Corrélation électronique et parallélisme à grande échelle

Anthony Scemama

Université Toulouse III — Paul Sabatier

Habilitation à Diriger les Recherches

Corrélation électronique et parallélisme à grande échelle

Anthony Scemama

Laboratoire de Chimie et Physique Quantiques

Discipline : Chimie Physique

Soutenue le 8/9/2017 devant la commission d'examen formée de :

Mme Marie-Bernadette Lepetit	Directrice de Recherche CNRS	Rapporteure
M. Daniel Borgis	Directeur de Recherche CNRS	Rapporteur
M. Gérald Monard	Professeur de l'Université de Lorraine	Rapporteur
M. Michel Caffarel	Directeur de Recherche CNRS	Examinateur
M. Franck Jolibois	Professeur de l'Université Toulouse III	Président

Table des matières

Chapitre 1

Introduction

La chimie quantique est une discipline qui a la particularité d'être extrêmement demandeuse en terme de ressources de calcul. Ainsi, l'utilisation des ordinateurs remonte aux débuts de la discipline, et la façon de penser les méthodes et les nouveaux algorithmes a toujours été intimement liée à la nature de l'architecture des calculateurs et à l'évolution de leurs capacités. Depuis le début de l'informatique, c'est le modèle d'architecture séquentielle conçu par Turing en 1936 qui prévaut.[1] Dans ce modèle, les instructions sont stockées en mémoire comme des nombres et exécutées par une unité de contrôle. Une unité arithmétique et logique (ALU) réalise les calculs en utilisant comme opérandes des données, elles aussi stockées en mémoire. Jusqu'en 2004, ce modèle intrinsèquement séquentiel a conduit l'industrie de l'informatique à concevoir des processeurs avec une fréquence d'horloge toujours plus élevée dans le but d'accroître le nombre d'instructions exécutées par seconde. Bien qu'un record de 500 GHz a été battu avec un transistor refroidi à $-269°C$ par de l'hélium liquide,[2] cette augmentation de fréquence est devenue impossible pour la diffusion à grande échelle des processeurs en raison des problèmes de consommation électrique et de dissipation thermique. En effet, la dissipation thermique D d'un processeur est donnée par

$$D = F \times V^2 \times C \tag{1.1}$$

où F est la fréquence du processeur, V la tension appliquée et C est une constante liée à la finesse de gravure. La tension nécessaire pour assurer la stabilité du processeur dépendant linéairement de sa fréquence, la dissipation thermique d'un processeur présente donc un comportement cubique avec sa fréquence. Si l'on veut doubler le nombre d'opérations effectuées par seconde, doubler le nombre d'unités de calcul aura un bien meilleur bilan thermique et électrique plutôt que de doubler la fréquence d'un processeur. Ainsi, si la puissance de calcul disponible continue à doubler tous les 18 mois cette dernière décennie, ce n'est plus parce que la fréquence d'horloge est accrue régulièrement mais c'est parce que le nombre d'unités de calcul est augmenté en gardant la fréquence relativement constante entre 2 et 3 GHz. Cette barrière de la fréquence a fait naître vers 2006 les processeurs multi-cœurs généralistes où plusieurs unités de calculs sont regroupées au sein d'un même processeur.

C'est cette histoire de la chimie quantique développée dans un contexte purement séquentiel qui fait que la majorité des algorithmes que nous utilisons actuellement dans les méthodes de fonction d'onde sont séquentiels, même si dans le nouveau contexte ce ne sont plus les plus pertinents. Un exemple important est celui des méthodes itératives qui sont aujourd'hui extrêmement répandues dans la discipline et qui ne sont pas naturellement adaptées aux nouvelles architectures. Plus généralement, il devient nécessaire de se dégager de l'ancienne façon de penser qui présupposait une architecture séquentielle. Par exemple, l'hypothèse que la réduction du nombre d'opérations implique nécessairement une réduction du temps de restitution n'est vraie que

dans un contexte séquentiel, mais pas nécessairement dans un contexte où plusieurs unités de calcul travaillent simultanément.

Utiliser pleinement la puissance des ordinateurs actuels nécessite donc d'abandonner la logique du calcul séquentiel et de développer les méthodes en tenant compte de la possibilité de réaliser des calculs en parallèle sur plusieurs centaines de milliers de cœurs de calcul. Aujourd'hui, les machines les plus puissantes comportent quelques millions de cœurs, et un accélérateur GPU en comporte à lui seul de l'ordre de quelques milliers, et cela va continuer à augmenter. Être capable de mobiliser simultanément et efficacement (c'est-à-dire sans attente) plusieurs millions de cœurs sur un même calcul constitue un vrai changement de perspective pour le calcul scientifique : on parle depuis 2005 de *changement de paradigme.*[3]

Certaines branches du calcul scientifique, comme celles qui reposent sur des méthodes de décomposition de domaine, se sont naturellement adaptées à ce changement, mais ce n'est pas la cas de toutes les communautés. Une conséquence pratique observée aujourd'hui dans les centres de calcul, et qui peut surprendre les utilisateurs, est qu'au contraire de ce qui se passait il y a quelques années, l'arrivée de nouvelles machines plus performantes n'implique pas nécessairement une diminution des temps d'exécution des programmes comme cela se passait quand il s'agissait d'augmenter la fréquence des processeurs. Obtenir de meilleures performances doit le plus souvent passer par une étape de réécriture des algorithmes (voire, de changement) en vue de profiter au mieux de la multiplication des unités de calcul. D'autre part il se pose aussi le problème de la portabilité et de la pérennité de la performance des applications sur des architectures très différentes, et cela impose de généraliser l'utilisation de bibliothèques dans les parties du code où la performance est critique.

Mes travaux de recherche portent sur le développement méthodologique en chimie quantique, dans un contexte où les architectures matérielles sont massivement parallèles. Cela implique donc de remettre en question l'utilisation de certains choix qui ont été faits à l'époque où les machines étaient séquentielles, et de proposer de nouvelles approches mieux adaptées au matériel actuel et futur. On peut constater depuis quelques années un renouveau dans les méthodes de fonction d'onde, concordant avec l'arrivée des processeurs multi-cœurs. Parmi ceux-ci on peut citer le *Full Configuration Interaction Quantum Monte Carlo* (FCIQMC)[4], les algorithmes stochastiques pour les théories de perturbation[5] et pour le Coupled Cluster[6], ou encore les développements dans le domaine des méthodes Monte Carlo Quantique (QMC) qui permettent aujourd'hui de traiter plusieurs centaines de milliers de déterminants de Slater[7]. Le FCIQMC, par exemple, permet aujourd'hui de converger des espaces actifs de 50 électrons dans 50 orbitales, ce qui était inimaginable il y a encore une dizaine d'années où la limite était plutôt autour de 16 électrons dans 16 orbitales. Tous ces développements ont plusieurs points communs :

1. ils peuvent exploiter les machines parallèles,
2. il ne réalisent pas le calcul complet, mais un calcul tronqué sur les éléments les plus importants,
3. les éléments les plus importants sont choisis d'une manière automatique au cours de l'évolution du calcul,
4. le calcul converge vers le calcul complet lorsque le temps CPU devient infini.

Les algorithmes stochastiques ne sont pas les seuls à avoir ces propriétés. D'ailleurs, l'introduction d'une composante déterministe dans un algorithme stochastique est souvent l'ingrédient qui permet d'aller encore au-delà. Il est donc nécessaire

de développer également des méthodes déterministes permettant une sélection automatique des éléments les plus importants, que l'on pourra par la suite combiner avec des méthodes stochastiques. Dans cette idée, nous avons contribué à la renaissance de la sélection de déterminants par perturbation (CIPSI) via la thèse d'Emmanuel Giner[8] dans le but de construire des fonctions d'ondes multi-déterminantales compactes pour les calculs QMC et de contrôler l'erreur des nœuds fixés, l'unique erreur des méthodes QMC. Étant donné que nous sommes partis de zéro, nous avons pu commencer à écrire un code de chimie quantique, le *Quantum Package*, pensé pour tourner sur des machines parallèles, et qui utilise l'algorithme CIPSI pour implémenter toutes les méthodes d'interactions de configurations. Aujourd'hui, nous avons donc un programme qui permet de faire du CISD, du MP2, du CAS, du MR-CI, du DDCI, du CCSD(T), du MR-CCSD(T) et du Full-CI, en utilisant la sélection perturbative des déterminants importants, et une implémentation massivement parallèle grâce au travail réalisé pendant la thèse de Yann Garniron.[9]

Ce programme n'a pas vocation à remplacer les grands programmes existants tels que GAMESS[10] ou Molpro[11], mais il s'agit plutôt d'un laboratoire virtuel pour expérimenter des approches alternatives permettant de montrer à la communauté que les méthodes de fonctions d'ondes peuvent elles aussi bénéficier des supercalculateurs de demain, voire même des infrastructures de type grille ou Cloud. Pendant la thèse de Thomas Applencourt,[12] nous nous sommes donnés comme challenge de ne pas utiliser l'API standard MPI car elle n'est pas bien adaptée aux calculs distribués dans le Cloud. Nous avons développé un serveur de tâches reposant sur la bibliothèque ZeroMQ[13] permettant d'implémenter la tolérance aux pannes, les communications sur des réseaux peu fiables et peu performants comme le réseau internet, et la gestion dynamique des ressources (ajout/suppression de processeurs au cours du calcul). A titre d'exemple, nous avons pu lancer en 2015 un calcul QMC utilisant simultanément des ressources d'un supercalculateur (CALMIP, Toulouse) et de deux Clouds (Strasbourg et Orsay), validant cette approche.

Dans ce document, je commence par présenter les différentes méthodes de la chimie quantique auxquelles je m'intéresse tout particulièrement. Cette section est plutôt destinée aux spécialistes du calcul scientifique qui ne connaissent pas la chimie quantique. Dans un second temps, j'explique en quelques mots pourquoi il est difficile d'utiliser les machines parallèles avec les algorithmes standard de la chimie quantique, et comment mon travail se situe par rapport à cette problématique. Puis, je présente les travaux qui illustrent au mieux mon activité, et enfin je présente les perspectives qui définissent mon projet de recherche à plus long terme.

Chapitre 2

Contexte général

2.1 Approximations de la chimie quantique

La chimie quantique décrit la structure électronique des systèmes moléculaires en tenant compte du caractère quantique des interactions entre les électrons. On se place en général dans l'approximation de Born-Oppenheimer où les N électrons se déplacent beaucoup plus vite que les M noyaux, si bien que le nuage électronique est toujours relaxé pour un jeu de positions nucléaires $\mathbf{R}$. Les noyaux peuvent donc être considérés comme des charges ponctuelles fixes, et on recherche la fonction d'onde électronique Ψ qui minimise l'énergie en essayant de résoudre l'équation de Schrödinger. Pour un système à N électrons avec pour coordonnées $\{\mathbf{r}_1, \ldots, \mathbf{r}_N\}$,

$$\widehat{H}\Psi(\mathbf{r}_1, \ldots, \mathbf{r}_N) = E_0\Psi(\mathbf{r}_1, \ldots, \mathbf{r}_N) \tag{2.1}$$

$\widehat{H}$ est l'opérateur hamiltonien électronique non-relativiste :

$$\widehat{H} = \widehat{T}_e + \widehat{V}_{ee} + \widehat{V}_{eN} + \widehat{V}_{NN}. \tag{2.2}$$

$\widehat{T}_e$ est l'opérateur d'énergie cinétique des électrons

$$\widehat{T}_e = -\frac{1}{2}\sum_{i=1}^{N}\Delta_i \tag{2.3}$$

et $\widehat{V}$ est l'opérateur d'énergie potentielle

$$\widehat{V}_{ee} = \sum_{i=1}^{N}\sum_{j>i}^{N}\frac{1}{|\mathbf{r}_i - \mathbf{r}_j|} \tag{2.4}$$

$$\widehat{V}_{eN} = \sum_{i=1}^{N}\sum_{A=1}^{M}\frac{-Z_A}{|\mathbf{r}_i - \mathbf{R}_A|} \tag{2.5}$$

$$\widehat{V}_{NN} = \sum_{A=1}^{M}\sum_{B>A}^{M}\frac{Z_A Z_B}{|\mathbf{R}_A - \mathbf{R}_B|}. \tag{2.6}$$

$$\tag{2.7}$$

L'équation de Schrödinger (Eq.(2.1)) est une équation aux dérivées partielles dans un espace de dimension $3N$ (chaque coordonnée x, y, et z des électrons). Obtenir l'énergie électronique exacte E_0 est donc une tâche dont la complexité augmente très rapidement avec le nombre d'électrons. Il est alors nécessaire d'avoir recours à des approximations pour permettre aux calculs d'être réalisables dans un temps satisfaisant. On comprend bien que plus le système étudié sera grand, plus les approximations utilisées seront sévères.

Le choix des approximations nécessite une certaine expertise de l'utilisateur. En effet, toute la chimie repose sur de très petites différences d'énergies : la *précision chimique* est de l'ordre de 1 kcal/mol, soit 0.0016 unités atomiques (u.a.) quelque soit la taille du système, et les énergies totales sont de l'ordre de quelques centaines voire quelques milliers d'unités atomiques. Il est impératif d'avoir des schémas d'approximation où les erreurs sont transférables d'un système à l'autre, si bien que les erreurs tendent à s'annuler lors d'un calcul de différence d'énergies. Plusieurs types d'approximations peuvent être faites. Dans un premier temps, on considère l'hamiltonien exact $\widehat{H}$ et on recherche une fonction d'onde approchée. Puisqu'il s'agit d'un problème de minimisation de l'énergie, l'énergie associée à la fonction d'onde approchée obéit au *principe variationnel*, et est toujours toujours supérieure (ou égale) à l'énergie exacte : $E \geq E_0$. Une autre approche consiste à estimer l'énergie par projection, comme dans la théorie des perturbations par exemple. Dans ce cas, l'énergie n'obéit plus au principe variationnel et peut être supérieure ou inférieure à l'énergie exacte.

À cause du principe de Pauli, une condition supplémentaire doit être imposée. $\Psi(\mathbf{r}_1, \ldots, \mathbf{r}_N)$ doit être antisymétrique par rapport à la permutation des électrons. La forme la plus simple qui permet d'assurer cette condition est un produit antisymétrisé de fonctions mono-électroniques $\{\varphi_i\}$ (orbitales moléculaires, OM) :

$$\Psi(\mathbf{r}_1, \mathbf{r}_2) = \varphi_1(\mathbf{r}_1)\varphi_2(\mathbf{r}_2) - \varphi_1(\mathbf{r}_2)\varphi_2(\mathbf{r}_1) \tag{2.8}$$

qui correspond à un déterminant. Formellement, la condition d'antisymétrie est réalisée en introduisant deux fonctions orthonormales α et β qui caractérisent le spin de l'électron. Une formulation sans spin (spin-free) a ensuite été proposée,[14, 15] permettant de s'affranchir des fonctions de spin en définissant alors deux types d'électrons avec un spin α ou β. Ainsi, on peut ré-écrire la fonction d'onde totale comme

$$\Psi(\mathbf{r}_1, \ldots, \mathbf{r}_{N_\alpha}, \mathbf{r}_{N_\alpha+1}, \ldots, \mathbf{r}_N; \alpha_1, \ldots, \alpha_{N_\alpha}, \beta_{N_\alpha+1}, \ldots, \beta_N) = \begin{vmatrix} \varphi_1(\mathbf{r}_1) & \ldots & \varphi_1(\mathbf{r}_{N_\alpha}) \\ \vdots & \ldots & \vdots \\ \varphi_{N_\alpha}(\mathbf{r}_1) & \ldots & \varphi_{N_\alpha}(\mathbf{r}_{N_\alpha}) \end{vmatrix} \begin{vmatrix} \varphi_1(\mathbf{r}_{N_\alpha+1}) & \ldots & \varphi_1(\mathbf{r}_N) \\ \vdots & \ldots & \vdots \\ \varphi_{N_\beta}(\mathbf{r}_{N_\alpha+1}) & \ldots & \varphi_{N_\beta}(\mathbf{r}_N) \end{vmatrix} \tag{2.9}$$

Dans la suite du texte nous appellerons *déterminant de Slater* cette forme de fonction d'onde.

2.1.1 Approximation de base finie

La première approximation est la définition d'un espace de fonctions mono-électroniques $\{\chi_k(\mathbf{r})\}$ (fonctions de base, orbitales atomiques, OAs) sur lesquelles les orbitales moléculaires (OMs) sont développées :

$$\varphi_i(\mathbf{r}) = \sum_k C_{ki}\chi_k(\mathbf{r}). \tag{2.10}$$

Cet espace restreint les variations possibles de la fonction d'onde dans la procédure de minimisation de l'énergie. Pour les systèmes qui ne sont pas périodiques comme une molécule dans le vide, le choix le plus courant est d'utiliser des fonctions gaussiennes de la forme :

$$\chi_k(\mathbf{r}) = (x - X_k)^{n_k}(y - Y_k)^{p_k}(z - Z_k)^{q_k} \sum_l^{N_k} \xi_{lk} e^{-\gamma_{lk}|\mathbf{r}-\mathbf{R}_k|^2} \tag{2.11}$$

faisant apparaître les coordonnées des électrons $\mathbf{r} = \{x, y, z\}$ et des noyaux $\mathbf{R}_k = \{X_k, Y_k, Z_k\}$. Les paramètres (n_k, p_k, q_k) sont définis par les nombres quantiques l et m de l'orbitale, et les paramètres $(N_k, \xi_{lk}, \gamma_{lk})$ sont des valeurs prétabulées caractérisant la base atomique choisie.

2.1.2 Méthode Hartree-Fock et au-delà

Dans l'approximation de Hartree-Fock, on représente la fonction d'onde comme un déterminant de Slater. La méthode consiste à chercher le jeu d'orbitales moléculaires $\{\varphi_i\}$ qui minimise l'énergie. Les paramètres variationnels sont les coefficients des orbitales moléculaires $\{C_{ki}\}$.

