essentially are the same we will from now on drop the explicit treatment of the EOS space. Implicitly all the following derivations hold for any of the spaces defined previously by using the correct overlap matrix. We can now express any orbital $\bar{a}^{(ij)}$ in any basis (kl) by simply inserting the definition of the overlap matrices after using Eqs. (8.9) and (8.7) (Eqs. (8.10) and (8.8)) consecutively

$$\bar{a}^{(ij)} = \sum_{\bar{b}^{(kl)}} \bar{b}^{(kl)} S^{(kl),(ij)}_{\bar{b}^{(kl)}\bar{a}^{(ij)}}, \tag{8.14}$$

Expressing amplitudes in the same way, using relations (8.11) and (8.12) consecutively and summing over the local index a, leads to

$$t_{ij}^{\bar{a}^{(ij)}b} = \sum_{\bar{a}^{(kl)}} S^{(ij),(kl)}_{\bar{a}^{(ij)}\bar{a}^{(kl)}} t_{ij}^{\bar{a}^{(kl)}b}. \tag{8.15}$$

Now we have collected all the relations are which may be inserted into the equations in Table 1.2 to arrive at the T_1 transformed PNO–CCSD expressions. We will begin by deriving the contribution for one of the terms in the next subsection and then give the T_1 transformed PNO–CCSD equations in Section 8.2.4.

8.2.3 Example derivation from a MP2 residual contribution

Here we show how the contribution $\Omega^{E2.2}_{aibj}$ to the CCSD vector function may be written, using the PNO representation for all amplitudes. The conventional T_1 transformed residual contribution $\Omega^{E2.2}_{aibj}$ may be written as (see Table 1.2)

$$\Omega^{E2.2}_{aibj} = \sum_k t_{ik}^{ab}\ {}^I\tilde{F}_{kj}. \tag{8.16}$$

Since the residual Ω_{aibj} enters the amplitudes t_{ij}^{ab} of the next iteration directly (see Eq. (2.30)), it is expressed it in terms of the same PNOs as the corresponding amplitudes, i.e. in the $\{a^{(ij)}\}$ basis, as

$$\Omega_{\bar{a}^{(ij)}i\bar{b}^{(ij)}j} = \sum_{ab} (d^{(ij)}_{a\bar{a}^{(ij)}})^* (d^{(ij)}_{b\bar{b}^{(ij)}})^* \Omega_{aibj}, \tag{8.17}$$

to obtain it in the same basis as the corresponding amplitudes it enters; also, the residual is transformed as the amplitudes in Eq. (8.11). In order to arrive at a compact representation for

all the doubles amplitudes entering this expression, the amplitudes t_{ik}^{ab} need to be expressed in the $\{a^{(ik)}\}$ basis, thus $t_{ik}^{\bar{a}^{(ik)}\bar{b}^{(ik)}}$ is the most compact representation (using Eq. (8.12))

$$t_{ik}^{ab} = \sum_{\bar{a}^{(ik)}\bar{b}^{(ik)}} d_{a\bar{a}^{(ik)}}^{(ik)} d_{b\bar{b}^{(ik)}}^{(ik)} t_{ik}^{\bar{a}^{(ik)}\bar{b}^{(ik)}}. \tag{8.18}$$

Using Eq. (8.18) in Eq. (8.16), and transforming $\Omega_{aibj}^{E2.2}$ to $\{a^{(ij)}\}$, we obtain

$$\begin{aligned}
\Omega_{\bar{a}^{(ij)}i\bar{b}^{(ij)}j}^{E2.2} &= \sum_{ab} (d_{a\bar{a}^{(ij)}}^{(ij)})^* (d_{b\bar{b}^{(ij)}}^{(ij)})^* \Omega_{aibj}^{E2.2}, \\
&= \sum_{ab} (d_{a\bar{a}^{(ij)}}^{(ij)})^* (d_{b\bar{b}^{(ij)}}^{(ij)})^* \sum_k \sum_{\bar{c}^{(ik)}\bar{d}^{(ik)}} d_{a\bar{c}^{(ik)}}^{(ik)} d_{b\bar{d}^{(ik)}}^{(ik)} t_{ik}^{\bar{c}^{(ik)}\bar{d}^{(ik)}} \ {}^I\tilde{F}_{kj}, \\
&= \sum_k \sum_{\bar{c}^{(ik)}d^{(\bar{i}k)}} S_{\bar{a}^{(ij)}\bar{c}^{(ik)}}^{(ij),(ik)} S_{\bar{b}^{(ij)}\bar{d}^{(ik)}}^{(ij),(ik)} t_{ik}^{\bar{c}^{(ik)}\bar{d}^{(ik)}} \ {}^I\tilde{F}_{kj},
\end{aligned} \tag{8.19}$$

which could have been obtained by using Eq. (8.15) directly.

8.2.4 The T_1 transformed PNO–CCSD vector equations

Following the strategy outlined in the previous subsection, we may express all the terms in Table 1.2 in terms of PNOs and thus in the most compact representation for all the doubles amplitudes. Since it is rather straightforward to introduce Eqs. (8.7)- (8.10) we will give the final expressions.