Dans cette approximation, les positions des électrons de même spin sont corrélées par la structure antisymétrique de la fonction d'onde. À ce niveau d'approximation, le seul effet qui n'est pas pris en compte est la corrélation des positions des électrons due à leur interaction coulombienne (*corrélation dynamique*) ou due aux interférences entre états électroniques quasi-dégénérés (*corrélation statique*). On considère donc la méthode Hartree-Fock comme le niveau zéro de la corrélation électronique, et on définit *l'énergie de corrélation* comme la différence entre l'énergie exacte et l'énergie Hartree-Fock obtenue dans une base infinie d'OAs, c'est-à-dire la *limite Hartree-Fock*. La limite Hartree-Fock prend en compte de l'ordre de 99% de l'énergie électronique totale. Cependant, une telle précision est généralement loin d'être suffisante pour décrire des processus chimiques qui impliquent des différences d'énergie bien en dessous de 1% de l'énergie totale.

Pour décrire les effets de corrélation électronique, il faut exprimer la fonction d'onde non pas dans une base de fonctions mono-électroniques, mais dans une base de fonctions à N électrons.[1] Pour cela, on commence par construire une base orthonormale à partir de la base des OAs. Puis, on lui applique une rotation qui permet de retrouver toutes les orbitales moléculaires du déterminant de Hartree-Fock qu'on appelle les *orbitales occupées*. Le complément orthogonal des orbitales occupées constitue l'ensemble des *orbitales virtuelles*. On peut maintenant constituer la base orthonormale multi-électronique composée de tous les déterminants de Slater $\{|D\rangle\}$ que l'on peut construire en utilisant cet ensemble d'orbitales moléculaires, c'est-à-dire toutes les façons de mettre simultanément N_α électrons de spin α dans N_{OM} orbitales moléculaires et N_β électrons de spin β dans N_{OM} orbitales moléculaires. La taille de cet espace croît exponentiellement vite avec le nombre d'électrons quand celui-ci est petit devant le nombre de OMs :

$$\aleph = \binom{N_{\mathrm{OM}}}{N_\alpha}\binom{N_{\mathrm{OM}}}{N_\beta} \tag{2.12}$$

L'orthonormalité de la base des déterminants de Slater a des conséquences extrêmement importantes. Grâce aux règles de Slater-Condon, provenant de la nature au plus bi-électronique de l'hamiltonien, on peut maintenant simplifier toutes les intégrales à $3N$ dimensions intervenant dans les éléments de matrices des opérateurs à une ou deux particules. Les intégrales à $3N$ dimensions deviennent des sommes d'intégrales à 3 ou 6 dimensions et les éléments de matrice deviennent donc calculables en des temps raisonnables.

1. Dans la suite du texte, le terme de *base* ne se limitera donc pas à la base atomique, mais pourra faire référence à la base des déterminants.

2.1.3 Méthodes post-Hartree-Fock variationnelles

La résolution de l'équation de Schrödinger dans une base de déterminants de Slater est appelée *interaction de configurations* (CI). Il s'agit de trouver l'état propre de plus basse énergie de l'hamiltonien exprimé dans la base orthonormale des déterminants :

$$|\Psi\rangle = \sum_i c_i |D_i\rangle \tag{2.13}$$

Si la base des déterminants est complète on parle alors d'*interaction de configurations complète* (Full-CI ou FCI) ce qui correspond à la résolution exacte de l'équation de Schrödinger, la seule approximation étant donc le caractère fini de la base des déterminants. Toutes les méthodes post-Hartree-Fock sont donc des approximations du Full-CI.

Les méthodes variationnelles visent à réduire le nombre de déterminants et à résoudre exactement le problème dans cette représentation réduite. On peut réaliser des interactions de configurations dans le sous-ensemble de déterminants qui ne diffèrent du déterminant de Hartree-Fock que par un certain nombre d'orbitales moléculaires. Si l'on considère une ou deux substitutions, il s'agit de la méthode CISD. Étant donné que la référence est le déterminant de Hartree-Fock, cette méthode est principalement utilisée pour décrire les effets de corrélation dynamique appliqués au déterminant de Hartree-Fock. Un problème majeur de cette approche est qu'elle ne possède pas la propriété d'extensivité en taille : l'énergie de deux systèmes A et B suffisamment éloignés pour ne pas interagir n'est pas égale à la somme des énergies de chacun des systèmes pris indépendamment,

$$E_{\text{CISD}}(A \dots B) \neq E_{\text{CISD}}(A) + E_{\text{CISD}}(B) \tag{2.14}$$

Cette erreur d'extensivité devient particulièrement importante sur le calcul de différences d'énergie lorsque le nombre de paires d'électrons en interaction diffère d'une situation à l'autre, comme dans le cas de réactions de fragmentation par exemple.

Une autre approche consiste à définir un petit sous-ensemble d'orbitales moléculaires occupées et virtuelles (de l'ordre de moins de 16 OMs en général) dans lesquelles on fera un Full-CI. Il s'agit de la définition d'un CAS (Complete Active Space). Si l'on minimise d'énergie en faisant l'interaction de configurations dans cet espace, et si l'on optimise également les coefficients des orbitales moléculaires, cela consiste à réaliser la méthode CAS-SCF.[16] À l'opposé de la méthode CISD, le CAS-SCF a la propriété d'extensivité, décrit particulièrement bien la corrélation statique, mais prend très peu en compte les effets de corrélation dynamique.

Si l'on veut considérer à la fois la corrélation statique et la corrélation dynamique, on peut définir un espace de déterminants composé d'un espace CAS et de toutes les simples et doubles substitutions que l'on peut faire sur les déterminants de plus gros poids dans l'espace CAS. On parle alors d'interaction de configurations multi-références (MR-CI), chacune des références correspondant aux déterminants de poids fort de l'espace actif. De la même façon que le CISD, le MR-CISD n'est pas extensif.

En plus de réduire l'expansion de la fonction d'onde en réduisant l'espace des déterminants, on peut également rendre sa représentation plus compacte afin de concentrer un maximum d'information sur un petit nombre de déterminants. Une possibilité est d'utiliser un jeu d'orbitales moléculaires locales. En effet, cela permet d'annuler un grand nombre d'éléments de la matrice hamiltonienne, et le vecteur de l'état fondamental recherché est beaucoup plus creux qu'en utilisant les orbitales

canoniques (Hartree-Fock). En définissant un seuil au dessous duquel les coefficients sont mis à zéro, on peut donc réduire considérablement la taille de l'expansion de la fonction d'onde avec une erreur assez faible. Malheureusement, ce type d'approche n'est pas adapté aux petites molécules car les orbitales moléculaires ne peuvent pas être suffisamment bien localisées à cause de leur contrainte d'orthogonalité.

2.1.4 Méthodes post-Hartree-Fock par projection

Si l'on exprime l'équation de Schrödinger projetée sur le déterminant de Hartree-Fock, on a

$$\langle D_0|\widehat{H}|\Psi\rangle = E\langle D_0|\Psi\rangle, \tag{2.15}$$

et l'on peut remarquer, grâce aux règles de Slater-Condon, que seuls les déterminants qui ne diffèrent que par une ou deux orbitales moléculaires du déterminant de Hartree-Fock (les simples et doubles excitations) ont un élément de matrice non-nul.

$$\sum_i c_i\langle D_0|\widehat{H}|D_i\rangle = E\sum_i c_i\langle D_0|D_i\rangle \tag{2.16}$$

$$(H_{00} - E)c_0 + \sum_{i\in\{\mathrm{SD}\}} c_i H_{0i} = 0 \tag{2.17}$$

Si l'on se place en normalisation intermédiaire ($\langle D_0|\Psi\rangle = 1$), l'énergie de corrélation est exprimée seulement à partir des coefficients des simples et doubles excitations :

$$E - E_{\mathrm{HF}} = \sum_{i\in\{\mathrm{SD}\}} c_i H_{0i}. \tag{2.18}$$

La première technique consiste à considérer la corrélation électronique comme une perturbation. Dans les méthodes de perturbation mono-référence au deuxième ordre (PT2), on considère l'espace CISD, mais seuls les éléments diagonaux ($H_{ii} = \langle D_i|\widehat{H}|D_i\rangle$) et les éléments hors-diagonaux impliquant le déterminant de Hartree-Fock ($H_{0i} = \langle D_0|\widehat{H}|D_i\rangle$) sont nécessaires, ce qui rend le calcul beaucoup moins coûteux que le calcul d'IC. Plusieurs choix d'approximations des coefficients de la fonction d'onde existent (Møller-Plesset ou MP2, Epstein-Nesbet, ...), et cette liberté permet de définir des modèles où l'énergie est extensive. On peut également utiliser une approche perturbative pour approcher l'énergie MR-CI en choisissant comme référence la fonction d'onde CAS-SCF au lieu du déterminant de Hartree-Fock. Dans ce cas, il s'agit de méthodes telles que CASPT2,[17] NEVPT2,[18] JM-MRPT2,[19] ...

Pour corriger l'erreur d'extensivité des méthodes d'IC, on peut choisir de modifier les éléments de la matrice hamiltonienne. De nombreux schémas existent, et peuvent tous se résumer à une extension de l'espace de départ des déterminants en ajoutant certains déterminants triplement et quadruplement excités par rapport au déterminant de Hartree-Fock qui sont nécessaires pour pouvoir représenter la fonction d'onde comme un produit. En effet, si deux systèmes A et B sont indépendants, la fonction d'onde de la réunion des systèmes doit s'exprimer comme $\Psi_{AB} = \Psi_A \otimes \Psi_B$. Pour qu'une double excitation sur A soit indépendante d'une double excitation sur B, il faut que la fonction d'onde permette de représenter la quadruple excitation qui est le produit de ces doubles excitations. Les déterminants additionnels permettant la factorisation constituent l'espace étendu. Le problème peut être exprimé de façon plus compacte en réalisant un "habillage" de la matrice hamiltonienne définie dans l'espace de départ, c'est à dire une modification des éléments de la matrice sous l'influence des termes couplant l'espace de départ avec l'espace étendu. L'hamiltonien n'est donc plus l'hamiltonien exact dans ce cas, et l'énergie n'obéit plus au principe

variationnel. Parmi ces méthodes, on peut citer les méthodes de type CEPA[20, 21] (Coupled Electron Pair Approximation), CC[22, 23] (Coupled Cluster), $(SC)^2CI$[24] (Self Consistent Size Consistent CI),*etc.*

2.1.5 Monte Carlo Quantique

Les méthodes Monte Carlo quantique (QMC) permettent de résoudre de façon stochastique l'équation de Schrödinger. Dans plusieurs domaines de la physique comme la physique nucléaire, les systèmes de spins ou la spectroscopie infrarouge, ce sont des méthodes utilisées très couramment. A l'inverse, pour le problème du calcul de la structure électronique les méthodes déterministes (DFT ou post-Hartree-Fock) sont beaucoup plus populaires. La raison principale est que le coût du calcul est beaucoup plus important en terme de temps CPU pour les petits systèmes, ce qui est plutôt en faveur de l'utilisation de méthodes déterministes extrêmement précises telles que le Coupled-Cluster explicitement corrélé (f_{12}-CC).

Bien que très consommatrices de temps CPU, les méthodes QMC ont l'avantage d'être parfaitement adaptées aux calculateurs massivement parallèles, contrairement aux approches traditionnelles. Par exemple, en 2011 nous avons pu réaliser des simulations petaflopiques (de l'ordre de 10^{15} opérations flottantes par seconde) en utilisant la quasi-totalité du supercalculateur Curie (76 000 cœurs).[25] Des simulations d'une qualité équivalente avec des méthodes traditionnelles (CCSD(T)/cc-pVTZ, 434 électrons, 2930 fonctions de base) auraient consommé beaucoup plus de temps CPU, mais auraient surtout nécessité une quantité de mémoire bien supérieure à la totalité de la mémoire disponible sur la machine. Nous sommes donc à une période de transition où l'architecture des supercalculateurs est en faveur de méthodes consommatrices de calcul, peu gourmandes en terme de mémoire et stockage, et massivement parallélisables. Cela tend de plus en plus à démocratiser les méthodes comme le QMC.

La seconde raison qui fait que le QMC est toujours moins populaire que les approches déterministes est le manque de contrôle de l'erreur due à l'approximation des nœuds fixés, la seule approximation de la méthode. En effet, bien que les énergies totales soient de bien meilleure qualité que celles que l'on obtient avec les méthodes variationnelles usuelles, l'erreur due à l'approximation des nœuds fixés sur une différence d'énergie peut être plus importante que la différence d'énergie recherchée. Ce qui permet aux méthodes usuelles de donner des différences d'énergies sensées est une compensation d'erreur extrêmement bien contrôlée. Le contrôle de l'erreur en QMC occupe une grande partie de mon projet de recherche, et nous avons pu montrer récemment qu'il est désormais possible de contrôler l'approximation des nœuds fixés en combinant les méthodes post-Hartree-Fock avec les méthodes QMC.[26]

2.1.6 Autres méthodes

La théorie de la fonctionnelle de la densité (DFT) repose sur le fait que l'énergie peut être écrite comme une fonctionnelle de la densité mono-électronique : $E_0 = E[\rho(\mathbf{r})]$. Cependant, la fonctionnelle exacte n'est pas connue et on utilise en pratique des fonctionnelles de la densité approchées. La formalisme de Kohn-Sham propose des équations semblables à celles de Hartree-Fock, mais la partie non-locale de l'opérateur de Fock est remplacée par un potentiel local. Ces méthodes sont extrêmement populaires car elles donnent d'excellents résultats à faible coût lorsque le système peut être bien décrit par un seul déterminant de Slater.

Une approche tout à fait différente consiste à approcher les éléments de la matrice hamiltonienne en paramétrant ses éléments. S'ils sont paramétrés sur les

distances entre paires d'atomes, il s'agit de l'approximation des liaisons fortes (tight binding). Lorsque l'hamiltonien est paramétré, le calcul n'est plus dit *ab initio* mais semi-empirique. Il est également possible de paramétrer l'opérateur de Kohn-Sham au lieu de paramétrer l'hamiltonien en utilisant une approche de type liaisons-fortes. Dans ce cas, il s'agit de la méthode DFTB (Density Functional Tight Binding).

2.2 Vers le parallélisme massif

2.2.1 Origines des difficultés de parallélisation

Les méthodes auto-cohérentes de type Hartree-Fock ou Kohn-Sham se parallélisent bien en général, bien que les synchronisations dues à leur caractère itératif empêchent un scaling idéal. Le problème est exprimé dans la base des orbitales atomiques, et les matrices sont distribuées sur plusieurs nœuds. La résolution du problème est réalisée en utilisant une bibliothèques d'algèbre linaire dense distribuée telle que ScaLAPACK[27] pour diagonaliser l'hamiltonien. Dans ce cas, la diagonalisation de l'hamiltonien est l'étape limitante. Des algorithmes plus évolués utilisent la localité des orbitales atomiques pour tirer profit de la nature creuse des matrices et remplacer la diagonalisation de l'hamiltonien par un gradient conjugué. Ces algorithmes ne nécessitent que des produits de matrices creuses et sont en général beaucoup plus rapides que la diagonalisation.[28]

Toutes les méthodes post-Hartree-Fock nécessitent l'utilisation d'intégrales bi-électroniques $(ik|jl)$ exprimées dans la base des orbitales moléculaires (OM). Le calcul des intégrales $(pq|rs)$ dans la base des orbitales atomiques (OA) peut être réalisé assez rapidement car en exploitant la localité des fonctions de base le nombre d'intégrales non-nulles tend vers une croissance en $\mathcal{O}(N^2)$. Cependant, à cause de l'orthogonalité imposée par la base des OM celle-ci est beaucoup moins locale que la base des OA. Une étape inévitable est donc la *transformation à 4 indices* qui consiste à transformer les intégrales $(pq|rs)$ en $(ik|jl)$. Transformer une seule intégrale coûte $\mathcal{O}(N^4)$:

$$(ik|jl) = \sum_{pqrs} C_i^p C_k^q C_j^r C_l^s (pq|rs) \tag{2.19}$$

mais la transformation complète de toutes les intégrales peut être réalisée en $\mathcal{O}(N^5)$[29] :

$$\begin{aligned}
(iq|rs) &= \sum_p C_i^p (pq|rs) \\
(iq|js) &= \sum_r C_j^r (iq|rs) \quad \text{(intégrales semi-transformées)} \qquad (2.20) \\
(ik|js) &= \sum_q C_k^q (iq|js) \\
(ik|jl) &= \sum_s C_l^s (ik|js) \qquad (2.21)
\end{aligned}$$

Cette étape n'est pas triviale à paralléliser. La première passe de l'algorithme consiste à produire les intégrales semi-transformées $(iq|js)$ où chaque nœud va produire toutes les intégrales du sous-ensemble $(i \cdot |j\cdot)$. Chaque nœud aura donc besoin de toutes les intégrales $(pq|rs)$. Puis, dans la seconde passe où les intégrales $(ik|jl)$ sont produites, chaque nœud va calculer toutes les intégrales du sous-ensemble $(\cdot k|\cdot l)$ et aura donc besoin de toutes les intégrales semi-transformées. On voit bien qu'une

grande quantité de données doit être synchronisée entre les nœuds, mais aussi réordonnée pour permettre un accès rapide dans la deuxième passe. Ainsi, la parallélisation distribuée de cette étape est difficilement scalable.

Les méthodes perturbatives mono-référence ne nécessitent pas toutes les intégrales bi-électroniques, mais seulement celles qui impliquent deux orbitales occupées et deux orbitales virtuelles[30]. L'algorithme Saunders et van Lenthe[31] permet d'utiliser cet avantage pour accélérer la transformation à 4 indices, ce qui rend les méthodes perturbatives de type Møller-Plesset extrêmement rapides.