One minor complication for all terms in Table 1.2, is the occurrence of the T_1 integrals,

$$\begin{aligned}
\tilde{g}_{\bar{a}^{(ij)}i\bar{b}^{(ij)}j} &= \sum_{ab} (d_{a\bar{a}^{(ij)}}^{(ij)})^* (d_{b\bar{b}^{(ij)}}^{(ij)})^* \tilde{g}_{aibj}, \\
&= \sum_{\mathfrak{pqrs}} \sum_{ab} (d_{a\bar{a}^{(ij)}}^{(ij)})^* (d_{b\bar{b}^{(ij)}}^{(ij)})^* \Lambda_{\mathfrak{p}a}^p \Lambda_{\mathfrak{q}i}^h \Lambda_{\mathfrak{q}b}^p \Lambda_{\mathfrak{s}j}^h g_{\mathfrak{pqrs}}, \\
&= \sum_{\mathfrak{pqrs}} \mathcal{A}_{\mathfrak{p}\bar{a}^{(ij)}}^p \Lambda_{\mathfrak{q}i}^h \mathcal{A}_{\mathfrak{r}\bar{b}^{(ij)}}^p \Lambda_{\mathfrak{s}j}^h g_{\mathfrak{pqrs}},
\end{aligned} \tag{8.20}$$

where we have introduced

$$\sum_a \Lambda_{\mathfrak{p}a}^p (d_{a\bar{a}^{(ij)}}^{(ij)})^* = \mathcal{A}_{\mathfrak{p}\bar{a}^{(ij)}}^p. \tag{8.21}$$

Since the T_1 transformation (one way or another) involves the transformation from the full basis to the PNO space in each iteration this implementation will become rather inefficient. We have recently learned that the transformations to the PNO basis usually is the most expensive part in a PNO code and therefore all necessary integrals are usually precomputed and stored. Also, the resolution of the identity approximation is frequently used to increase the performance. We will thus not expect the current code to scale well, but we may be able to reduce the number of parameters to be determined for each fragment in a tailored way, for the energy (EOS) amplitudes and the coupling amplitudes. Using the strategy from Eq. (8.19) and from Eq. (8.20) for the all the terms in Table 1.2, we arrive at the expressions given in Figs. 8.2and 8.3.

We have chosen to calculate the singles residual contribution in the basis of the EOS, even though there is no reason for them to be described in the same compact fashion as the doubles amplitudes. In fact, we could (and perhaps should) have chosen to use the local basis instead for the single to rigorously analyze the fragment energy when using the combined FO/PNO approach.

Figure 8.2: The explicit T_1 transformed PNO–CCSD doubles residual expressions.

$$\Omega^{A2}_{\bar{a}^{(ij)}i\bar{b}^{(ij)}j} = \tilde{g}_{\bar{a}^{(ij)}i\bar{b}^{(ij)}j} + \sum_{\bar{c}^{(ij)}\bar{d}^{(ij)}} t^{\bar{c}^{(ij)}\bar{d}^{(ij)}}_{ij}\, \tilde{g}_{\bar{a}^{(ij)}\bar{c}^{(ij)}\bar{b}^{(ij)}\bar{d}^{(ij)}} \tag{8.22}$$

$$\Omega^{B2}_{\bar{a}i\bar{b}j} = \sum_{kl} \sum_{\bar{c}^{(ij)}\bar{d}^{(ij)}} S^{(ij),(kl)}_{\bar{a}^{(ij)}\bar{c}^{(kl)}} S^{(ij),(kl)}_{\bar{b}^{(ij)}\bar{d}^{(kl)}} t^{\bar{c}^{(kl)}\bar{d}^{(kl)}}_{kl}\, B_{kilj} \tag{8.23}$$

$$B_{kilj} = \tilde{g}_{kilj} + \sum_{\bar{c}^{(ij)}\bar{d}^{(ij)}} t^{\bar{c}^{(ij)}\bar{d}^{(ij)}}_{ij}\, \tilde{g}_{k\bar{c}^{(ij)}l\bar{d}^{(ij)}} \tag{8.24}$$

$$\Omega^{C2}_{\bar{a}^{(ij)}i\bar{b}^{(ij)}j} = -\left(1 + \frac{1}{2}\hat{P}_{ij}\right) \sum_{k} \sum_{\bar{b}^{(ki)}\bar{c}^{(ij)}} S^{(ij),(ki)}_{\bar{b}^{(ij)}\bar{b}^{(ki)}} t^{\bar{b}^{(ki)}\bar{c}^{(ki)}}_{ki}\, C_{kj\bar{a}^{(ij)}\bar{c}^{(ki)}} \tag{8.25}$$

$$C_{kj\bar{a}^{(ij)}\bar{c}^{(ki)}} = \tilde{g}_{kj\bar{a}^{(ij)}\bar{c}^{(ki)}} - \frac{1}{2} \sum_{\bar{a}^{(lj)}} S^{(ij),(lj)}_{\bar{a}^{(ij)}\bar{a}^{(lj)}} \sum_{\bar{d}^{(lj)}l} t^{\bar{a}^{(lj)}\bar{d}^{(lj)}}_{lj}\, \tilde{g}_{k\bar{d}^{(lj)}l\bar{c}^{(ki)}} \tag{8.26}$$

$$\Omega^{D2}_{\bar{a}^{(ij)}i\bar{b}^{(ij)}j} = \frac{1}{2} \sum_{k} \sum_{\bar{c}^{(jk)}\bar{b}^{(jk)}} S^{(ij),(jk)}_{\bar{b}^{(ij)}\bar{b}^{(jk)}} u^{\bar{b}^{(jk)}\bar{c}^{(jk)}}_{jk}\, D_{\bar{a}^{(ij)}ik\bar{c}^{(jk)}} \tag{8.27}$$

$$D_{\bar{a}^{(ij)}ik\bar{c}^{(jk)}} = \tilde{L}_{\bar{a}^{(ij)}ik\bar{c}^{(jk)}} + \frac{1}{2} \sum_{l} \sum_{\bar{a}^{(il)}\bar{d}^{(il)}} S^{(ij),(il)}_{\bar{a}^{(ij)}\bar{a}^{(il)}} u^{\bar{a}^{(il)}\bar{d}^{(il)}}_{il}\, \tilde{L}_{l\bar{d}^{(il)}k\bar{c}^{(jk)}} \tag{8.28}$$

$$\Omega^{E2}_{\bar{a}^{(ij)}i\bar{b}^{(ij)}j} = \sum_{\bar{c}^{(ij)}} t^{\bar{a}^{(ij)}\bar{c}^{(ij)}}_{ij}\, E^{1}_{\bar{b}^{(ij)}\bar{c}^{(ij)}} - \sum_{k} \sum_{\bar{a}^{(ik)}\bar{b}^{(ik)}} S^{(ij),(ik)}_{\bar{a}^{(ij)}\bar{a}^{(ik)}} S^{(ij),(ik)}_{\bar{b}^{(ij)}\bar{b}^{(ik)}} t^{\bar{a}^{(ik)}\bar{b}^{(ik)}}_{ik}\, E^{2}_{kj} \tag{8.29}$$

$$E^{1}_{\bar{b}^{(ij)}\bar{c}^{(ij)}} = {}^{I}\tilde{F}_{\bar{b}^{(ij)}\bar{c}^{(ij)}} - \sum_{kl} \sum_{\bar{b}^{(kl)}\bar{d}^{(kl)}} S^{(ij),(kl)}_{\bar{b}^{(ij)}\bar{b}^{(kl)}} u^{\bar{b}^{(kl)}\bar{d}^{(kl)}}_{kl}\, \tilde{g}_{l\bar{d}^{(kl)}k\bar{c}^{(ij)}} \tag{8.30}$$

$$E^{2}_{kj} = {}^{I}\tilde{F}_{kj} - \sum_{l} \sum_{\bar{c}^{(lj)}\bar{d}^{(lj)}} u^{\bar{c}^{(lj)}\bar{d}^{(lj)}}_{lj}\, \tilde{g}_{k\bar{d}^{(lj)}l\bar{c}^{(lj)}} \tag{8.31}$$

Figure 8.3: The explicit T_1 transformed PNO–CCSD doubles singles expressions.

$$\Omega^{A1}_{\bar{a}^{(P_v)}i} = \sum_{k} \sum_{\bar{c}^{(ki)}\bar{d}^{(ki)}} u^{\bar{c}^{(ki)}\bar{d}^{(ki)}}_{ki}\, \tilde{g}_{\bar{a}^{(P_v)}\bar{d}^{(ki)}k\bar{c}^{(ki)}} \tag{8.32}$$

$$\Omega^{B1}_{\bar{a}^{(P_v)}i} = -\sum_{kl} \sum_{\bar{a}^{(kl)}\bar{c}^{(kl)}} S^{(P_v),(kl)}_{\bar{a}^{(P_v)}\bar{a}^{(kl)}} u^{\bar{a}^{(kl)}\bar{c}^{(kl)}}_{kl}\, \tilde{g}_{kil\bar{c}^{(kl)}} \tag{8.33}$$

$$\Omega^{C1}_{\bar{a}^{(P_v)}i} = \sum_{k} \sum_{\bar{a}^{(ik)}\bar{c}^{(ik)}} S^{(P_v),(ik)}_{\bar{a}^{(P_v)}\bar{a}^{(ik)}} u^{\bar{a}^{(ik)}\bar{c}^{(ik)}}_{ik}\, {}^{I}\tilde{F}_{k\bar{c}^{(ik)}} \tag{8.34}$$

$$\Omega^{D1}_{\bar{a}^{(P_v)}i} = {}^{I}\tilde{F}_{\bar{a}^{(P_v)}i} \tag{8.35}$$

Table 8.1: color and symbol coding for the thresholds used for truncating the PNO spaces T_{ij} (symbols), and FO spaces T_{P_v} (colors).

color	blue	dark green	red	cyan	magenta	yellow green	black	green	light grey	brown
T_{P_v}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}

symbol	○	□	▽	△	◁	▷	◊
T_{ij}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}

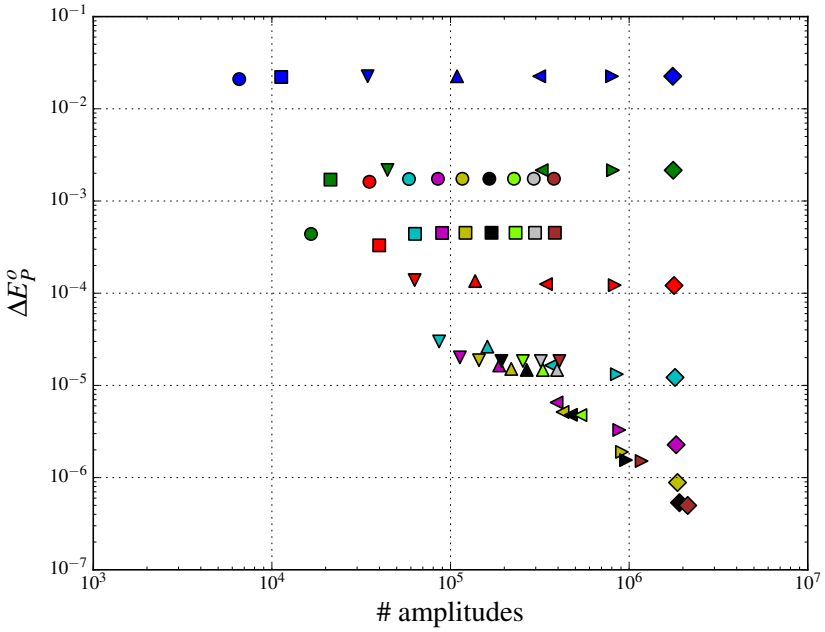
8.3 Results

All T_1 transformed PNO–CCSD equations have been programmed in an integral–direct algorithm following the algorithm outlined in Chapter 4 in many cases. Since the memory–conservative integral–direct approach has been chosen (in order to align the algorithm with the programming paradigms in **LSDALTON**), the transformations to and between the PNO spaces render the algorithm unnecessarily expensive. Yet, a useful first glance at the compactness of the description of the WF in the PNO space could be obtained.