Les méthodes d'interaction de configurations (IC) sont celles qui ont la moins bonne efficacité parallèle car elles nécessitent de diagonaliser l'hamiltonien dans la base des déterminants. Cela impose d'avoir réalisé au préalable un transformation à 4 indices beaucoup plus coûteuse que dans les méthodes perturbatives car les intégrales impliquant 3 et 4 orbitales virtuelles sont nécessaires et ce sont en général les plus nombreuses. Étant donné la taille des matrices à diagonaliser (plusieurs millions voir centaines de millions de fonctions de bases), la recherche des états propres est réalisée en utilisant l'algorithme de Davidson[32] qui est une méthode itérative, et qui impose donc des barrières de synchronisation. Deux principales approches existent. La première, dite *integral driven*, consiste à parcourir la liste des intégrales moléculaires non nulles et à regarder quelle contribution peut être calculée avec chacune des intégrales. L'avantage de cette approche est que le nombre d'intégrales est en $\mathcal{O}(N^4)$ tandis que la taille de l'espace de Hilbert est en $\mathcal{O}(N!)$. Ainsi, aucun travail n'est fait inutilement puisque chacune des intégrales est associée à un travail utile pour la recherche des états propres. L'inconvénient de ce type de méthodes est qu'elles nécessitent un accès direct en écriture aux fonctions de base ce qui implique d'avoir une fonction efficace pour associer facilement un déterminant à son indice dans l'espace de Hilbert. Une fonction remplissant ce rôle est une *fonction de hachage* (hash function en anglais). Souvent, on choisira de résoudre des problèmes dans des sous-espaces complets (toutes les simples et doubles excitations, espace CAS, *etc*) pour avoir une fonction de hachage simple et efficace, ce qui engendrera des tailles de vecteurs gigantesques, et ceux-ci devront être stockés sur disque. Pour ce type d'algorithme, si les vecteurs sont distribués tous les nœuds doivent avoir accès à (presque) toutes les intégrales, et si les intégrales sont distribuées tous les nœuds doivent avoir accès à l'intégralité des vecteurs. Quoi qu'il en soit, le goulot d'étranglement reste inévitablement les communications collectives ou l'accès au système de fichiers.

Les algorithmes de type *determinant driven* parcourent les paires de déterminants une à une et vont chercher l'intégrale correspondante. Il faut donc avoir une fonction de hachage efficace pour les intégrales qui seront accédées dans un ordre aléatoire. Puisque seuls les déterminants ne différant que par une mono- ou une di-excitation interagissent, on voit bien que la grande majorité des paires de déterminants ont des éléments de matrice nuls. Ainsi, pour avoir un algorithme efficace il est nécessaire d'organiser les déterminants afin de pouvoir déterminer *a priori* quels petits groupes de déterminants sont connectés entre eux par l'hamiltonien. En général, le temps de calcul lié à un élément de matrice est considérablement plus long que pour les algorithmes *integral driven* mais ces algorithmes permettent de travailler avec des sous-espaces de tailles beaucoup plus réduites. Cela permet de s'affranchir de la nécessité d'accéder au système de fichiers, et ces algorithmes semblent être de meilleurs candidats pour aller vers le parallélisme massif.

2.2.2 Aspects logiciels

Dans la communauté du calcul haute performance (HPC), le standard est l'utilisation de langages impératifs tels que Fortran, C ou C++, couplés à l'utilisation des API MPI (Message Passing Interface)[33] pour le parallélisme distribué, et OpenMP[34] pour le parallélisme à mémoire partagée. Ces choix se sont révélés être très efficaces jusqu'à aujourd'hui dans de nombreux domaines scientifiques et industriels, cependant on peut se demander si il n'existe pas des approches alternatives qui permettraient, en se plaçant dans un paradigme différent, de proposer des algorithmes originaux et adaptés au parallélisme à grande échelle.

Il est vrai que les compilateurs C ou Fortran permettent aujourd'hui de produire des sous-programmes qui approchent la performance crête des processeurs. Indiscutablement, ce sont ces langages qu'il faut utiliser pour les régions du code où la performance est critique. Cependant, malgré les améliorations qui ont été apportées à ces langages au cours des dernières années, il existe d'autres langages peu utilisés dans la communauté du calcul qui ont des propriétés extrêmement intéressantes pour le développement de nouveaux algorithmes parallèles. Par exemple, le langage Erlang a été conçu dans le contexte de la téléphonie mobile pour permettre de développer des applications massivement distribuées sur des réseaux peu fiables, tolérantes aux pannes et communiquant par échanges de messages. Le langage Rust, quant à lui, a une syntaxe proche du C mais garantit une gestion sûre de la mémoire même dans un contexte multithreadé et avec des performances proches de celles du C. Le langage OCaml, par son typage fort aide le développeur à intercepter un très grand nombre d'erreurs à la compilation, et par son aspect fonctionnel garantit également un certain niveau de sureté en ce qui concerne les accès à la mémoire. Il me semble nécessaire de contribuer à l'introduction de ces langages dans notre communauté en montrant ce qu'ils permettent de nous apporter pour le développement et la maintenance des programmes, tout en conservant les zones de calcul intensif en C ou Fortran.

L'API MPI a été conçue au départ dans un contexte de parallélisme synchrone, et certaines fonctions permettant la programmation asynchrone y ont été introduites récemment. Cette API permet de standardiser la communication par échange de messages entre des processus distribués, et implémente des fonctions de synchronisation, de communications point-à-point ou de communications collectives. Dans les supercalculateurs, l'utilisation de bibliothèques spécialisées permet d'avoir accès à toute la performance du réseau à faible latence, et cette API est s'est donc imposée comme la norme pour le calcul distribué. Le contexte synchrone des applications développées avec MPI permet une grande simplicité de développement, au prix de certaines contraintes :

- Les ressources de calcul sont supposées fixes, ce qui empêche une gestion élastique du nombre du processeurs réservés
- Le matériel est supposé fiable : si un des processus meurt, le calcul est interrompu
- Le paradigme utilisé est SPMD (Single Program, Multiple Data), et programmer un système de type client/serveur n'est pas naturel

Toutes ces contraintes sont parfaitement justifiées dans le cadre d'algorithmes déterministes et synchrones, mais ne sont pas justifiées pour toutes les applications, comme nous le verrons pas la suite pour le code QMC=Chem.

2.2.3 Aspects matériels

Les plus gros supercalculateurs actuels ont de l'ordre du million de cœurs. Sur de telles machines, la fréquence des processeurs est devenue variable pour des

raisons d'économie d'énergie, et l'utilisation d'accélérateurs va bientôt devenir incontournable. Dans un tel contexte, les modèles synchrones correspondent de moins en moins bien au matériel. Si l'ont veut exécuter un calcul sur une telle machine, on ne peut plus se reposer sur la performance du réseau ni espérer que toutes les ressources resteront fonctionnelles pendant le temps nécessaire pour que le calcul se termine. Il est absolument nécessaire d'introduire la prise en compte des pannes dans le développement du logiciel, et de considérer que les latences des communications ne sont pas uniformes et extrêmement chères. Ces diverses contraintes nous ramènent au calcul distribué sur grille où le réseau est le réseau internet, donc peu fiable, peu performant et inhomogène, et où des processeurs de types différents peuvent être alloués pour le même calcul. On est donc amené à penser que les programmes qui passeront à l'échelle de l'exascale ne sont pas forcément les programmes de la communauté du HPC, mais plutôt ceux qui viennent de la communauté des grilles/cloud où ces problèmes sont déjà bien connus.

2.3 Résumé

Mes travaux de recherche portent sur le développement de nouvelles méthodes pour traiter le problème de la corrélation électronique en chimie quantique. Ces développements sont guidés d'une part par l'évolution des architectures des supercalculateurs, et d'autre part par l'évolution des langages et des paradigmes de programmation.

Chapitre 3

Travail de recherche au LCPQ

3.1 Interaction de configurations sélectionnées

L'expérience montre que parmi tous les déterminants de l'espace de Hilbert, seulement une toute petite fraction contribue à l'énergie et aux propriétés des états de plus basse énergie. De plus, le poids d'un déterminant dans la fonction d'onde n'est pas directement relié à son degré d'excitation. Par exemple, certains déterminants quadruplement excités par rapport au déterminant de Hartree-Fock peuvent être plus importants que d'autres déterminants doublement excités (Figure 3.1). La sélection de déterminants qui contribuent majoritairement à l'énergie est une idée naturelle qui a été introduite il y a longtemps et qui a régulièrement été redécouverte.[32, 35, 36, 37]. Nous utilisons l'algorithme *Configuration Interaction using a Perturbative Selection made Iteratively* (CIPSI) où une estimation en perturbations de la contribution d'un déterminant à l'énergie est le critère qui va permettre de l'inclure ou non dans la fonction d'onde. Contrairement aux méthodes de type CISD ou CAS, dans lesquelles l'espace de Hilbert est tronqué *a priori*, l'algorithme CIPSI travaille dans l'espace de Hilbert complet et permet de choisir les déterminants au fur et à mesure que le calcul se raffine. Si on laisse tourner indéfiniment l'algorithme, il converge vers le Full-CI. Il ne s'agit donc pas d'une méthode, mais plutôt d'un algorithme de résolution du Full-CI que l'on peut interrompre avant la convergence complète.

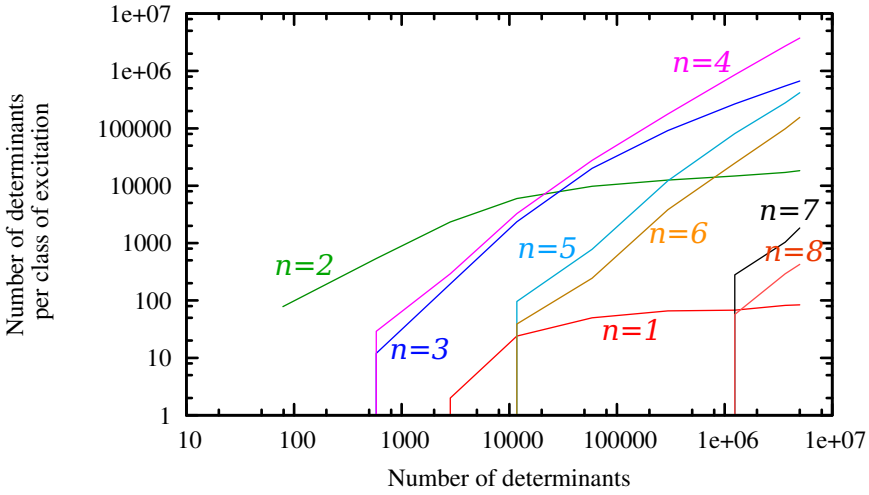


FIGURE 3.1 – N_2 dans la base cc-pVTZ (R_{N-N}=1.0977 Å). Variation du nombre de déterminants n-excités par rapport au déterminant de Hartree-Fock dans l'expansion CIPSI en fonction du nombre total de déterminants sélectionnés.

3.1.1 Algorithme CIPSI

A l'itération n :

1. On dispose d'une fonction d'onde définissant un *espace de référence* $S^n = \{|D_0\rangle, \ldots\}$. On diagonalise l'hamiltonien $\widehat{H}$ dans cet espace et on obtient l'énergie E_0^n de l'état fondamental et le vecteur propre correspondant :

$$|\Psi_0^n\rangle = \sum_{i \in S^n} c_i^n |D_i\rangle \tag{3.1}$$

2. On établit la liste sans répétition de tous les déterminants $|D_k\rangle$ qui sont connectés à $|\Psi_0^n\rangle$ par $\widehat{H}$ et qui n'appartiennent pas à l'espace S^n.
3. Pour chacun des déterminants connectés, on estime avec la théorie des perturbations la modification de l'énergie qui serait apportée par l'inclusion du déterminant dans la fonction d'onde :

$$\delta e(|D_k\rangle) = \frac{\langle \Psi_0^n | \widehat{H} | D_k \rangle^2}{E_0^n - H_{kk}} \tag{3.2}$$

4. On ajoute à l'espace de référence pour l'itération suivante les déterminants $\{|D_{k^*}\rangle\}$ correspondant aux plus grandes contributions $|\delta e(|D_{k^*}\rangle)|$:

$$S^{n+1} = S^n \cup \{|D_{k^*}\rangle\} \tag{3.3}$$

L'algorithme peut être interrompu après M itérations en utilisant plusieurs critères comme par exemple le nombre de déterminants inclus dans la fonction d'onde $|\Psi_0^M\rangle$. On peut ensuite raffiner le résultat en calculant une correction perturbative à l'énergie comme :

$$E_{\mathrm{PT2}} = \sum_{|D_k\rangle \in \Omega} \frac{\langle \Psi_0^M | \widehat{H} | D_k \rangle^2}{E_0^M - H_{kk}} \tag{3.4}$$

où Ω est l'ensemble des déterminants qui sont connectés à $|\Psi_0^M\rangle$ par $\widehat{H}$ et qui n'appartiennent pas à l'espace S^M. En pratique, cette dernière étape permet d'obtenir l'énergie Full-CI avec la précision chimique si la fonction d'onde variationnelle est suffisamment grande.

3.1.2 Applications à la molécule d'eau

Pour montrer l'efficacité de la méthode, nous avons réalisé des calculs CIPSI pour des systèmes de différentes tailles. Dans un premier exemple, nous reproduisons les résultats DMRG de Chan *et al*[38] à la géométrie R_{OH}=1 Å et θ_{OH}=104.5° dans la base *Roos Augmented Double Zeta ANO* comportant 41 orbitales[39]. L'espace de Hilbert est composé d'environ 5.6 10^{11} déterminants, et la convergence de l'énergie en fonction du nombre de déterminants sélectionnés est présentée sur la figure 3.2.

Les deux courbes supérieures sont les courbes de l'énergie variationnelle, et les deux courbes inférieures sont les courbes de l'énergie avec la correction perturbative. Les deux courbes en rouge sont calculées en utilisant les orbitales moléculaires de Hartree-Fock (orbitales canoniques) et les courbes en bleu sont obtenues en utilisant les orbitales naturelles calculées à partir du dernier point de la courbe rouge. On peut remarquer que les quatre courbes convergent vers la valeur de -76.31471(1) de Chan *et al*, et la convergence des courbes utilisant la correction perturbative est beaucoup plus rapide que la convergence de la courbe variationnelle. L'utilisation d'orbitales

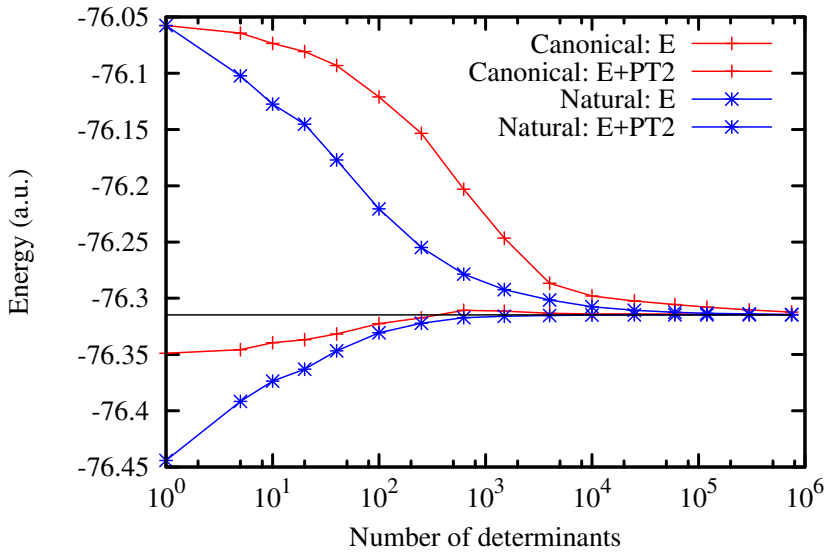


FIGURE 3.2 – Convergence de l'énergie en fonction du nombre de déterminants sélectionnés, en utilisant des orbitales canoniques ou naturelles.

naturelles accélère la convergence de la courbe variationnelle, mais ralentit la convergence de la courbe utilisant la correction perturbative. Pour obtenir la valeur Full-CI, on a donc besoin d'environ un million de déterminants au niveau variationnel et 1 000 déterminants si l'on utilise la correction perturbative, ce qui est négligeable devant la taille de l'espace de Hilbert (< 0.0002 %).

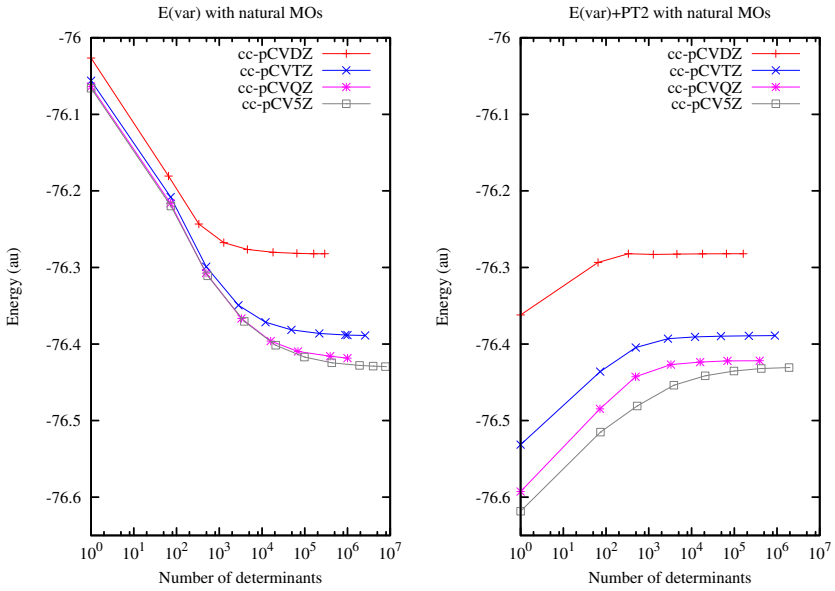


FIGURE 3.3 – Convergence de l'énergie variationnelle (à gauche) et avec la correction perturbative (à droite) en fonction du nombre de déterminants sélectionnés.

Pour illustrer la possibilité de réaliser des calculs dans des bases beaucoup plus grandes, nous avons utilisé la série des bases cc-pCVnZ[40] avec $2 \leq n \leq 5$. La géométrie choisie est désormais la géométrie d'équilibre expérimentale R_{OH}=0.9572 Å

et θ_{OH}=104.52°. Le nombre de fonctions de base est 28, 71, 174 et 255 pour les bases cc-pCVDZ, cc-pCVTZ, cc-pCVQZ et cc-pCV5Z. Le nombre de déterminants dans l'espace de Hilbert est de l'ordre de 10^{10}, $1.7\,10^{14}$, $1.6\,10^{18}$ et $7.5\,10^{19}$. Dans la partie gauche de la figure 3.3 est représentée la convergence de l'énergie variationnelle et dans la partie droite la convergence de l'énergie incluant la correction perturbative. Ces figures montrent que la convergence vers l'énergie Full-CI peut toujours être atteinte, même avec de grandes bases.

3.1.3 Généralisation à l'ensemble G1

Pour montrer que l'algorithme CIPSI est utilisable pour estimer les énergies Full-CI, nous avons réalisé des calculs d'énergies d'atomisation dans les bases cc-pVDZ et cc-pVTZ pour molécules de l'ensemble G1 de Pople *et al*[41]. Cet ensemble est composé de 55 molécules composées de 11 atomes (Be,C,Cl,F,H,Li,N,Na,O,P,S).

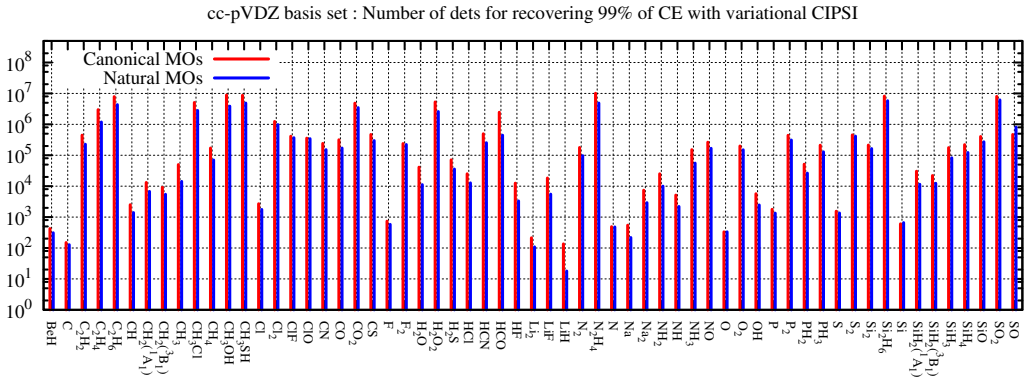


FIGURE 3.4 – Nombre de déterminants nécessaires pour obtenir 99 % de l'énergie de corrélation au niveau CIPSI/cc-pVDZ variationnel.

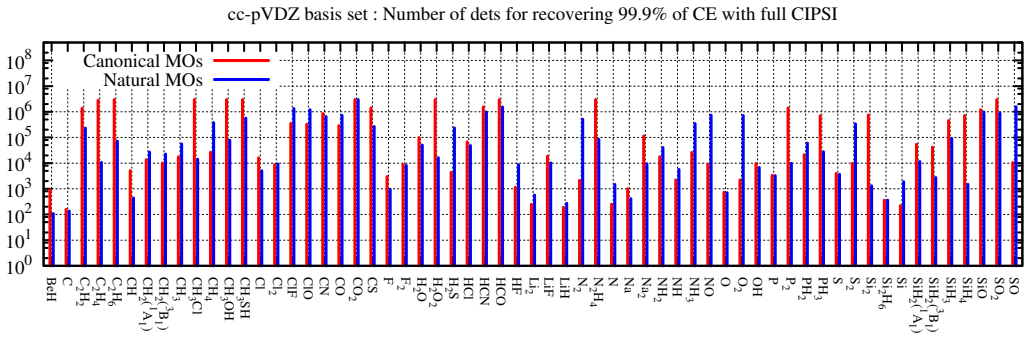


FIGURE 3.5 – Nombre de déterminants nécessaires pour obtenir 99.9 % de l'énergie de corrélation au niveau CIPSI/cc-pVDZ avec la correction perturbative.

Pour chacun des systèmes une convergence quasi-Full-CI a été obtenue ($|E_{PT2}| <$ 0.001 ua). La figure 3.4 montre le nombre de déterminants nécessaires pour obtenir 99 % de l'énergie de corrélation dans la base cc-pVDZ. Ce nombre est très variable en fonction du nombre d'électrons et du caractère multi-configurationnel du système. L'utilisation d'orbitales naturelles est telle que le nombre de déterminants nécessaires est presque toujours plus faible qu'avec des orbitales canoniques. La figure 3.5 est

semblable à la première figure et montre le nombre de déterminants nécessaires pour obtenir 99.9 % de l'énergie de corrélation lorsque la correction perturbative est appliquée. Il est remarquable qu'une telle précision puisse être obtenue avec un nombre de déterminants toujours inférieur à 10^7.

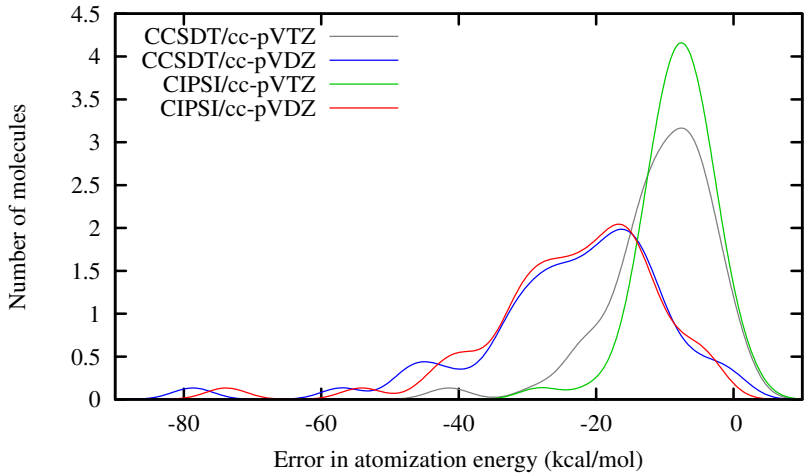


FIGURE 3.6 – Distribution des erreurs sur les énergies d'atomisation de l'ensemble G1 calculées au niveau CIPSI et CCSD(T) dans les bases cc-pVDZ et cc-pVTZ.

Enfin, les résultats sont comparés avec les calculs CCSD(T) dans la même base. La figure 3.6 montre la distribution des erreurs sur les énergies d'atomisation calculées au niveau CIPSI et CCSD(T) dans les bases cc-pVDZ et cc-pVTZ. Dans la base cc-pVDZ, les courbes CIPSI et CCSD(T) sont semblables, montrant que les calculs CCSD(T) ont atteint une qualité quasi-Full-CI. Dans la base cc-pVTZ, les deux courbes restent similaires, mais quelques différences apparaissent montrant que les calculs CIPSI sont légèrement plus précis. Cela provient du fait que le CCSD(T) est une méthode mono-référence, et que certains systèmes de l'ensemble ont un caractère multi-configurationnel qui est correctement traité par l'algorithme CIPSI.

3.1.4 Calcul stochastique de l'énergie PT2

Dans la théorie des perturbations multi-références Epstein-Nesbet, l'hamiltonien de référence est

$$H_0 = E_0|\Psi\rangle\langle\Psi| + \sum_{\alpha\in\mathcal{C}} H_{\alpha\alpha}|\alpha\rangle\langle\alpha|, \tag{3.5}$$

où la fonction d'onde

$$|\Psi\rangle = \sum_{I\in\mathcal{D}} c_I|I\rangle \tag{3.6}$$

est exprimée dans l'espace de référence $\mathcal{D}$ et E_0 est l'énergie correspondante. La somme dans l'Eq. (3.5) court sur les déterminants de l'ensemble $\mathcal{C}$ défini comme

$$\mathcal{C} = \{|\alpha\rangle \mid (|\alpha\rangle \notin \mathcal{D}) \wedge (\exists I \in \mathcal{D})(H_{\alpha I} \neq 0)\}. \tag{3.7}$$

La correction de second ordre à l'énergie s'écrit comme

$$E_2 = \sum_{\alpha\in\mathcal{C}} \frac{|\langle\alpha|H|\Psi\rangle|^2}{\Delta E_\alpha} \tag{3.8}$$

avec $\Delta E_\alpha = E_0 - H_{\alpha\alpha}$. On voit que le nombre de termes de la somme croît comme $N_{\text{elec}}^2 \times (N_{\text{OM}} - N_{\text{elec}})^2 \times N_{\text{det}}$. Un tel coût de calcul devient rapidement prohibitif, et il est nécessaire de recourir à des approximations. Puisque chaque terme de la somme est négatif, si l'on néglige certains termes on va inévitablement sous-estimer E_2 et introduire un biais. Le but de l'algorithme proposé ici est de transformer ce biais incontrôlé en erreur statistique par un algorithme de Monte Carlo.

Partitionnement de l'espace

La première étape de la méthode consiste à partitionner l'ensemble des déterminants $|\alpha\rangle$ en sous-ensembles disjoints $\mathcal{A}_I$ associés aux déterminants de référence $|I\rangle$:

$$\begin{aligned} \mathcal{C} &= \bigcup_{I=1}^{N_{\text{det}}} \mathcal{A}_I, \\ &(\forall I \neq J)(\mathcal{A}_I \cap \mathcal{A}_J = \emptyset) \end{aligned} \tag{3.9}$$

Pour définir les $\mathcal{A}_I$, on trie les déterminants par ordre décroissant de leur poids dans la fonction d'onde $c_I^2 / \sum_J c_J^2$. $\mathcal{A}_1$ est défini comme l'ensemble de tous les déterminants de $\mathcal{C}$ qui sont connectés par $\widehat{H}$ au premier déterminant. Puis, chaque $\mathcal{A}_I$ est défini récursivement comme l'ensemble des déterminants de $\mathcal{C}$ connectés par $\widehat{H}$ au déterminant $|I\rangle$, mais tel qu'ils n'appartiennent à aucun $\mathcal{A}_{J<I}$ défini précédemment :

$$\mathcal{A}_I = \left\{ |\alpha\rangle \mid (|\alpha\rangle \in \mathcal{C}) \wedge (H_{\alpha I} \neq 0) \wedge (\forall J < I)(|\alpha\rangle \notin \mathcal{A}_J) \right\}. \tag{3.10}$$

On peut alors ré-écrire

$$E_2 = \sum_{I=1}^{N_{\text{det}}} e_I \tag{3.11}$$

où les e_I sont définis comme

$$e_I = \sum_{\alpha \in \mathcal{A}_I} \frac{\left| \langle \alpha | \hat{H} | \Psi \rangle \right|^2}{\Delta E_\alpha}. \tag{3.12}$$

Une conséquence importante de cette partition est qu'aucun des déterminants α appartenant à $\mathcal{A}_I$ n'est connecté aux déterminants $|J\rangle$ précédant $|I\rangle$. On peut donc ré-écrire les e_I comme

$$e_I = \sum_{\alpha \in \mathcal{A}_I} \frac{\left| \langle \alpha | \hat{H} | \Psi_I \rangle \right|^2}{\Delta E_\alpha}. \tag{3.13}$$

où la fonction d'onde tronquée Ψ_I est donnée par

$$\Psi_I = \sum_{J=I}^{N_{\text{dets}}} c_J |J\rangle. \tag{3.14}$$

Estimateur Monte Carlo

Pour obtenir une expression utilisable dans le cadre de simulations Monte Carlo, E_2 est ré-écrite comme

$$E_2 = \sum_{I=1}^{N_{\text{det}}} p_I \left(\frac{e_I}{p_I}\right), \tag{3.15}$$

où p_I est une distribution de probabilités arbitraire dont le meilleur choix est donné par la condition de zéro variance

$$p_I^* = \frac{e_I}{E_2}. \tag{3.16}$$

Pour obtenir une simulation Monte Carlo efficace, il est nécessaire de trouver une bonne approximation de p_I^*. En considérant que les ΔE_α sont constants, puisque les termes dominants dans l'Eq. (3.13) sont les éléments diagonaux nous avons choisi de prendre

$$p_I = \frac{c_I^2}{\sum_J c_J^2}. \tag{3.17}$$

Les contributions dues aux quelques déterminants de poids fort sont les plus importantes. Nous avons donc choisi de traiter ces contribution de façon déterministe, et de réaliser le calcul stochastique seulement sur les contributions restantes

$$E_2 = \sum_{I=1}^{n_0} e_I + \left\langle \frac{e_I}{p_I} \right\rangle_{p_I} = E_{2,D} + E_{2,S} \tag{3.18}$$

Les déterminants de la partie déterministe sont notés $\{I\}_D$ et les déterminants de la partie stochastique sont notés $\{I\}_S$.

Le tirage Monte Carlo est réalisé de façon standard, en utilisant la fonction de distribution cumulative de la distribution de probabilités

$$R(I) = \sum_{J=n_0+1}^{I} p_J, \tag{3.19}$$

et en tirant à chaque pas Monte Carlo un nombre aléatoire $u \in [0,1]_{\mathbb{R}}$ pour choisir le déterminant qui vérifie

$$R(I) \leq u \leq R(I+1). \tag{3.20}$$

Pour rendre l'échantillonnage plus efficace et donc réduire les fluctuations, on utilise un échantillonnage corrélé. Pour cela, l'intervalle $[0,1]_{\mathbb{R}}$ est divisé en M intervalles de longueur $1/M$, et M nombre aléatoires corrélés sont construits à partir d'un seul nombre u

$$(\forall k \in [1,M]_{\mathbb{N}}) \left(u_k = \frac{k-1+u}{M}\right) \tag{3.21}$$

À l'itération n de la simulation Monte Carlo, $N^{(n)} = nM$ déterminant ont été tirés dans $\{I\}_S$. Un nombre aléatoire u est tiré, et M nouveaux déterminants sont obtenus. La partie stochastique est réactualisée comme

$$E_{2,S}^{(n+1)} = \frac{N^{(n)} E_{2,S}^{(n)} + \sum_{k=1}^{M} \frac{e_{I_k}}{p_{I_k}}}{N^{(n+1)}}. \tag{3.22}$$

et la valeur de E_2 à l'itération n est

$$E_2^{(n)} = E_{2,D} + E_{2,S}^{(n)} \tag{3.23}$$

et $E_2^{(n)}$ converge vers E_2 avec une erreur en $\sim \frac{1}{\sqrt{nM}}$.

Ajustement dynamique de la partie déterministe

Dans cet algorithme, la partie la plus coûteuse est le calcul de e_I. Ainsi, lorsqu'une valeur de e_I est calculée le résultat est stocké, si bien que lorsque le déterminant $|I\rangle$ est tiré à nouveau la valeur de e_I peut être reprise en mémoire. Chaque e_I n'est donc calculé qu'une seule fois. Si l'on néglige le temps passé dans les autre parties de l'algorithme, le temps de calcul est borné par le temps de calcul exact déterministe de E_2, et on comprend bien que l'erreur en fonction du temps de calcul va diminuer beaucoup plus rapidement que $1/\sqrt{t}$.

Au cours du calcul, certains e_I ont été calculés contrairement à d'autres. On peut à tout moment choisir de faire passer certains déterminants dont e_I a été calculé dans la partie déterministe et de mettre à jour la moyenne courante et l'erreur. Ainsi, puisque la valeur absolue de la contribution stochastique diminue, l'erreur associée diminue également. Pour implémenter cette idée, on décompose E_2 comme une somme de M contributions

$$E_2 = \sum_{J=1}^{n_0} e_J + \sum_{k=1}^{M} \sum_{I \in \mathcal{I}_k} e_I. \tag{3.24}$$

où $\mathcal{I}_k$ est défini comme

$$\mathcal{I}_k = \left\{ |I\rangle \,\middle|\, \frac{k-1}{M} \leq R(I) \leq \frac{k}{M} \right\} \tag{3.25}$$

Lorsque tous les déterminants de $\mathcal{I}_k$ ont été tirés, on supprime ces déterminants de l'espace des $\{I\}_S$ vers l'espace des $\{I\}_D$ et la contribution $\sum_{I \in \mathcal{I}_k} e_I$ est ajoutée à la partie déterministe.

Quelques résultats

Pour illustrer l'efficacité de l'algorithme stochastique, nous avons préparé plusieurs fonctions d'ondes pour la molécule Cr_2 (cœur gelé). La première, dans la base cc-pVDZ, est composée de 5 millions de déterminants. Cela correspond à corréler 28 électrons en utilisant 76 OMs. Le calcul déterministe a été fait sur 150 nœuds (2400 cœurs) de Curie (TGCC/CEA/GENCI) en 2h40. Le calcul stochastique a été réalisé sur 50 nœuds (800 cœurs), et la convergence de l'énergie et de l'erreur en fonction du temps de calcul sont représentées sur la figure 3.7. D'après la courbe de l'erreur on peut constater qu'une précision de 0.01 mE_h peut être atteinte en 12.5 minutes, ce qui représente une réduction du temps CPU d'un facteur 40 par rapport au calcul exact.

La seconde fonction d'onde comporte plus de 28 millions de déterminants dans la base cc-pVQZ, ce qui correspond à corréler 28 électrons en utilisant 198 OMs. Pour ce calcul, on estime le temps de calcul déterministe à 24 heures sur 250 nœuds (4000 cœurs). La convergence de l'énergie et de l'erreur en fonction du temps de calcul sont représentées sur la figure 3.8, sur 250 nœuds également.