The main reason for introducing the complicated basis set transformations in Figs. 8.2and 8.3 is that the FO and PNO bases may be truncated, by selecting eigenvalues of a certain size in Eqs. (8.2) and (8.5b) and still capture the main features of the amplitudes. With this procedure, the number of parameters to be determined compared to a standard calculation may be reduced drastically, as will be demonstrated in this section. Since we have chosen to partition the amplitudes into the target EOS amplitudes that directly enter the fragment energy expressions in Eqs. (6.6) and (6.7), and amplitudes which couple to the EOS amplitudes, there will be two parameters to be considered, T_{P_v} for selecting eigenvalues from Eq. (8.5b), and T_{ij} for selecting eigenvalues from Eq. (8.2). All the calculations have been carried out on a set of nodes with 96GB main memory and 12 cores per node on the **Grendel** cluster using a development version of the **LSDALTON** program in order to obtain comparable timings. The system under investigation was decanoic acid, where the terminal methyl carbon was taken as atomic center P (the same as for Figure 8.1) and where errors of type **3** are not present. The number of practically determined parameters in the full calculation ($O = 48$,$V = 220$) is $OV + O^2V^2 = 111\,524\,160 \approx 1.1 \cdot 10^9$, while the number of unique parameters ($t_{ij}^{ab} = t_{ji}^{ba}$) for the full calculation is $OV + \frac{1}{2}O(O+1)V^2 = 56\,928\,960 \approx 5.7 \cdot 10^8$.

In all the following plots we use colors to distinguish different values for T_{P_v} and different symbols to distinguish the different values for T_{ij}, as given in Table 8.1. We will first investigate the compactness of the description upon variation of the two thresholds, by plotting the numerical value of the difference in the occupied fragment energy $\Delta E_P^o = |E_P^o - E_P^{o,f}|$ as a function of the number of parameters in the calculation. In Figure 8.4 we have given the numerical value of deviation ΔE_P^o of the fragment energy calculated with all combinations of T_{P_v} and T_{ij} given in Table 8.1. It is remarkable that it is possible to close in on the correct CCSD fragment energy $E_P^{o,f}$ to within an error of 10^{-6} a.u., only using (3.3%) 1.7% of the (non–redundant)

Figure 8.4: The dependence of the fragment energy E_P^o on the two cutoff thresholds T_{ij}, and T_{P_v} for the FO and PNO spaces (and thus, on the number of parameters to be determined), where the color and shape coding for the thresholds has been given in Table 8.1.



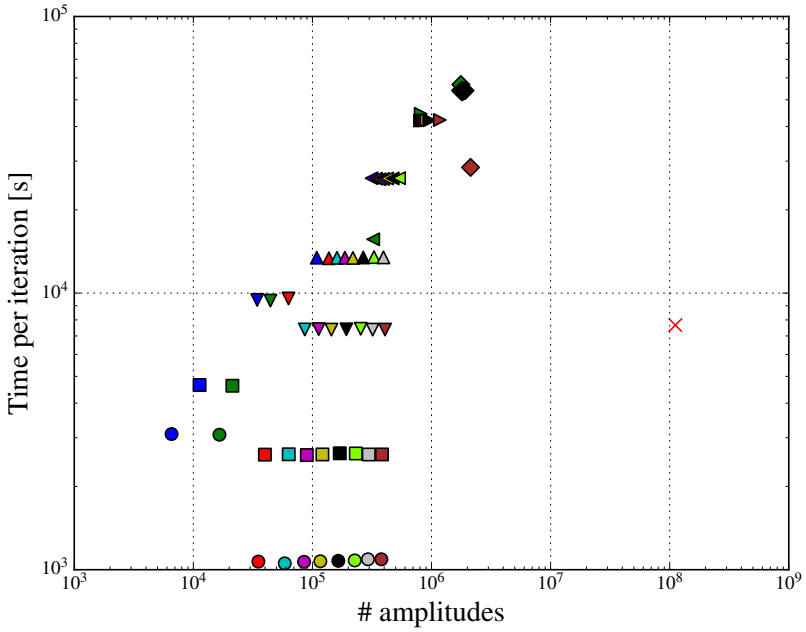
amplitudes. It is further apparent that both decreasing T_{P_v} and T_{ij}, increases the number of amplitudes to be determined.

For a constant T_{P_v} (a given color in Figure 8.4), i.e. the same error of type **1** throughout the calculation series, decreasing T_{ij} has little effect on the precision of the overall calculation down to about 10^{-4} a.u. In these series, decreasing T_{ij} may be interpreted as gradually improving the description of coupling effects, i.e. gradually decreasing the error of type **2**. Thus, only if the errors of type **1** are small enough, the coupling effects become important. Also, for smaller T_{P_v} the effect of decreasing T_{ij} is more pronounced.

Moreover, when the coupling error T_{ij} is kept constant (a given symbol in Figure 8.4), decreasing T_{P_v} improves the description of the fragment energy up to a certain region, where the errors are mainly due to an insufficient description of the coupling space, and the curve flattens out.

It is furthermore interesting to compare the computational timings of the algorithms, despite the fact that we have already stated that the transformations will render the PNO calculation rather expensive using the current pilot implementation. In Figure 8.5 we show how the time for one (average) CCSD iteration is dependent on the two choices of thresholds T_{P_v} and T_{ij}, where the color coding is again given in Table 8.1. The time for a standard CCSD iteration is marked with a red x. Despite being inefficient, one could benefit from using the PNO code in its current state for low precision DEC calculations, e.g. for FOTs of about 10^{-3} a.u. For higher precisions, the cost of using this particular integral–direct PNO–CCSD implementation becomes prohibitively high.

Figure 8.5: The dependence of the time for one iteration (in seconds) on the two cutoff thresholds T_{ij}, and T_{P_v} for the FO and PNO spaces (and thus, on the number of parameters to be determined), where the color and shape coding for the thresholds has been given in Table 8.1. The red x marks the standard CCSD reference calculation.