Grâce à l'algorithme stochastique que nous avons proposé, le calcul de la contribution perturbative à l'énergie est toujours faisable pour les fonctions d'ondes

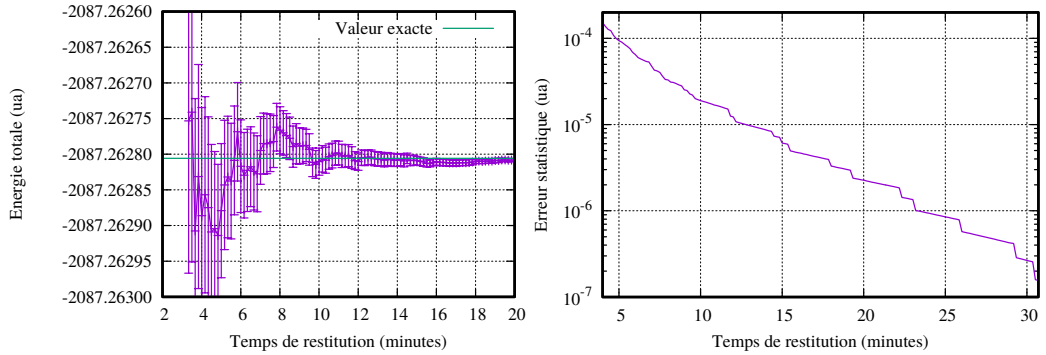


FIGURE 3.7 – Cr_2, cc-pVDZ, 5×10^6 déterminants. Convergence de l'énergie $E_0 + E_{PT2}$ (à gauche) et de l'erreur statistique (à droite) en fonction du temps de restitution.

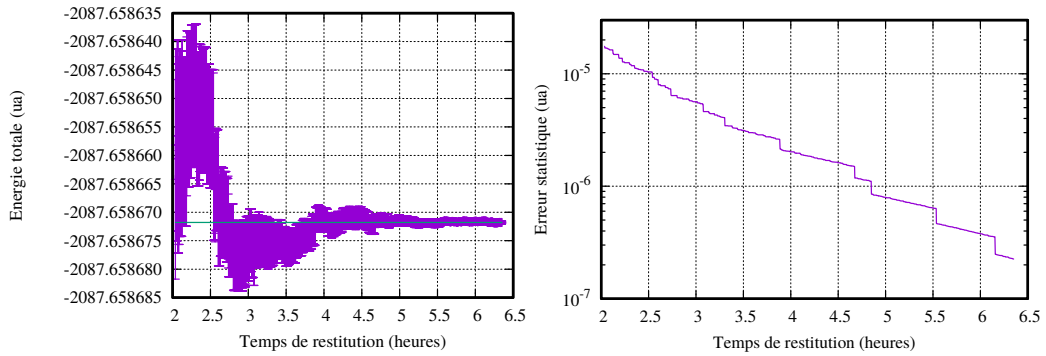


FIGURE 3.8 – Cr_2, cc-pVQZ, 28.1×10^6 déterminants. Convergence de l'énergie $E_0 + E_{PT2}$ (à gauche) et de l'erreur statistique (à droite) en fonction du temps de restitution.

variationnelles que nous sommes capables de produire avec l'algorithme CIPSI, sans aucune approximation mais avec une incertitude contrôlée.

3.1.5 Coupled-Cluster multi-références

Bien que la proportion de l'espace de Hilbert nécessaire pour estimer l'énergie Full-CI soit très faible, le nombre de déterminants demeure néanmoins trop important pour pouvoir réaliser des calculs de grande précision pour des systèmes possédant plus d'une centaine d'électrons. Nous avons donc permis à notre programme de travailler dans un espace de déterminants sélectionnés *a priori*, ce qui nous donne immédiatement accès à des variantes sélectionnées de toutes les méthodes post-Hartree-Fock. Ainsi, la méthode MR-CI totalement décontractée est implémentée via l'algorithme CIPSI ce qui rend possible l'utilisation du programme pour des systèmes de plus grandes tailles.

Cependant, les méthodes d'IC dans des espaces tronqués par degré d'excitation ne sont pas extensives. Pour remédier à ce problème, il est nécessaire d'appliquer des corrections au MR-CI introduisant les déterminants triplement et quadruplement excités par rapport à la référence, et avec des coefficients ajustés tels qu'ils permettent d'exprimer la fonction d'onde comme un produit. La méthode Coupled Cluster permet de retrouver exactement la propriété d'extensivité si la référence est elle-même extensive.

Nous verrons d'abord qu'il est possible d'exprimer la méthode Coupled Cluster comme une série de calculs d'IC avec un hamiltonien modifié (habillé).[42] Cela

nous permettra d'implémenter sans difficulté la méthode Coupled Cluster en utilisant l'algorithme CIPSI pour réduire considérablement le coût du calcul. Puis, nous introduirons un formalisme multi-références pour le Coupled Cluster et nous montrerons son efficacité.[43]

Habillage Coupled Cluster mono-référence

En Coupled Cluster mono-référence (CCSD), la fonction d'onde est exprimée comme

$$|\Psi_{\mathrm{CC}}\rangle = \widehat{\Omega}|D_0\rangle = \exp(\widehat{T})|D_0\rangle \tag{3.26}$$

où l'opérateur d'excitation $\widehat{T}$ appliqué sur le déterminant de référence $|D_0\rangle$ est limité aux excitations simples et doubles :

$$\begin{aligned} \exp(\widehat{T}) &= \exp(\widehat{T}_1 + \widehat{T}_2) \\ &= 1 + (\widehat{T}_1 + \widehat{T}_2) + \frac{1}{2}(\widehat{T}_1 + \widehat{T}_2)^2 + \dots \end{aligned} \tag{3.27}$$

La fonction d'onde s'exprime donc sur la base *complète* des déterminants

$$\begin{aligned} |\Psi_{\mathrm{CC}}\rangle &= |D_0\rangle + \sum_i d_{1i}\widehat{T}_{1i}|D_0\rangle + \sum_j d_{2j}\widehat{T}_{2j}|D_0\rangle \\ &\quad + \frac{1}{2}\sum_i d_{1i}^2\widehat{T}_{1i}^2|D_0\rangle + \frac{1}{2}\sum_j d_{2j}^2\widehat{T}_{2j}^2|D_0\rangle + \sum_i\sum_j d_{1i}d_{2j}\widehat{T}_{1i}\widehat{T}_{2j}|D_0\rangle + \dots \\ &= |D_0\rangle + \sum_{I\in\mathrm{FCI}} c_I|I\rangle \end{aligned} \tag{3.28}$$

Si l'énergie est calculée par projection sur la référence

$$E_{\mathrm{CC}} = \frac{\langle D_0|\widehat{H}|\Psi_{\mathrm{CC}}\rangle}{\langle D_0|\Psi_{\mathrm{CC}}\rangle}$$

seules les simples et doubles excitations contribuent à l'énergie car les autres déterminants ne sont pas connectés par $\widehat{H}$ au déterminant de référence. Il suffit donc seulement d'obtenir les amplitudes correctes d_{1i} et d_{2j} pour avoir l'énergie.

Pour exprimer le problème comme une IC, on écrit les équations aux valeurs propres projetées sur la référence :

$$\begin{aligned} \widehat{H}e^{\widehat{T}}|D_0\rangle &= Ee^{\widehat{T}}|D_0\rangle \\ \langle i|\widehat{H}e^{\widehat{T}}|D_0\rangle &= d_iE \end{aligned} \tag{3.29}$$

$$\begin{aligned} \langle i|\widehat{H}|D_0\rangle + \sum_j d_j\langle i|\widehat{H}|j\rangle + \sum_\alpha d_\alpha\langle i|\widehat{H}|\alpha\rangle &= d_iE \\ H_{i0} + d_i(H_{ii} - E) + \sum_{j\neq i} d_jH_{ij} + \sum_\alpha d_\alpha H_{i\alpha} &= 0 \end{aligned} \tag{3.30}$$

où $|i\rangle$ et $|j\rangle$ sont les déterminants simplement ou doublement excités, et les $|\alpha\rangle$ sont les déterminants triplement ou quadruplement excités. La paramétrisation exponentielle est telle que les amplitudes d_α sont calculés à partir des amplitudes de toutes les simples et doubles excitations qui peuvent être appliquées à la référence pour

engendrer $|\alpha\rangle$:

$$d_\alpha = \sum_{(k,l)} d_k d_l \qquad \{(k,l) \mid \widehat{T}_\alpha = \widehat{T}_k \widehat{T}_l\}. \tag{3.31}$$

On démarre l'algorithme avec un calcul CISD qui définit les amplitudes de départ. Puis on estime la contribution des triples et quadruples excitations en utilisant l'Eq. (3.31). Cette contribution ne dépend que du déterminant $|i\rangle$, donc on peut définir une matrice Δ :

$$\Delta_{ij} = \begin{cases} \sum_\alpha d_\alpha H_{i\alpha} & \text{si } j = 0 \\ 0 & \text{sinon} \end{cases} \tag{3.32}$$

pour générer une matrice hamiltonienne habillée $\tilde{H} = H + \Delta$. L'équation aux valeurs propres devient donc celle d'un CISD avec un hamiltonien modifié

$$\tilde{H}_{i0} + d_i(\tilde{H}_{ii} - E) + \sum_{j \neq i} d_j \tilde{H}_{ij} = 0 \tag{3.33}$$

on résout ce système, et les coefficients optimaux nous permettent de définir les amplitudes de départ pour l'itération suivante. Lorsque les d_i deviennent stationnaires, l'algorithme a convergé.

Coupled Cluster multi-références

Dans ce paragraphe, on notera $|I\rangle$ les déterminants de l'espace de référence, $|i\rangle$ les déterminants simplement ou doublement occupés par rapport à au moins un des $|I\rangle$ et qui ne sont pas dans la référence, et $|\alpha\rangle$ les déterminants qui sont simplement ou doublement occupés par rapport à au moins un des $|i\rangle$ et qui ne sont pas dans $\{|I\rangle\} \cup \{|i\rangle\}$.

De la même façon que dans le cas mono-référence, on démarre avec un calcul MR-CISD où la référence $|\Psi_0\rangle$ est un CAS si l'on veut avoir la propriété d'extensivité. La fonction d'onde s'écrit

$$\Psi_{\text{MR-CISD}} = \sum_I c_I |I\rangle + \sum_i c_i |i\rangle \tag{3.34}$$

On peut utiliser l'expression de Jeziorski et Monkhorst[44] où l'opérateur d'onde $\widehat{\Omega}$ est exprimé comme une combinaison linéaire d'opérateurs d'ondes appliqués sur chacun des déterminants de la référence.

$$\widehat{\Omega} = \sum_I \widehat{\Omega}_I |I\rangle\langle I| \tag{3.35}$$

Comme dans le cas mono-référence, chacun des opérateurs $\widehat{\Omega}_I$ a une forme exponentielle et est tronqué aux doubles excitations par rapport à $|I\rangle$. En utilisant cette forme d'opérateur d'onde, il est commode de réécrire la fonction d'onde sous la forme

$$|\Psi\rangle = \sum_I c_I \left(|I\rangle + \sum_i d_{Ii} |i\rangle \right) \tag{3.36}$$

où les d_{Ii} sont les amplitudes des excitations qui transforment $|I\rangle$ en $|i\rangle$ et telles que $c_i = \sum_I d_{Ii}$.

Il se pose donc le problème de la définition des d_{Ii} car il existe un nombre infini de possibilités (problème de multiple parenté). Dans un premier travail,[43] nous avons défini la méthode λ-MR-CCSD en faisant un choix motivé par la théorie des

perturbations :

$$c_i = \frac{\langle i|\widehat{H}|\Psi_0\rangle}{E_0 - \langle i|\widehat{H}|i\rangle} = \sum_I \frac{\langle i|\widehat{H}|I\rangle}{E_0 - \langle i|\widehat{H}|i\rangle} = \sum_I c_I d_{Ii} \tag{3.37}$$

On peut remarquer que le rapport des amplitudes ne dépend pas de la référence[45] :

$$\frac{d_{Ii}}{d_{Ji}} = \frac{\langle i|\widehat{H}|I\rangle}{\langle i|\widehat{H}|J\rangle} \tag{3.38}$$

On choisit donc une expression des d_{Ii} qui vérifie cette propriété, ce qui nous donne la définition suivante

$$d_{Ii} = \lambda_i \langle I|\widehat{H}|i\rangle \text{ avec } \lambda_i = \frac{c_i}{\langle i|\widehat{H}|\Psi_0\rangle} \tag{3.39}$$

Dans un second travail,[46] nous avons proposé une autre définition des amplitudes, obtenues par la minimisation de $\|\Psi - (1+\widehat{T})\Psi_{\text{CAS}}\|$. Nous avons appelé cette méthode le μ-MR-CCSD.

On peut maintenant exprimer la fonction d'onde Coupled Cluster tronquée aux quadruples excitations en faisant intervenir les déterminants $|\alpha\rangle$:

$$\begin{aligned} \Psi &= \sum_I c_I|I\rangle + \sum_i c_i|i\rangle + \sum_\alpha c_\alpha|\alpha\rangle \\ &= \sum_I c_I \left(|I\rangle + \sum_i d_{Ii}|i\rangle + \sum_\alpha d_{I\alpha}|\alpha\rangle\right) \end{aligned} \tag{3.40}$$

et en appliquant le formalisme Coupled Cluster à chacun des déterminants de la référence, on obtient une définition des amplitudes des triples et quadruples excitations en fonction des simples et doubles :

$$d_{I\alpha} = \sum_{(i,j)} d_{Ii} d_{Ij}, \qquad \{(i,j) \mid \widehat{T}_\alpha = \widehat{T}_i \widehat{T}_j\}. \tag{3.41}$$

On écrit maintenant les équations aux valeurs propres projetées sur les $\{|i\rangle\}$:

$$\begin{aligned} E\langle i|\Psi\rangle &= \langle i|\widehat{H}|\Psi\rangle \\ 0 &= c_i\left(H_{ii} - E\right) + \sum_I c_I H_{Ii} + \sum_{i\neq j} c_j H_{ij} + \sum_\alpha c_\alpha H_{i\alpha} \\ 0 &= c_i\left(H_{ii} - E\right) + \sum_I c_I \left(H_{Ii} + \sum_{i\neq j} d_{Ij} H_{ij} + \sum_\alpha d_{I\alpha} H_{i\alpha}\right) \end{aligned} \tag{3.42}$$

et l'on définit la matrice d'habillage comme

$$\Delta_{Ii} = \sum_\alpha d_{I\alpha} H_{i\alpha}. \tag{3.43}$$

L'équation aux valeurs propres devient

$$\begin{aligned} 0 &= c_i \left(H_{ii} - E\right) + \sum_I c_I \left(H_{Ii} + \Delta_{Ii} + \sum_{i \neq j} d_{Ij} H_{ij} \right) \\ 0 &= c_i \left(H_{ii} - E\right) + \sum_I c_I \left(H_{Ii} + \Delta_{Ii}\right) + \sum_{i \neq j} c_j H_{ij} \\ 0 &= c_i \left(\tilde{H}_{ii} - E\right) + \sum_I c_I \tilde{H}_{Ii} + \sum_{i \neq j} c_j \tilde{H}_{ij}. \end{aligned} \tag{3.44}$$

Le problème MR-CCSD est donc exprimé comme un interaction de configurations avec un hamiltonien modifié. Ainsi, nous pouvont utiliser cet algorithme a partir de fonctions d'ondes sélectionnées dans l'espace CAS+SD.

Applications

Nous avons calculé la courbe de dissociation de F_2 dans la base cc-pVDZ aux niveaux Full-CI, MR-CISD et MR-CCSD. L'espace de référence est un CAS-SCF comportant 2 électrons dans 2 orbitales. Cette molécule est un bon exemple car il s'agit d'étudier la brisure en singulet d'une simple liaison en deux atomes comportant chacun un électron célibataire. À cause des forts effets de corrélation dynamique, l'énergie de liaison au niveau CAS-SCF est deux fois trop faible par rapport à sa valeur Full-CI. Pour montrer l'efficacité de la méthode, nous avons tracé la courbe qui montre la différence d'énergie entre l'énergie Full-CI et l'énergie MR-CISD ou MR-CCSD (Figure 3.9).

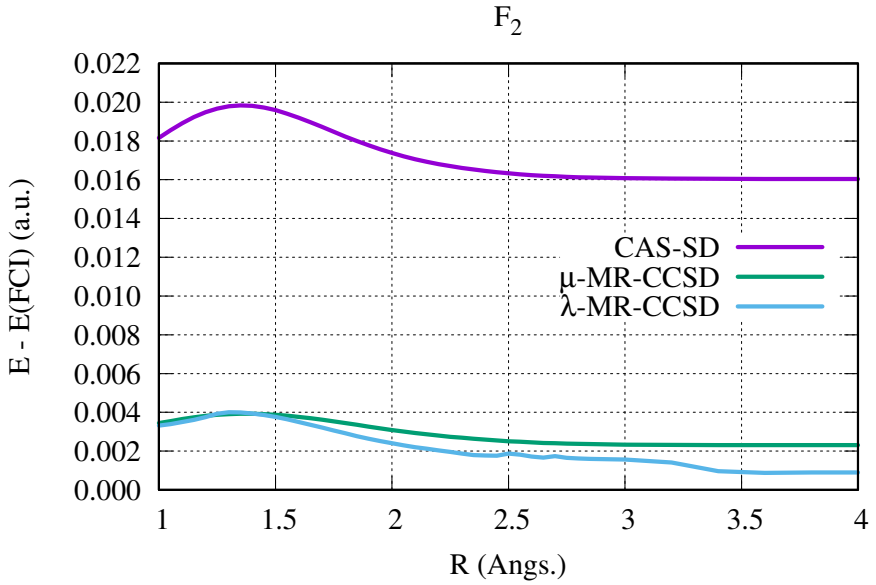


FIGURE 3.9 – Courbe de dissociation de F_2. Différence d'énergie entre le Full-CI et le MR-CISD ou le MR-CCSD.

Sur cette courbe on peut constater tout d'abord que l'énergie MR-CCSD ne diffère de l'énergie Full-CI que de moins de 4 mE_h, tandis que l'énergie MR-CISD diffère de plus de 16 mE_h. Cela montre que la quasi-totalité de l'énergie de corrélation est prise en compte. De plus, puisque la différence par rapport à l'énergie Full-CI fluctue moins en MR-CCSD qu'en MR-CISD, le calcul des différences d'énergies est de meilleure qualité.

De la même façon, sur d'autres tests tels que la brisure de N_2, de l'éthane, de l'éthylène, la rotation de l'éthylène le long de l'axe C–C, l'insertion d'un atome de Be dans la molécule de dihydrogène, ou la brisure simultanée des deux liaisons O–H dans l'eau, la méthode MR-CCSD proposée a toujours donné des résultats nettement supérieurs au MR-CISD.

3.1.6 Implémentation massivement parallèle

Quantum Package

En 2013, au cours de la thèse d'Emmanuel Giner nous avons créé le *Quantum Package*, un code open source qui facilite le développement de nouvelles méthodes de chimie quantique. Ce code n'a pas pour vocation de remplacer les gros codes généralistes de la communauté, mais il s'agit plutôt d'un laboratoire virtuel de développement de nouvelles idées. Par exemple, nous expérimentons dans ce programme un modèle de parallélisme avec une gestion dynamique des ressources et tolérant aux pannes.

Au début de ce chapitre, la méthode CIPSI a été présentée comme un algorithme de résolution du Full-CI. Pour généraliser cet algorithme à toutes les méthodes d'interaction de configurations, il est nécessaire d'introduire quelques définitions. On définit d'abord des sous-ensembles d'orbitales permettant de construire les opérateurs de simples et doubles d'excitations. Ces ensembles sont codés comme des *masques* permettant de filtrer les orbitales. On a donc un masque de trou et un masque de particules pour les mono-excitations, et pour les doubles excitations on a un masque de premier trou, de second trou, de première particule et de seconde particule. On définit ensuite les *déterminants générateurs*. Il s'agit de l'ensemble des déterminants de la fonction d'onde variationnelle sur lesquels on pourra appliquer les opérateurs d'excitation. Enfin, on définit les *déterminants sélecteurs*. Ce sont les déterminants de la fonction d'onde variationnelle $|i\rangle$ qui interviennent dans l'expression de la perturbation

$$\delta e_\alpha = \frac{\left(\sum_i c_i \langle i|\widehat{H}|\alpha\rangle\right)^2}{\Delta E_\alpha}. \tag{3.45}$$

Ainsi, on peut programmer n'importe quelle méthode d'interaction de configurations dans sa version sélectionnée en quelques minutes. De même, en changeant la définition des dénominateurs dans le calcul de la perturbation on peut facilement définir de nouvelles méthodes de perturbation multi-références. Par exemple, pour programmer la méthode CID il suffit de définir les masques des simples excitations comme des masques vides (0), et de définir les masques qui permettent toutes les doubles excitations depuis le déterminant de Hartree-Fock. L'ensemble des déterminants générateurs et des sélecteurs sont définis comme le déterminant de Hartree-Fock. Si maintenant on veut définir un CAS+SD, il suffit de définir les opérateurs d'excitation tels qu'il est possible de faire des trous et des particules dans toutes les orbitales définies par le CAS, des trous dans les orbitales inactives et des particules dans les orbitales virtuelles. Les déterminants générateurs sont définis comme tous les déterminants qui ont exactement deux électrons dans chacune des orbitales inactives et exactement zéro dans chacune des orbitales virtuelles. Les déterminants sélecteurs sont tous les déterminants de la fonction d'onde variationnelle.

Algorithmes determinant-driven

Si l'on regarde l'algorithme CIPSI, on constate que le nombre de déterminants à prendre en compte reste toujours petit devant le nombre de déterminants dont on a besoin avec les algorithmes d'interaction de configurations usuels. On a rarement plus de 10 millions de déterminants à traiter variationnellement, donc les vecteurs impliqués dans la diagonalisation de l'hamiltonien tiennent toujours en mémoire. Cependant, on ne peut pas savoir exactement quels déterminants ont été sélectionnés sans les regarder explicitement. Ainsi, pour que ces algorithmes soient efficaces il est crucial de pouvoir comparer deux déterminants extrêmement rapidement. Une fonction de comparaison permet d'une part de trier les déterminants (arbitrairement), et grâce à cela la recherche d'un déterminant dans la fonction d'onde peut être effectué en temps logarithmique. Aussi, pour chaque couple de déterminants il est nécessaire de pouvoir connaître rapidement le degré d'excitation de l'un par rapport à l'autre car s'ils sont plus que doublement excités ils n'interagissent pas. S'ils interagissent, il faut être capable de retrouver l'opérateur d'excitation qui transforme un déterminant en l'autre, afin de déterminer quelles sont les intégrales impliquées dans l'élément de matrice considéré. À ces fins, nous avons proposé une implémentation des règles de Slater-Condon extrêmement efficace[47] qui permet d'obtenir le degré d'excitation entre deux déterminants plus rapidement qu'une division flottante, et qui permet de retrouver l'opérateur d'excitation pour un coût de l'ordre d'un accès aléatoire à la mémoire. Au-delà de cette implémentation efficace, les déterminants sont triés pour permettre d'extraire des petites listes intermédiaires de déterminants qui interagissent. Les déterminants sont regroupés par composante α commune, si bien que pour un déterminant donné, tous les déterminants qui interagissent avec lui sont

- au sein du même groupe α, tous les déterminants qui ont une simple ou double excitation β
- dans les groupes où α est une mono-excitation, tous les déterminants où la partie β diffère par au plus une orbitale
- dans les groupes ou α est une double excitation, le determinant qui a la même partie β.

Une seconde contrainte pour les algorithmes *determinant-driven* est la nécessité d'avoir un accès aléatoire rapide aux intégrales mono- et bi-électroniques en base d'OM. Si l'on utilise un tableau à quatre indices pour les stocker, on ne peut alors traiter que de tout petits systèmes car la taille du tableau croît comme $\mathcal{O}(N^4)$. Il est donc nécessaire d'avoir un format de stockage compacte qui permet un accès très rapide, mais qui en plus autorise les lectures/écritures en parallèle. Dans le programme, on dispose d'une structure de données de type clé-valeur qui s'apparente à une table de hachage où les insertions et modifications peuvent être effectuées en parallèle. La clé est calculée par une fonction $f(i,j,k,l) \to I$ qui transforme quatre indices en un entier long, en tenant compte des symétries de permutation entre les indices ($i \leq k$, $j \leq l$ et $i \leq j$). Pour des raisons d'efficacité, un cache de 2 milliards d'intégrales intégrales impliquant 4 orbitales dont les indices sont les plus proches du niveau de Fermi est également stocké et fournit un accès plus rapide à ces intégrales.

Parallélisme

Quand on développe un programme parallèle, on cherche toujours a avoir la meilleure efficacité parallèle possible. Un programme de chimie quantique passe par de nombreuses phases différentes (calcul des intégrales atomiques, calcul SCF, transformation à quatre indices, sélection de déterminants, perturbation, diagonalisation, *etc*). En utilisant les modèles usuels du HPC où les ressources sont fixes, si

l'on veut obtenir une bonne efficacité parallèle il faut que *toutes* les étapes aient une bonne efficacité parallèle. Si maintenant on se place dans un modèle où les ressources sont flexibles, on peut adapter au vol les ressources afin que l'efficacité parallèle soit toujours bonne, même si une étape a un moins bon scaling qu'une autre. Cela permet un meilleur partage des ressources avec les autres utilisateurs (mais aussi avec soi-même).

Pour ce qui est de la parallélisation intra-nœud du Quantum Package, on utilise OpenMP pour les tâches assez rapides. Pour les tâches plus lourdes, on fait appel au modèle client/serveur du Quantum Package. Dans ce modèle, un serveur de tâches, écrit en OCaml, gère la file d'attente des tâches à exécuter, et connaît l'état du système en sachant quelle tâche s'exécute sur quel nœud. Lorsqu'une nouvelle section parallèle démarre, le processus maître Fortran crée une nouvelle file d'attente pour le calcul parallèle en envoyant un message par le réseau au serveur de tâches. Puis, il peuple ce serveur avec une liste de tâches à effectuer. Une thread du processus Fortran maître, le *collecteur*, se prépare à recevoir les résultats par le réseau, et les autres threads sont des esclaves de calcul. Les threads de calcul profitent de la mémoire partagée pour économiser de la mémoire en lecture seule (comme les intégrales bi-électroniques par exemple), mais elles n'écrivent pas dans la mémoire partagée. Elles ne communiquent qu'avec le serveur de tâches ou le collecteur par échange de messages. Cela garantit d'une part une certaine sécurité pour l'écriture des données en mémoire, et d'autre part dans ce modèle, la programmation multi-thread ou distribuée utilise exactement la même API, ce qui facilite grandement l'écriture du code. L'API choisie est celle fournie par la bibliothèque ZeroMQ.[13]

Lorsqu'un nouveau processus esclave doit participer au travail collectif, il s'enregistre sur le serveur de tâches et il peut alors obtenir une tâche à effectuer à condition qu'il soit bien le client adéquat et dans le bon état. Lorsqu'une tâche est terminée, le client envoie au serveur de tâches l'information qu'il a fini de calculer la tâche, et il envoie au collecteur son résultat. À ce moment-là, le client a complété son travail et peut demander une nouvelle tâche au serveur. Quand le collecteur reçoit le résultat, il informe le serveur de tâches que le résultat a bien été traité. Cette gestion des tâches permet à tout moment de garantir que toutes les tâches ont bien suivi toutes les étapes pour produire le résultat du calcul. Si une tâche a échoué, il est assez facile de remettre la tâche en queue pour qu'elle soit effectuée à nouveau et que le résultat ne soit pas perdu. À tout moment un nouveau client peut se connecter au serveur de tâches pour participer au calcul, ou il peut se déconnecter s'il n'est pas en train d'exécuter une tâche.

À l'heure actuelle, plusieurs sous-programmes ont été parallélisés en utilisant ce modèle : le calcul des intégrales atomiques, le calcul de $\widehat{H}|\Psi\rangle$ dans la diagonalisation de Davidson, la sélection perturbative des déterminants où chaque générateur correspond à une tâche, et le calcul (stochastique ou non) de la correction perturbative de l'énergie. L'efficacité parallèle est en général très satisfaisante : pour le moment nous réalisons de façon routinière des calculs utilisant sans difficulté 250 nœuds (4000 cœurs) du supercalculateur Curie. Cependant, l'algorithme de Davidson est celui qui pose le plus de difficultés car il repose fortement sur la qualité du réseau.

Un grand avantage de ce modèle de parallélisation est la possibilité de soumettre au gestionnaire de batch un job maître, puis plusieurs jobs esclaves (d'une dizaine à une cinquantaine de nœuds) qui viendront augmenter plus tard les ressources du calcul en cours. Si l'utilisateur décide de libérer des ressources, il peut également le faire. Au laboratoire dans le cadre d'un partenariat avec HP nous disposons de 68 nœuds avec des processeurs à 8 cœurs à faible consommation de type Atom. Ces processeurs sont parfaitement bien adaptés au calcul de la sélection de déterminants

dans l'algorithme CIPSI, mais la mauvaise qualité du réseau rend l'étape de diagonalisation peu efficace. Si deux utilisateurs du cluster lancent chacun un job sur un nœud standard (bisocket de type Xeon), chacun peu mettre en queue une réservation des 68 nœuds Atom pour l'étape de sélection. Lorsque la sélection est terminée, les ressources sont libérées et le noeud Xeon entre dans la phase de diagonalisation. A ce moment-là, les ressources sont disponibles pour que le second calcul utilise à son tour les nœuds Atom pour la sélection. Ainsi, plusieurs calculs peuvent s'échanger des ressources sans que les calculs soient interrompus, et les ressources peuvent être ajustées au vol pour garantir une efficacité parallèle optimale.

3.2 Monte Carlo Quantique (QMC)

3.2.1 Principe

Le terme *Monte Carlo quantique* englobe plusieurs méthodes. La plus simple est la méthode *Monte Carlo Variationnel* (VMC), qui consiste à utiliser une méthode stochastique d'intégration pour évaluer les valeurs moyennes des observables de la fonction d'onde d'essai. Puis, les méthodes *Diffusion Monte Carlo* (DMC) avec nœuds fixés sont les méthodes de référence pour effectuer des calculs au-delà de la méthode variationnelle (VMC). Elles consistent en une extraction de la composante de l'état fondamental de la fonction d'essai par une technique de projection.

Monte Carlo Variationnel (VMC)

Une observable O est calculée comme

$$O = \frac{\langle\Psi|\widehat{O}|\Psi\rangle}{\langle\Psi|\Psi\rangle} = \frac{\int \Psi(\mathbf{R})\widehat{O}\Psi(\mathbf{R})\ \mathrm{d}\mathbf{R}}{\int |\Psi(\mathbf{R})|^2\ \mathrm{d}\mathbf{R}} \tag{3.46}$$

Lorsque la fonction d'onde est telle que son carré ne peut pas être intégré analytiquement, il faut avoir recours à des méthodes d'intégration numérique telles que les méthodes de Monte Carlo. Le Monte Carlo Variationnel (VMC) consiste à réécrire le calcul de l'observable comme

$$O = \frac{\int |\Psi(\mathbf{R})|^2 O_{\mathrm{loc}}(\mathbf{R})\ \mathrm{d}\mathbf{R}}{\int |\Psi(\mathbf{R})|^2\ \mathrm{d}\mathbf{R}} \ \text{ avec } \ O_{\mathrm{loc}}(\mathbf{R}) = \frac{\widehat{O}\Psi(\mathbf{R})}{\Psi(\mathbf{R})}. \tag{3.47}$$

Ainsi, la valeur de l'observable est exprimée comme la valeur moyenne de O_{loc} pondérée par la densité à $3N$ particules donnée par le carré de la fonction d'onde. Pour estimer la valeur de O, il suffit de tirer au hasard un très grand nombre de points $\mathbf{R}_M$ dans l'espace à $3N$ dimensions distribués suivant $\frac{|\Psi(\mathbf{R})|^2}{\langle\Psi|\Psi\rangle}$, et de calculer la moyenne

$$O = \lim_{M\to\infty} \frac{1}{M} \sum_{i=1}^{M} O_{\mathrm{loc}}(\mathbf{R}_M). \tag{3.48}$$

Pour un nombre M fini de tirages, les points sont obtenus par une marche aléatoire Brownienne avec dérive :

$$\mathbf{R}_{n+1} = \mathbf{R}_n + \tau \frac{\nabla\Psi(\mathbf{R}_n)}{\Psi(\mathbf{R}_n)} + \mathbf{G}_n \tag{3.49}$$

où τ est le pas de temps et $\mathbf{G}$ est un vecteur aléatoire gaussien de variance $\sigma^2 = \tau$. L'algorithme de Metropolis-Hastings est utilisé pour supprimer l'erreur de discrétisation.

Diffusion Monte Carlo (DMC)

L'équation de Schrödinger en temps imaginaire est

$$\begin{aligned} -\frac{\partial \Psi(\mathbf{R},t)}{\partial t} &= \left[-\frac{1}{2}\nabla^2 + V(\mathbf{R}) - E_T\right]\Psi(\mathbf{R},t) \\ \frac{\partial \Psi(\mathbf{R},t)}{\partial t} &= D\nabla^2\Psi(\mathbf{R},t) - \left[V(\mathbf{R}) - E_T\right]\Psi(\mathbf{R},t) \end{aligned} \tag{3.50}$$

Le premier terme de cette équation peut être assimilé à une équation de diffusion dans l'espace des $3N$ coordonnées où $\Psi(\mathbf{R},t)$ joue le rôle de la densité de particules qui diffusent, et le facteur $D = 1/2$ devant le laplacien correspond au coefficient de diffusion. Une telle équation peut être facilement simulée avec une marche aléatoire des particules dans l'espace des configurations $\mathbf{R}$ (Eq. (3.49)). Le second terme ressemble, quant à lui, à une équation de vitesse décrivant un processus de mort et de naissance d'individus dans une population. Ainsi, l'équation (3.50) peut être simulée par la combinaison d'un processus de diffusion et d'un processus de branchement, dans lequel le nombre de particules diffusant augmente ou diminue de façon à réduire la densité de probabilité dans les régions où $V(\mathbf{R})$ est grand et à l'augmenter dans les zones d'énergie potentielle minimale.

Pour que l'interprétation en terme de diffusion reste valide, il faut que $\Psi(\mathbf{R},t)$, qui représente une densité de particules dans l'équation (3.50), reste d'un signe constant (choisi positif) dans tout l'espace. Cette méthode paraît donc restreinte aux systèmes bosoniques dont la fonction d'onde fondamentale n'admet pas de zéros (*nœuds* dans l'espace des $3N$-coordonnées). Compte tenu du caractère antisymétrique de la fonction d'onde des fermions, celle-ci est nécessairement négative dans certaines régions de l'espace. La méthode des nœuds fixés rend possible le DMC pour les fermions en séparant la simulation des régions positives et négatives de la fonction d'essai, et en interdisant la diffusion à travers l'hypersurface nodale. Ainsi, les simulations convergent vers l'état fondamental fermionique d'un hamiltonien ayant des conditions aux limites légèrement modifiées (la contrainte nodale), ce qui introduit une erreur si l'hypersurface nodale est différente de celle associée à la fonction d'onde exacte.

L'approximation des nœuds fixés

L'*approximation des nœuds fixés* (Fixed-Node DMC) consiste à effectuer la simulation DMC en imposant comme surface nodale celle de la fonction d'essai Ψ. On obtient donc la meilleure énergie possible sous cette contrainte, et puisque l'approximation des nœuds fixés est la seule approximation de la méthode FN-DMC, celle-ci donne l'énergie exacte si l'hypersurface nodale est exacte. Si ce n'est pas le cas, elle donne une borne supérieure de l'énergie.

Toute la difficulté de la méthode FN-DMC réside dans le contrôle de l'erreur des nœuds fixés. Plus la fonction d'essai est "bonne" (de basse énergie), plus elle est proche de la fonction d'onde exacte. On peut donc supposer que si la fonction d'essai est suffisamment proche de la fonction d'onde exacte, abaisser son énergie va induire une réduction de l'erreur due aux nœuds fixés.