8.4 Conclusion and outlook

In this chapter we have identified three types of error associated with a DEC fragment calculation, and based on the knowledge about their nature, we have investigated the possibility of using an overall reduced virtual space for a given fragment. It was possible to find a space in which the EOS amplitudes could be expressed very compactly (the truncated FO space), once they had been determined. For MP2 the amplitude equation could also be solved in the truncated FO space to obtain accurate EOS amplitudes. For CCD and CCSD this was not possible, since the FO approach only avoids errors of type **1** and neglects the amplitudes the EOS amplitudes couple with, which is of critical importance when solving CCD and CCSD equations. It was found that the PNOs should be able to describe these coupling amplitudes approximately with a tunable precision, and the explicit form of the T_1 transformed PNO–CCSD equations were given and the combined FO/PNO approach for a fragment calculation was introduced.

In other PNO based CC approaches, e.g. DLPNO-CCSD [95], one set of equations is solved for the full molecule, and all the amplitudes determined, therefore enter the final correlation energy directly. For this reason, it is sufficient to use one threshold when choosing the PNO spaces in these approaches. In the presented fragment approach, we tried to divide the space into a set of target (EOS) amplitudes and a set of auxiliary (coupling) amplitudes where the former set was expressed in a set of FOs and the latter in PNOs. Therefore it was necessary to use two thresholds in the combined FO/PNO approach.

Since the combined FO/PNO approach has two thresholds T_{P_v} and T_{ij} the investigations in the numerical section were focused on exploring the precision of the fragment energy with respect to both thresholds and a crucial balance between T_{P_v} and T_{ij} became apparent. The balance between these two parameters was explained in terms of error **1** and error **2**, but no

practical solution for the determination of these thresholds has been given yet.

Also, it was investigated to which extent this pilot implementation was able to compete with a standard CCSD implementation in terms of speed.

Despite the rather bad performance of the FO/PNO–CCSD code with respect to iteration time, it was uplifting, to see the number of parameters that could be discarded using PNOs. It is now a matter of the implementation details to reduce the time–to–solution for the FO/PNO–CCSD algorithm. In a DEC context, it is not desirable to store individual fragment information on disk, and therefore the integral–direct approach was chosen. This strategy, on the other hand, turned out to be the main reason for the bad performance of the current PNO–CCSD implementation. In order to achieve both reducing the number of transformations from the AO to the PNO basis and avoiding using a local disk, we may need to store the necessary quantities in a parallel distributed fashion (see Section 3) and/or try to reduce the memory requirements by, e.g. using the resolution of the identity approximation, in a future implementation of the FO/PNO–CCSD algorithm.

Chapter 9

Summary and outlook

9.1 English summary

The work presented in this thesis has been focused on extending the application range of the hierarchy of coupled cluster (CC) models. In order to do so, two strategies have been pursued i) extending the application range of the classical CC singles and doubles (CCSD) model by using the parallelism of modern supercomputers, and ii) using a reformulation of the conventional theory to exploit locality and arrive at a linear–scaling algorithm.

i) For the extension of the application range of the conventional CCSD formalism to supercomputers it is of critical importance to reduce the memory requirements of the overall algorithm. Since hard drive storage is not an option on these supercomputers, it is imperative to use a solver for non–linear sets of equations which has a minimal memory footprint. Therefore, the conjugate residual with optimal trial vectors (CROP), which only needs information from three iterations to retain the convergence rate, has been implemented in a parallel–distributed way, where none of the vectors are stored in local memory. However, the most expensive step in the solver algorithm is the evaluation of the vector function, both memory– and time–wise. Thus, the evaluation of the CCSD vector function has been implemented in a flexible, scalable and memory efficient way resulting in a short time–to–solution for a broad variety of systems. For both of these algorithms the development and implementation of a general parallel distributed tensor framework has been necessary. Due to the use of very modern programming paradigms and the associated technical problems, the work on the parallel CCSD code has been delayed and it has only recently been possible to run the calculations presented in this thesis. Therefore, the bottlenecks for large node counts have only recently been identified and thus one future goal is to continue optimizing the algorithm in order to further increase the application range.

ii) The recently developed Divide–Expand–Consolidate (DEC) CC method is based on the availability of local Hartree–Fock orbitals and a black–box optimization of the fragments occurring in such a calculation. In order to retain a rigorous error control, especially when the formalism is extended to gradients and properties, it is of critical importance that the errors introduced at the fragment level are of the predefined size. The foundation for this fragment optimization algorithm has been reviewed and improved, yet a few unresolved problems with the virtual partitioning remain. These will be addressed in the near future. In order to reduce the overall cost of a DEC calculation, an energy–based pair–fragment

screening method has been developed. It has been demonstrated how the overall error control in DEC may be achieved for the CCSD with perturbative triples (CCSD(T)). Furthermore, the natural parallelism in a DEC calculation has been exploited in order to run calculations on up to almost 10 000 computational nodes at a near perfect load balance. However, a bottleneck in the currently used master–slave approach has been identified and will be addressed in the near future.

A fragment CC calculation in the DEC formalism may grow very large, this is why the algorithm of part i) has been developed. However, it would be more elegant if the cost of a fragment calculation could be reduced by other means, e.g. by using the successful pair–natural–orbital (PNO) approach in a fragment calculation. This strategy has been explored and is going to be further pursued. After fixing the fragment optimization, it may be interesting to investigate, how non–local properties, e.g. charge–transfer excitations could be described using a fragment–based approach.