La pratique usuelle consiste donc à optimiser la fonction d'onde tant que faire se peut, et d'espérer que l'erreur des nœuds fixés va s'en retrouver suffisamment amoindrie. Pour cela, on introduit une fonction d'essai compacte reproduisant au mieux les propriétés physiques et mathématiques de la fonction d'onde exacte. Puisque le QMC ne nécessite pas de calculer des intégrales bi-électroniques, la liberté de choix de forme de fonction d'onde est bien plus grande, et la forme généralement utilisée est de type Slater-Jastrow :

$$\Psi(\mathbf{R}) = \left(\sum_i c_i D_i(\mathbf{R})\right) \exp(J(\mathbf{R})) \tag{3.51}$$

où le facteur de Jastrow $\exp(J(\mathbf{R}))$ décrit explicitement les interactions électron-électron (comme dans les méthodes de type f_{12}), et les interactions électron-électron-noyau. Sa forme est telle qu'elle ne permet pas l'intégration analytique, et l'énergie de la fonction d'onde doit être calculée en VMC. Remarquons que le facteur de Jastrow est une fonction positive en tout point, donc il ne change pas les surface nodales qui sont imposées par la composante déterminantale. L'énergie obtenue par un calcul DMC à nœuds fixés est donc la même que celle obtenue en absence du facteur de Jastrow. Afin d'améliorer les surfaces nodales, la composante déterminantale est ré-optimisée en présence du facteur de Jastrow dans une série de calculs de type VMC. Puisqu'il s'agit d'une optimisation non-linéaire dans un environnement bruité, sa mise en œuvre n'est pas triviale et de nombreux travaux ont été élaborés pour rendre possible cette optimisation[48, 49, 50, 51, 52, 53, 54]. Sur de petits systèmes, cette technique s'est avérée extrêmement efficace,[54] mais lorsqu'on tends vers les gros systèmes

- l'erreur des nœuds fixés augmente avec la taille du système
- la qualité de la fonction décroît car sa préparation devient trop coûteuse
- l'optimisation devient de plus en plus difficile à cause du bruit statistique et du nombre croissant de paramètres

L'optimalité de cette approche peut donc être remise en question pour le calcul de différences d'énergie sur des systèmes complexes. En effet, pour qu'une différence d'énergie entre deux systèmes A et B soit exacte il n'est pas nécessaire de supprimer l'erreur. Il suffit seulement que l'erreur commise sur A soit exactement égale à l'erreur commise sur B. La méthode d'optimisation tend à égaler les erreurs sur A et B en les rapprochant le plus possible de zéro. L'approche que nous proposons consiste à ne pas essayer de supprimer l'erreur des nœuds fixés, mais plutôt de chercher à la rendre cohérente sur les systèmes A et B, et continue le long d'un chemin réactionnel. Pour cela, nous choisissons des formes de fonction d'onde qui permettent l'optimisation par des méthodes déterministes, telles que les fonctions d'ondes issues de calculs d'interaction de configurations.

3.2.2 Couplage IC/QMC

Nous avons vu que les fonctions d'ondes de type Full-CI permettent d'obtenir la solution *exacte* dans un espace tronqué. Lorsque la base tend vers la base complète de dimension infinie, le résultat Full-CI devient exact. Les méthodes QMC vont au-delà de l'approximation de base finie : la seule erreur provient de la qualité de la surface nodale qui est paramétrée par la base dans laquelle est exprimée la fonction d'essai. Ainsi, pour des bases suffisamment grandes, plus le calcul Full-CI est exprimé dans une grande base, plus la qualité des surfaces nodales peut être améliorée. On voit bien qu'il est possible d'améliorer de façon mécanique et systématique un calcul QMC,

et donc de contrôler l'erreur des nœuds fixés. Dans ce paradigme, on ne considèrera pas le QMC comme une méthode alternative aux méthodes post-HF, mais plutôt comme une méthode d'amélioration du calcul Full-CI (*post-Full-CI*).

Dans la pratique, il n'est pas envisageable de réaliser de manière routinière des calculs QMC sur des fonctions d'ondes Full-CI. Mais il est tout-à-fait raisonnable de penser que des calculs QMC peuvent être réalisé avec des fonctions d'essai *quasi-Full-CI* obtenues par l'algorithme CIPSI. Bien que le nombre de déterminants soit considérablement réduit par rapport à la taille de l'espace de Hilbert complet, le nombre de déterminants à considérer est tout de même important pour les calculs QMC. Quand nous avons commencé les premiers calculs CIPSI/QMC, le calcul moléculaire avec le plus grand nombre de déterminants de la littérature (à ma connaissance) était le calcul de Clark *et al* sur la molécule d'eau avec environ 18 000 déterminants de Slater.[55] Pour rendre les calculs faisables, il était nécessaire de pouvoir aller bien au-delà et d'atteindre le million de déterminants.

Réduction de la complexité du calcul multi-déterminantal

Dans un calcul QMC, on doit calculer à chaque pas Monte Carlo la valeur de la fonction d'onde $\Psi(\mathbf{R})$, les dérivées premières $\nabla_i \Psi(\mathbf{R}) = \partial\Psi(\mathbf{R})/\partial\mathbf{R}_i$ et le laplacien de la fonction d'onde $\Delta_i \Psi(\mathbf{R})$ par rapport aux coordonnées des électrons. Il est donc critique de pouvoir calculer ces quantités extrêmement rapidement.

La fonction d'onde dans un formalisme *spin-free* s'exprime comme

$$\Psi(\mathbf{R}_\uparrow, \mathbf{R}_\downarrow) = \sum_{k=1}^{N_{\text{det}}} c_k \det\left[\mathbf{S}_k^\uparrow(\mathbf{R}_\uparrow)\right] \det\left[\mathbf{S}_k^\downarrow(\mathbf{R}_\downarrow)\right]. \tag{3.52}$$

Pour un spin $\sigma \in \{\uparrow, \downarrow\}$, les éléments de la matrice de Slater $\mathbf{S}_k^\sigma$ de taille $N_\sigma \times N_\sigma$ sont $[\mathbf{S}_k^\sigma]_{ij} = \phi_j(\mathbf{r}_i^\sigma)$ où les ϕ sont les orbitales moléculaires. Cette expression comporte en général un grand nombre de matrices de Slater identiques. Nous utiliserons donc la forme bi-linéaire suivante qui permet de n'exprimer qu'une seule fois chacun des déterminants $D^\uparrow$ et $D^\downarrow$ [7] :

$$\Psi(\mathbf{R}_\uparrow, \mathbf{R}_\downarrow) = \sum_{i=1}^{N_{\text{det}}^\uparrow} \sum_{j=1}^{N_{\text{det}}^\downarrow} C_{ij} D_i^\uparrow(\mathbf{R}_\uparrow) D_j^\downarrow(\mathbf{R}_\downarrow) \tag{3.53}$$

Dans l'espace de Hilbert complet, $N_{\text{det}} = N_{\text{det}}^\uparrow \times N_{\text{det}}^\downarrow$. On voit donc que le nombre de déterminants à calculer peut être de l'ordre de $\sqrt{N_{\text{det}}}$.

$$\Psi = \left(\mathbf{D}^{\uparrow\dagger}\mathbf{C}\right)\mathbf{D}^\downarrow \tag{3.54}$$

$$\nabla\Psi = \nabla\mathbf{D}^{\uparrow\dagger}\left(\mathbf{C}\mathbf{D}^\downarrow\right) + \left(\mathbf{D}^{\uparrow\dagger}\mathbf{C}\right)\nabla\mathbf{D}^\downarrow \tag{3.55}$$

$$\Delta\Psi = \mathbf{\Delta}\mathbf{D}^{\uparrow\dagger}\left(\mathbf{C}\mathbf{D}^\downarrow\right) + \left(\mathbf{D}^{\uparrow\dagger}\mathbf{C}\right)\mathbf{\Delta}\mathbf{D}^\downarrow \tag{3.56}$$

Une fois la fonction d'onde réécrite sous la forme bilinéaire, elle peut être calculée comme deux produits de vecteurs (Eq (3.54)) : le produit vecteur-matrice $\mathbf{D}^{\uparrow\dagger}\mathbf{C}$ impliquant N_{det} multiplications et additions, suivi d'un produit vecteur-vecteur n'impliquant que $N_{\text{det}}^\downarrow \sim \sqrt{N_{\text{det}}}$ multiplications et additions. On voit que pour le calcul du gradient et du laplacien les produits $\mathbf{D}^{\uparrow\dagger}\mathbf{C}$ et $\mathbf{C}\mathbf{D}^\downarrow$ peuvent être pré-calculés, ce qui permet de n'avoir besoin que de $\mathcal{O}(\sqrt{N_{\text{det}}} \times N_{\text{elec}})$ opérations flottantes. La seule

étape en $\mathcal{O}(N_{\text{det}})$ est le calcul des deux produits $\mathbf{D}^{\uparrow\dagger}\mathbf{C}$ et $\mathbf{C}\mathbf{D}^{\downarrow}$. Il s'agit du produit d'un vecteur dense par une matrice creuse, et ce type d'opération est nécessairement *memory-bound*. Ainsi, pour minimiser le temps de calcul, ces deux produits sont effectués simultanément afin de réutiliser les valeurs C_{ij} chargées. En pratique, quand $N_{\text{det}} = 10^6$, nous avons mesuré que cette étape ne prend qu'environ 20% du temps de calcul total.

Pour aller encore plus loin dans la réduction du temps de calcul, nous avons proposé une méthode qui permet de tronquer la fonction d'onde d'une manière qui permet d'obtenir le meilleur rapport nombre de déterminants / temps de calcul. On a vu que la majorité du temps est passée pour le calcul des déterminants (les vecteurs $\mathbf{D}^{\uparrow}$ et $\mathbf{D}^{\downarrow}$). Il s'agit donc de définir un seuil qui permet de supprimer spécifiquement ceux qui sont les moins importants. Pour cela, on définit la contribution à la norme d'un déterminant $\uparrow$ ou $\downarrow$ d'après

$$\mathcal{N} = \sum_{i=1}^{N_{\text{det}}^{\uparrow}} \mathcal{N}_i^{\uparrow} \quad = \quad \sum_{j=1}^{N_{\text{det}}^{\downarrow}} \mathcal{N}_j^{\downarrow} \tag{3.57}$$

$$\mathcal{N}_i^{\uparrow} = \sum_{j=1}^{N_{\text{det}}^{\downarrow}} C_{ij}^2 \qquad \mathcal{N}_j^{\downarrow} = \sum_{i=1}^{N_{\text{det}}^{\uparrow}} C_{ij}^2 \tag{3.58}$$

et on supprime les déterminants pour lesquels $\mathcal{N}^{\uparrow}$ et $\mathcal{N}^{\downarrow}$ sont inférieurs à un seuil ϵ. La figure 3.10 montre l'effet de l'application du seuil sur l'énergie DMC d'une fonction d'onde à $N_{\text{det}} = 10^6$ pour l'atome de Chlore (cc-pVTZ). Avec un seuil $\epsilon = 10^{-5}$ l'énergie DMC a convergé. On peut constater d'après le tableau 3.1 que cela correspond à une accélération du temps de calcul d'un facteur 172×, soit un coût de calcul seulement 7× plus long que le calcul mono-déterminantal.

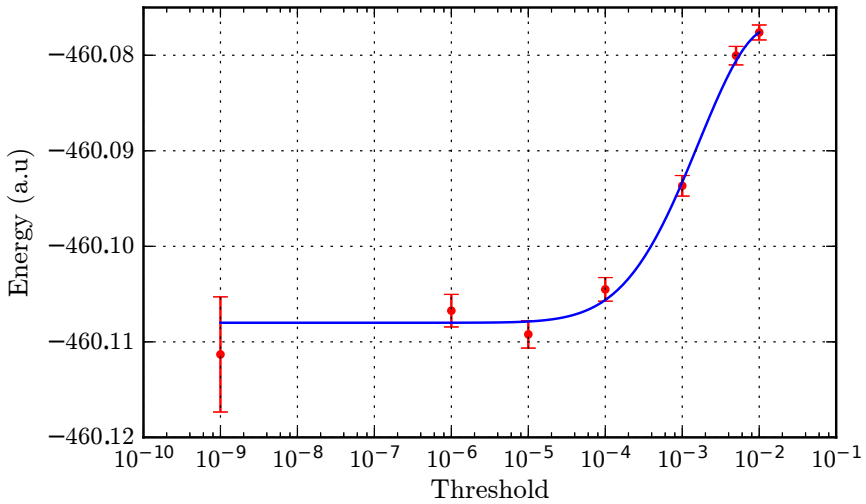


FIGURE 3.10 – Énergie DMC de l'atome de Chlore en fonction du seuil appliqué à la fonction d'onde.

La molécule d'eau

Nous avons réalisé des calculs CIPSI sur la molécule d'eau dans sa géométrie d'équilibre avec des bases de plus en plus grandes (cc-pCVnZ, $n = D, T, Q, 5$) en vue d'obtenir une estimation de l'énergie exacte.[26] Les valeurs CIPSI corrigées avec la perturbation à l'ordre 2 et les valeurs DMC sont représentées sur la figure 3.11.

ϵ	Ndet	Ndet$^{\uparrow}$	Ndet$^{\downarrow}$	CPU time(ms)
10^{-2}	1	1	1	0.02450
5.10^{-3}	3	3	2	0.02688
10^{-3}	86	21	17	0.03899
5.10^{-4}	214	29	28	0.04636
10^{-4}	1 361	93	74	0.08808
5.10^{-5}	2 424	120	89	0.1054
10^{-5}	9 485	234	166	0.1855
10^{-6}	54 016	772	523	0.5960
10^{-7}	207 995	2 279	1 389	1.740
10^{-8}	459 069	5 797	3 291	4.196
10^{-9}	748 835	14 456	8 054	9.724
10^{-10}	926 299	30 320	16 571	19.18
0	1 000 000	52 433	26 833	31.92

TABLE 3.1 – Nombre de déterminants dans la fonction d'onde de l'atome de Chlore et temps de calcul d'un pas Monte Carlo après application du seuil.

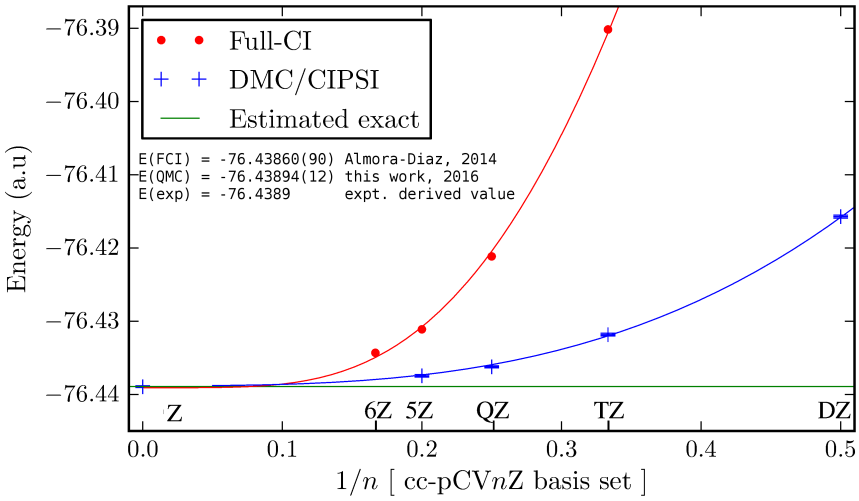


FIGURE 3.11 – Extrapolation en base infinie de l'énergie Full-CI et CIPSI/DMC de la molécule d'eau.

L'extrapolation en base infinie du calcul CIPSI/DMC donne une valeur de −76.43894(12) ua. Ce résultat est compatible avec la valeur de −76.4390(4) ua obtenue par l'extrapolation de l'énergie Full-CI de la littérature.[56] C'est un résultat crucial car il démontre que l'erreur des nœuds fixés est maintenant contrôlée et peut donc être améliorée de façon *systématique*. Le tableau 3.2 montre un autre résultat important : la valeur de -76.43744(18) que nous avons obtenue en cc-pCV5Z est la plus basse valeur variationnelle obtenue à ce jour.

Diffusion Monte Carlo (borne supérieure)	
Clark *et al.*,[55] DMC mono-déterminant	-76.3938(4)
Casula *et al.*,[57] DMC AGP	-76.4175(4)
Gurtubay *et al.*,[58] DMC B3LYP	-76.4230(1)
Gurtubay *et al.*,[58]DMC SJB	-76.42830(5)
Lüchow et Fink,[59] DMC PNO-CI	-76.429(1)
Clark *et al.*,[55] DMC MSDJ	-76.4368(4)
Ce travail,[26] DMC cc-pCV5Z $\sim 1.410^6$ dets	-76.43744(18)
CI (borne supérieure) et Coupled Cluster	
Almora-Dìaz,[60] CISDTQQnSx/cc-pCV6Z	-76.4343
Muller ad Kutzelnigg ,[61] CCSD(T)/R12	-76.4373
Helgaker *et al.*,[62] CCSD(T)/R12	-76.439(2)
Limite en base complète (CBS)	
Almora-Dìaz,[60] FCI + CBS (cc-pCVnZ)	-76.4386(9)
Halkier *et al.*,[63] CCSD(T)+CBS	-76.4386
Bytautas et Ruedenberg,[64] FCI+CBS	-76.4390(4)
Ce travail,[26] DMC + CBS (cc-pCVnZ)	-76.43894(12)
Klopper,[56] estimation exacte non-relativiste	-76.4389

TABLE 3.2 – Comparaison des énergies totales non relativistes de la molécule d'eau (au).

Dissociation de F_2

F_2 est une molécule particulière car elle possède un fort couplage entre la corrélation statique et la corrélation dynamique. La prise en compte d'un seul de ces effets ne décrit pas correctement le système. En effet, la corrélation dynamique des paires libres des atomes de fluor tend à repousser les atomes entre eux. Cet effet affaiblit la liaison et se traduit par des effets de corrélation statique.