9.2 Dansk resumé

Mit bidrag til kvatekemien er dokumenteret i denne afhandling. Som PhD studerende har jeg fokuseret på udviklingen af metoder til at udvide coupled cluster (CC) teoriens brugbarhed til store systemer, med fokus på CC med singles og doubles eksitationer (CCSD). For at opnå dette er to forskellige strategier blevet forfulgt, i) at udvide de klassiske modeller brugbarhed med fokus på store computeres parallelisme, og ii) at gøre brug af en alternativ formulering som udnytter den fysiske lokalitet af elektronernes korrelation og derved gør det muligt at opnå en linrt skalerende algoritme.

i) For at udvide anvendeligheden af CCSD på store supercomputer, er det vigtigt at mindske de hukommelsesmssige krav forbundet med sådan en beregning. Derfor er det essentielt at have en ligningsløser med et minimalt hukommelsesmæssigt fodaftryk. Derfor er der blevet implementeret en version af conjugate residual med optimale trial vektorer (CROP), hvor alt data er gemt i parallelt distribueret hukommelse. Derudover er der blevet implementeret en massiv parallel og fleksibel version af CCSD vektor ligningerne. For både CROP ligningsløseren og CCSD ligningerne har det været nødvendigt at udvikle og implementere en tensor struktur, til håndtering af generelle tensorer fordelt over mange noder. Da mange nye implementeringsstrategier er blevet anvendt, er der i forløbet opstået flere tekniske problemer og det har først været muligt at køre “at scale” kort tid forinden afleveringen af denne afhandling. Dog har det allerede været muligt at identificere nogle tekniske flaskehalse som helst skal løses indenfor en kort tidshorisont.

ii) Nøjagtigheden af Divide–Expand–Consolidate (DEC) metoden baseret på lokale Hartree-Fock orbitaler, der er unik for denne gruppe, er blevet undersøgt med hensyn til variation af fragmenternes størrelse. Især med henblik på molekylære egenskaber og gradienter, er det afgørende at have rigorøs fejlkontrol, og i denne sammenhng er fundamentet for fragmentoptimeringsalgoritmen blevet forbedret. Dog er der til stadighed problemer ved benyttelse af virtuel partionering. Derudover er det blevet vist at det er muligt at udføre linrt skalerende beregniger på CCSD med perturbative triples (CCSD(T)) niveau og at koden skalerer til ti–tusindvis af noder. En DEC–fragment beregning kan blive stor og dyr, hvilket har været drivkraften bag udviklingen i punkt i). En mere elegant løsning ville imidlertid være at reducere omkostingerne af en fragment beregning ved brug af

f.eks. pair–natrual–orbitals (PNO). Denne strategi er blevet undersøgt og er under stadig udvikling, og det ville her være interessant at undersøge hvordan man kan beskrive ikke lokale egenskaber, som f.eks. charge–transfer–excitations, med lokale metodikker.

Computing systems

- **Titan**: On **Titan**, each compute node contains an AMD Opteron^{TM} 6274 (Interlagos) CPU which splits the node into two NUMA (Non-Uniform Memory Access) nodes (dies), each with 4 "bulldozer" floating-point units accompanied by two 2.2 GHz integer cores (i.e. 16/8 integer cores/floating-point units per NUMA node), and 16 GB of RAM (32GB per node). All compute nodes contain one NVIDIA Kepler^{TM} GPU with 6GB of DDR5 memory. The network is a Cray Gemini^{TM}architecture which is set up as a 3D torus.
- **Chester**: Each **Chester** node is configured like, a **Titan** node. There are only 80 nodes on **Chester**
- **Grendel**: **Grendel** is a local commodity cluster at Aarhus University with currently 13 queues of nodes of very different specification (`http://www.cscaa.dk/grendel/overview/`).

Bibliography

[1] G. E. Moore, Solid-State Circuits Society Newsletter, IEEE **11**, 33 (2006).

[2] Homepage of the nobel organization, Available online.

[3] V. Fock, Zeitschrift für Physik **61**, 126 (1930).

[4] F. Coester and H. Kümmel, Nuc. Phys. **17**, 477 (1960).

[5] J. Čížek, JCP **45**, 4256 (1966).

[6] J. Čížek and J. Paldus, Physica Scripta **21**, 251 (1980).

[7] E. Schrödinger, Ann. d. Phys. **385**, 437 (1926).

[8] M. Born and R. Oppenheimer, Ann. d. Phys. **389**, 457 (1927).

[9] T. Helgaker, P. Jørgensen, and J. Olsen, *Molecular Electronic Structure Theory, First Edition*, Wiley, Chichester, England, 2000.

[10] I. Shavitt, Mol. Phys. **94**, 3 (1998).

[11] C. Møller and M. S. Plesset, Phys. Rev. **46**, 618 (1934).

[12] G. Purvis and R. J. Bartlett, J. Chem. Phys. **76**, 1910 (1982).

[13] K. Raghavachari, G. W. Trucks, J. A. Pople, and M. Head-Gordon, Chem. Phys. Letters **157**, 479 (1989).

[14] J. Noga and R. J. Bartlett, J. Chem. Phys. **86**, 7041 (1987).

[15] H. Koch, O. Christiansen, R. Kobayashi, P. Jørgensen, and T. Helgaker, Chem. Phys. Lett. **228**, 233 (1994).

[16] P. Pulay, Chem. Phys. Lett. **73**, 393 (1980).

[17] G. E. Scuseria, T. J. Lee, and H. F. Schaefer III, Chem. Phys. Lett. **130**, 236 (1986).

[18] M. Ziolkowski, V. Weijo, P. Jorgensen, and J. Olsen, J. Chem. Phys. **128**, 204105 (2008).

[19] K. Raghavachari, G. W. Trucks, and M. Pople, J. A. Head-Gordon, Chem. Phys. Lett. **157**, 479 (1989).

[20] I. Shavitt and R. J. Bartlett, *Many–Body Methods in Chemistry and Physics: Many–Body Perturbation Theory and Coupled–Cluster Theory*, Cambridge University Press, Cambridge, UK, 2009.

[21] P. Piecuch, S. A. Kucharski, K. Kowalski, and M. Musiał, Comp. Phys. Comm. **149**, 71 (2002).

[22] H. Koch, O. Christiansen, P. Jørgensen, A. Sanchez de Merás, and T. Helgaker, J. Chem. Phys. **106**, 1808 (1997).

[23] R. Ahlrichs, P. Scharf, and C. Ehrhardt, J. Chem. Phys. **82**, 890 (1985).

[24] P. Wormer, F. Visser, and J. Paldus, J. Comput. Phys. **48**, 23 (1982).

[25] J. H. van Lenthe and P. Pulay, Journal of Computational Chemistry **11**, 1164 (1990).

[26] Y. Saad, *Iterative Methods for Sparse Linear Systems*, SIAM, Philadelphia, PA, USA, 2nd edition, 2003.

[27] B. Jansík, S. Høst, K. Kristensen, and P. Jørgensen, J. Chem. Phys. **134** (2011).

[28] T. H. Dunning, J. Chem. Phys. **90**, 1007 (1989).

[29] LSDALTON*, a linear scaling molecular electronic structure program, Release Dalton2011, see http://daltonprogram.org*, 2011.

[30] T. Kolda and B. Bader, SIAM Review **51**, 455 (2009).

[31] B. W. Bader et al., Matlab tensor toolbox version 2.6, Available online, 2015.

[32] A. C.Limache and S. R. Fredini, ltensor, Available online.

[33] E. Solomonik, D. Matthews, J. Hammond, and J. Demmel, in *Parallel Distributed Processing (IPDPS)*, pages 813–824, 2013.

[34] M. E. Harding, T. Metzroth, J. Gauss, and A. A. Auer, JCTC **4**, 64 (2008).

[35] M. Valiev et al., Comp. Phys. Comm. **181**, 1477 (2010).

[36] A. Asadchev and M. S. Gordon, JCTC **9**, 3385 (2013).

[37] R. Kobayashi and A. P. Rendell, Chem. Phys. Lett. **265**, 1 (1997).

[38] I.-M. Høyvik, K. Kristensen, B. Jansík, and P. Jørgensen, J. Chem. Phys. **136**, 014105 (2012).

[39] K. Kristensen, M. Ziółkowski, B. Jansík, T. Kjærgaard, and P. Jørgensen, J. Chem. Theory Comput. **7**, 1677 (2011).

[40] S. Saebo and P. Pulay, J. Chem. Phys. **86**, 914 (1987).

[41] G. E. Scuseria, C. L. Janssen, and H. F. Schaefer III, J. Chem. Phys. **89**, 7382 (1988).

[42] K. Kristensen et al., Mol. Phys. **111**, 1196 (2013).

[43] K. Kristensen, P. Ettenhuber, J. J. Eriksen, F. Jensen, and P. Jrgensen, J. Chem. Phys. **142**, (2015).

[44] B. Liu and A. McLean, J. Chem. Phys. **59**, 4557 (1973).

[45] S. F. Boys and F. Bernardi, Mol. Phys. **19**, 553 (1970).

[46] J. Daudey, P. Claverie, and J. Malrieu, Int. J. Quantum Chem. **8**, 1 (1974).

[47] J. H. van Lenthe, T. van Dam, F. B. van Duijneveldt, and L. M. J. Kroon-Batenburg, Faraday Symp. Chem. Soc. **19**, 125 (1984).

[48] M. Gutowski, J. van Lenthe, J. Verbeek, F. van Duijneveldt, and G. Chałasinski, Chem. Phys. Letters **124**, 370 (1986).

[49] M. Gutowski, F. B. van Duijneveldt, G. Chalasinki, and L. Piela, Mol. Phys. **61**, 233 (1987).

[50] G. Chalasinski and M. Gutowski, Chem. Rev. **88**, 943 (1988).

[51] J. H. van Lenthe, J. G. C. M. van Duijneveldt-van de Rijdt, and F. B. van Duijneveldt, Adv. Chem. Phys. **69**, 521 (1987).

[52] S. Scheiner, *Reviews in Computational Chemistry*, page 165, VCH, New York, 1991.

[53] I. Mayer and L. Turi, J. Mol. Struct. (Theochem) **227**, 43 (1991).

[54] M. Gutowski and G. Chal, J. Chem. Phys. **98**, 5540 (1993).

[55] D. Cook, J. A. Sordo, and T. L. Sordo, Int. J. Quantum Chem. **48**, 375 (1993).

[56] F. B. van Duijneveldt, J. G. C. M. van Duijneveldt-vande Rijdt, and J. H. van Lenthe, Chem. Rev. **94**, 1873 (1994).

[57] R. Wieczorek, L. Haskamp, and J. J. Dannenberg, J. Phys. Chem. A **108**, 6713 (2004).

[58] L. M. Mentel and E. J. Baerends, J. Chem. Theory Comput. **10**, 252 (2013).

[59] M. Schütz, S. Brdarski, P.-O. Widmark, R. Lindh, and G. Karlström, J. Chem. Phys. **107**, 4597 (1997).

[60] L. A. Burns, M. S. Marshall, and C. D. Sherrill, J. Chem. Theory Comput. **10**, 49 (2014).

[61] B. Brauer, M. K. Kesharwani, and J. M. Martin, J. Chem. Theory Comput. **10**, 3791 (2014).

[62] H. Kruse and S. Grimme, J. Chem. Phys. **136**, 154101 (2012).

[63] B. Jeziorski and W. Kolos, *Molecular Interactions*, page 1, Wiley, New York, 1982.

[64] B. Jeziorski, R. Moszynski, and K. Szalewicz, Chem. Rev. **94**, 1887 (1994).

[65] I. Mayer, Int. J. Quantum Chem. **23**, 341 (1983).

[66] J. Noga and A. Vibok, Chem. Phys. Letters **180**, 114 (1991).

[67] J. M. Cullen, Int. J. Quantum Chem.: Quantum Chem, Symp. **25**, 193 (1991).

[68] A. J. Sadlej, J. Chem. Phys. **95**, 6705 (1991).

[69] A. Halkier et al., Chem. Phys. Letters **286**, 243 (1998).

[70] P. Pulay, Chem. Phys. Letters **100**, 151 (1983).

[71] C. Hampel and H.-J. Werner, J. Chem. Phys. **104**, 6286 (1996).

[72] M. Schütz, G. Hetzer, and H.-J. Werner, J. Chem. Phys. **111**, 5691 (1999).

[73] H.-J. Werner and M. Schütz, J. Chem. Phys. **135**, 144116 (2011).

[74] C. Krause and H.-J. Werner, Phys. Chem. Chem. Phys. **14**, 7591 (2012).

[75] H. Stoll, Chem. Phys. Letters **191**, 548 (1992).

[76] J. Friedrich, M. Hanrath, and M. Dolg, J. Chem. Phys. **126**, 154110 (2007).

[77] G. E. Scuseria and P. Y. Ayala, J. Chem. Phys. **111**, 8330 (1999).

[78] O. Christiansen, P. Manninen, P. Jørgensen, and J. Olsen, J. Chem. Phys. **124**, 084103 (2006).

[79] P. Y. Ayala and G. E. Scuseria, J. Chem. Phys. **110**, 3660 (1999).

[80] D. S. Lambrecht, B. Doser, and C. Ochsenfeld, J. Chem. Phys. **123**, 184102 (2005).

[81] B. Doser, D. S. Lambrecht, and C. Ochsenfeld, Phys. Chem. Chem. Phys. **10**, 3335 (2008).

[82] P. E. Maslen, M. S. Lee, and M. Head-Gordon, Chem. Phys. Letters **319**, 205 (2000).

[83] S. Li, J. Ma, and Y. Jiang, J. Comput. Chem. **23**, 237 (2002).

[84] W. Li and P. Piecuch, J. Phys. Chem. A **114**, 8644 (2010).

[85] W. Li, Y. Guo, and S. Li, Phys. Chem. Chem. Phys. **14**, 7854 (2012).

[86] N. Flocke and R. J. Bartlett, J. Chem. Phys. **121**, 10935 (2004).

[87] D. G. Fedorov and K. Kitaura, J. Chem. Phys. **123**, 134103 (2005).

[88] T. Ishikawa and K. Kuwata, Chem. Phys. Letters **474**, 195 (2009).

[89] D. G. Fedorov, T. Nagata, and K. Kitaura, Phys. Chem. Chem. Phys. **14**, 7562 (2012).

[90] M. Kobayashi and H. Nakai, J. Chem. Phys. **129**, 044103 (2008).

[91] M. Katouda, M. Kobayashi, H. Nakai, and S. Nagase, J. Comput. Chem. **32**, 2756 (2011).

[92] M. Kobayashi and H. Nakai, Phys. Chem. Chem. Phys. **14**, 7629 (2012).

[93] A. A. Auer and M. Nooijen, J. Chem. Phys. **125**, 024104 (2006).

[94] F. Neese, F. Wennmohs, and A. Hansen, J. Chem. Phys. **130**, 114108 (2009).

[95] C. Riplinger and F. Neese, J. Chem. Phys. **138**, 034106 (2013).

[96] J. Yang, Y. Kurashige, F. R. Manby, and G. K. L. Chan, J. Chem. Phys. **134**, 044123 (2011).

[97] J. Yang, G. K. L. Chan, F. R. Manby, M. Schütz, and H.-J. Werner, J. Chem. Phys. **136**, 114105 (2012).

[98] M. Ziółkowski, B. Jansík, T. Kjærgaard, and P. Jørgensen, J. Chem. Phys. **133**, 014107 (2010).

[99] M. Ziółkowski, B. Jansík, P. Jørgensen, and J. Olsen, J. Chem. Phys. **131**, 124112 (2009).

[100] B. Jansík, S. Høst, K. Kristensen, and P. Jørgensen, J. Chem. Phys. **134**, 194104 (2011).

[101] I.-M. Høyvik, B. Jansík, and P. Jørgensen, J. Chem. Theory Comput. **8**, 3137 (2012).

[102] K. Kristensen, P. Jørgensen, B. Jansík, T. Kjærgaard, and S. Reine, J. Chem. Phys. **137**, 114102 (2012).

[103] M. Feyereisen, G. Fitzgerald, and A. Komornicki, Chem. Phys. Lett. **208**, 359 (1993).

[104] E. G. Hohenstein, R. M. Parrish, and T. J. Martínez, J. Chem. Phys. **137**, (2012).

[105] K. Aidas et al., WIRES **4**, 269 (2014).

[106] J. W. Boughton and P. Pulay, J. Comput. Chem. **14**, 736 (1993).

[107] M. D. Hanwell et al., Journal of Cheminformatics **4**, 17 (2012).

[108] J. Dongarra, K. London, S. Moore, P. Mucci, and D. Terpstra, *Using PAPI for Hardware Performance Monitoring on Linux Systems. In Conference on Linux Clusters: The HPC Revolution*, Linux Clusters Institute, Urbana, Illinois, 2002.

[109] K. Kristensen et al., Phys. Chem. Chem. Phys. **14**, 15706 (2012).

[110] C. Riplinger, B. Sandhoefer, A. Hansen, and F. Neese, J. Chem. Phys. **139**, (2013).

[111] D. P. Tew, B. Helmich, and C. Hättig, The Journal of Chemical Physics **135**, (2011).

[112] Z. Rolik, L. Szegedy, I. Ladjánszki, B. Ladóczki, and M. Kállay, The Journal of Chemical Physics **139**, (2013).

[113] W. Meyer, J. Chem. Phys. **58**, 1017 (1973).

[114] F. Neese, F. Wennmohs, and A. Hansen, J. Chem. Phys. **130**, (2009).

[115] P.-O. Löwdin, Phys. Rev. **97**, 1474 (1955).

[116] E. R. Davidson, Rev. Mod. Phys. **44**, 451 (1972).