La description d'une courbe de dissociation met en avant une difficulté : pour que la courbe soit correcte, il faut qu'elle soit le plus parallèle possible à la courbe exacte afin de retrouver les propriétés spectroscopiques (distance d'équilibre R_{eq}, constante de force k et énergie d'atomisation D_0) qui sont reliées à des différences d'énergies. Ainsi, il est nécessaire que l'erreur sur chacun des points de la courbe soit parfaitement contrôlée.[65]

Lorsque l'on effectue un calcul CIPSI et que le nombre de déterminants est fixé *a priori* égal en chaque point, on peut constater que la contribution perturbative à l'énergie n'est pas constante tout au long de la courbe (tableau ??). Il en résulte un non-parallélisme entre la courbe d'énergie variationnelle CIPSI et la courbe Full-CI. Lorsque ces nœuds sont utilisés comme fonction d'essai pour le QMC, ce défaut de parallélisme se répercute sur la qualité des surfaces nodales et les propriétés sont mal estimées (tableau ??).

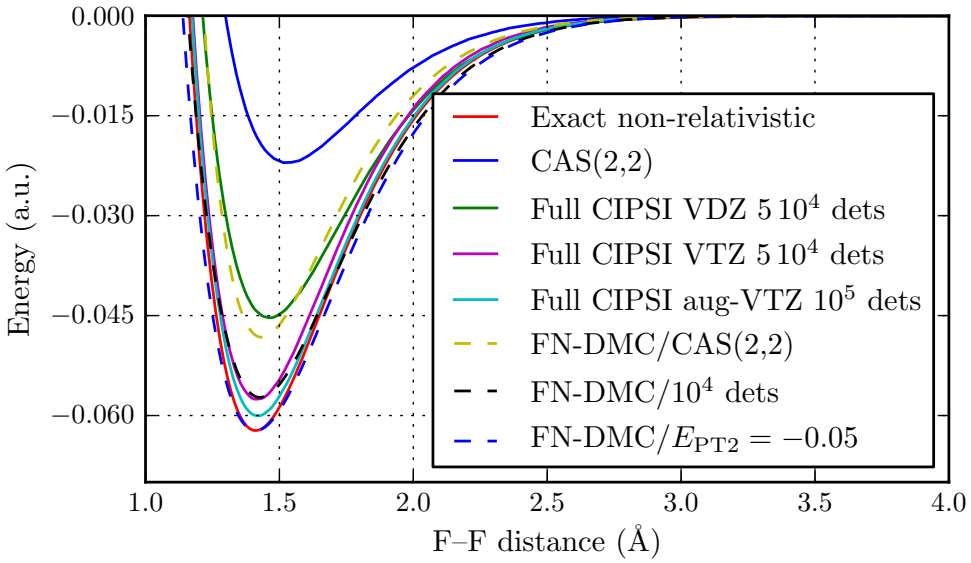


FIGURE 3.12 – Courbes d'énergie potentielle en utilisant les diverses approches, et comparaison avec l'estimation exacte non-relativiste. Chaque courbe est décalée en énergie pour imposer la limite atomique exacte.

Pour obtenir des courbes de dissociation variationnelles les plus homogènes possible, au lieu d'imposer un nombre fixe de déterminants à chaque point de la courbe il est préférable d'utiliser pour chaque distance un nombre de déterminants tel que la valeur de la contribution perturbative à l'énergie E(PT2) est constante. Cela revient à imposer le parallélisme de la courbe variationnelle à l'estimation de la courbe Full-CI, et il en découle que les propriétés spectroscopiques de la courbe variationnelle sont très proches de celles de la courbe Full-CI par construction. Nous avons utilisé ces fonctions d'ondes à nombres de déterminants variables pour réaliser

Estimation exacte non-relativiste					
R_{eq}			1.412		
D_0			62.35 (Ref [66])		
k			1.121		
cc-pVDZ	CAS-SCF		CIPSI / CIPSI+E(PT2)		
N_{dets}	2	$5\ 10^3$	10^4	$5\,10^4$	10^5
R_{eq}	1.531	1.465/1.460	1.464/1.460	1.463/1.462	1.463/1.463
D_0	22.1	40.9/45.7	41.9/45.6	43.8/45.3	43.97/45.17
k	0.43	0.73/0.78	0.75/0.78	0.77/0.77	0.76/0.76
cc-pVTZ	CAS-SCF		CIPSI / CIPSI+E(PT2)		
N_{dets}	2	$5\,10^3$	10^4	$5\,10^4$	10^5
R_{eq}	1.469	1.410/1.412	1.409/1.415	1.418/1.417	1.418/1.417
D_0	26.5	56.8/61.0	49.8/58.6	52.9/57.6	54.14/57.6
k	0.63	1.067/1.132	1.062/1.092	1.037/1.079	1.030/1.079
aug-cc-pVTZ	CAS-SCF		CIPSI / CIPSI+E(PT2)		
N_{dets}	2	$5\,10^3$	10^4	$5\,10^4$	10^5
R_{eq}	1.463	1.405/1.413	1.392/1.414	1.415/1.418	1.418/1.419
D_0	28.4	59.6/66.4	7.5/63.8	52.1/60.2	54./60.0
k	0.66	1.158/1.151	1.108/1.117	1.013/1.078	1.03/1.073
cc-pVDZ	CAS-SCF+DMC			CIPSI+DMC	
N_{dets}	2		10^3	$5\,10^3$	10^4
R_{eq}	1.434		1.419	1.424	1.428
D_0	48.3±0.7		59.6±1.3	56.9±1.6	55.2±1.2
k	1.0±0.2		1.4±0.6	1.0±0.5	1.0±0.4
cc-pVDZ			CIPSI+DMC		
E(PT2)		-0.2	-0.1	-0.05	-0.02
N_{dets}		$\sim 1.6\times10^2$	$\sim 5\times10^2$	$\sim 1.1\times10^3$	$\sim 3.5\times10^3$
R_{eq}		1.414	1.408	1.414	1.425
D_0		53.1±0.4	56.2±0.5	60.3±0.6	54.2±1.3
k		1.839	1.271	1.131	1.125

TABLE 3.3 – Constantes spectroscopiques de F_2. R_{eq} en Å, D_0 en mE_h et k en $E_h/Å^2$. Données obtenues à partir de calculs CIPSI variationnels ou avec correction perturbative, en fonction de la base et du nombre de déterminants. Les valeurs sont représentées comme X_1/X_2 où X_1 est la valeur variationnelle et X_2 est la valeur obtenue avec correction perturbative.

la courbe de dissociation CIPSI/DMC, et nous avons pu constater une bien meilleure homogénéité de la qualité des fonctions d'essai tout au long de la courbe (figure 3.12).

Énergies totales de métaux de transition

Compte tenu du nombre d'électrons dans les éléments de transition 3*d*, leur énergie totale ne peut pas être calculée avec autant de précision que celles des atomes de la première ligne du tableau périodique. Il est cependant nécessaire d'avoir accès aux énergies totales atomiques, comme par exemple dans le développement de nouvelles fonctionnelles d'échange-corrélation. Nous avons donc réalisé des calculs CIPSI+DMC en base de Slater pour tous ces atomes en utilisant entre 11 000 et

48 000 déterminants,[67] et nous avons obtenu les plus basses énergies totales variationnelles obtenues à ce jour (entre 92 et 95% de l'énergie de corrélation, tableau 3.4). Les développements ultérieurs de QMC=Chem sont tels que si nous refaisions cette étude aujourd'hui nous pourrions encore améliorer la qualité des résultats.

Atome	Nœuds HF	% EC	Nœuds OEP[68]	% EC	Nœuds CIPSI	% EC
Sc $[s^2d^1]$	-760.5265(13)	89.2(2)	-760.5288(6)	89.50(7)	-760.5718(16)	94.4(2)
Ti $[s^2d^2]$	-849.2405(14)	89.6(2)	-849.2492(7)	90.55(7)	-849.2841(19)	94.2(2)
V $[s^2d^3]$	-943.7843(13)	89.6(1)	-943.7882(6)	89.95(6)	-943.8234(16)	93.4(2)
Cr $[s^1d^5]$	-1044.3292(16)	91.0(2)	-1044.3289(6)	90.93(6)	-1044.3603(17)	93.9(2)
Mn $[s^2d^5]$	-1150.8880(17)	90.4(2)	-1150.8897(7)	90.54(6)	-1150.9158(20)	92.9(2)
Fe $[s^2d^6]$	-1263.5589(19)	90.1(2)	-1263.5607(6)	90.26(5)	-1263.5868(21)	92.4(2)
Co $[s^2d^7]$	-1382.6177(21)	90.5(2)	-1382.6216(8)	90.85(6)	-1382.6377(24)	92.1(2)
Ni $[s^2d^8]$	-1508.1645(23)	91.6(2)	-1508.1743(7)	92.27(5)	-1508.1901(25)	93.4(2)
Cu $[s^1d^{10}]$	-1640.4271(26)	92.4(2)	-1640.4266(7)	92.34(4)	-1640.4328(29)	92.7(2)
Zn $[s^2d^{10}]$	-1779.3371(26)	91.9(2)	-1779.3425(8)	92.24(5)	-1779.3386(31)	92.0(2)

TABLE 3.4 – Énergies totales (ua) DMC des éléments $3d$ de transition avec différentes surfaces nodales.

3.2.3 Implémentation massivement parallèle

QMC=Chem[69, 70, 71] est le programme QMC que je développe depuis mon arrivée au laboratoire en 2006. De façon générale il calcule des *blocs*, un bloc étant une sous-partie de la simulation où N_w marcheurs ont réalisé M_{bloc} pas Monte Carlo (Figure 3.13). Les propriétés calculées sont moyennées sur toutes les positions de tous les marcheurs dans le bloc, et une valeur moyenne pour chaque propriété est associée au bloc. Si les blocs sont suffisamment longs, ils peuvent être considérés indépendants et les valeurs moyennes des blocs ont alors une distribution gaussienne. Les positions des marcheurs en fin de bloc sont distribuées suivant la densité échantillonnée par le processus, donc il est possible de créer un nouveau bloc en faisant partir les marcheurs de leurs positions finales dans un autre bloc, et avec une nouvelle semence pour le générateur de nombres aléatoire. L'objectif est donc de calculer le plus de blocs possibles le plus rapidement possible. Pour cela, on utilise un modèle client/serveur où les clients calculent des blocs et le serveur collecte les résultats.

Dans l'algorithme DMC, l'étape de branchement impose des synchronisations. Pour éviter cela, chaque cœur gère sa propre population sans communication avec les autres populations. Pour permettre de conserver des nombres de marcheurs relativement petits (de l'ordre de la centaine) et pour éviter les problèmes d'équilibrage de charge entre les processeurs, nous utilisons une variante de la méthode DMC proposée par R. Assaraf *et al*[72] qui permet d'utiliser une population de taille fixe. Dans cette approche, l'étape de branchement est remplacée par une étape de *reconfiguration*. On définit les poids de branchement normalisés comme

$$p_k = \frac{w_k}{\sum_{i=1}^{M} w_i} \tag{3.59}$$

et la population de marcheurs est reconfigurée en tirant à chaque pas M marcheurs parmi les M marcheurs avec la probabilité p_k.

Lorsque l'on utilise des dizaines de milliers de cœurs, une nouvelle préoccupation devient la tolérance aux pannes. En effet, si un nœud de calcul tombe en panne en moyenne au bout de cinq ans, un calcul utilisant simultanément 2000 nœuds ne

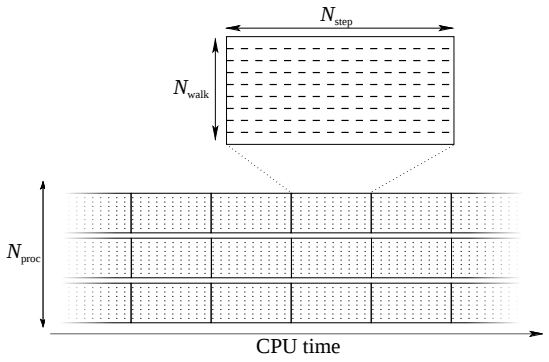


FIGURE 3.13 – Représentation d'une simulation QMC. Chaque processus génère des blocs indépendamment.

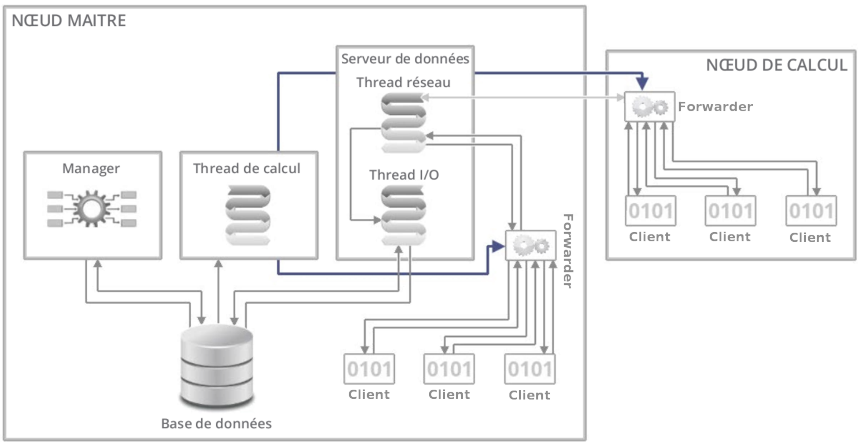


FIGURE 3.14 – Schéma fonctionnel du modèle algorithmique client/-serveur.

pourra statistiquement pas atteindre 24 heures de calcul sans panne (si les nœuds sont indépendants). On doit donc disposer d'un système qui survit coûte que coûte. Dans notre contexte stochastique, perdre des blocs au cours de la simulation ne change pas les moyennes, mais influe seulement sur la précision du résultat. J'ai donc fait le choix de ne pas utiliser de bibliothèque de type MPI pour la parallélisation, mais j'ai implémenté un modèle client/serveur avec la bibliothèque de communication asynchrone ZeroMQ[13], où les clients calculent des blocs et envoient les résultats au serveur qui les stocke dans une base de données. Si un client meurt, peu importe, les autres clients continuent à calculer. Notons que les communications étant non-bloquantes et peu intensives, les latences dues à la couche TCP ne posent aucun problème. L'utilisation de ZeroMQ permet également l'utilisation du programme au-delà du réseau des supercalculateurs, comme on a pu le montrer en 2015 en utilisant simultanément le supercalculateur Eos du méso-centre CALMIP et des ressources du cloud France Grilles.

Les clients (Figure 3.14) sont des exécutables Fortran mono-cœurs qui ne savent que calculer des blocs et envoyer les résultats au serveur. Ils sont écrits en Fortran car c'est le cœur du calcul et ce langage permet d'utiliser au mieux le jeu d'instructions du processeur. Le serveur quant à lui est un programme écrit en OCaml, car cette partie du programme ne nécessite pas d'utiliser les instructions avancées du processeur, mais doit plutôt garantir l'intégrité des données reçues par le réseau. Cela est réalisé via le typage des messages. Entre les clients et le serveur, on insère des programmes appelés *Forwarders*. Ces programmes jouent le rôle de cache entre le

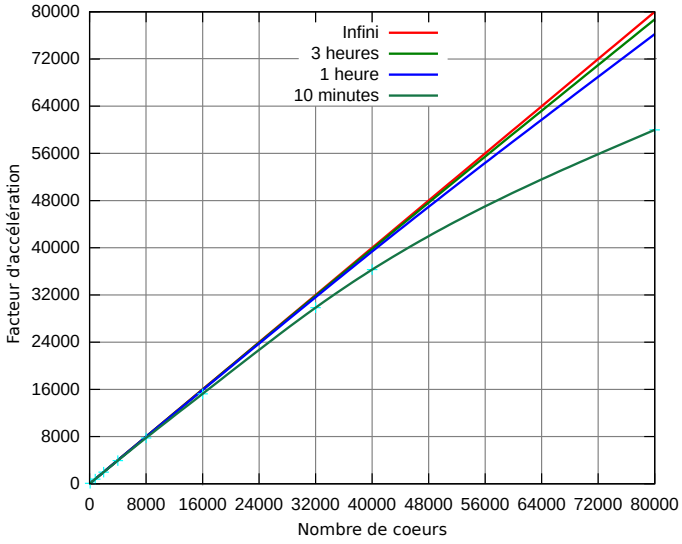


FIGURE 3.15 – Facteur d'accélération parallèle en fonction du nombre de cœurs de calcul.

serveur et les clients et permettent de réduire considérablement les communications au niveau du serveur. Sur chacun des nœuds disponibles, on lance autant de clients qu'il y a de cœurs physiques. Dès que le résultat est communiqué au Forwarder, le client commence immédiatement à calculer le bloc suivant. Simultanément, le Forwarder transmet ses résultats au serveur. Le serveur comporte deux threads principales : une thread *réseau* et une thread *I/O*. Le serveur reçoit des données par la thread réseau, les traite et les met dans une file d'attente. Simultanément la thread I/O vide la file d'attente pour mettre les résultats sur disque. Notons que les clients n'accèdent jamais au système de fichiers afin d'échapper aux pannes liées au stockage (disque plein, panne physique, bug de Lustre, etc).

Puisque les blocs sont indépendants, aucune synchronisation n'est nécessaire entre les clients, et cela permet d'obtenir un facteur d'accélération presque idéal en fonction du nombre de nœuds. Cependant, il y a deux étapes critiques qui vont nuire à l'idéalité du facteur d'accélération : l'initialisation et la terminaison. Pour avoir une initialisation rapide, tous les clients sont lancés via la commande `srun` disponible avec le gestionnaire SLURM[73]. Ainsi, chaque nœud de calcul démarre dès que possible. Pour la terminaison, étant donné que tous les nœuds de calcul sont désynchronisés, il faudrait attendre que chacun des nœuds ait fini de calculer son bloc. Dans ce cas, il peut en résulter un temps d'attente important où beaucoup de nœuds attendent que les derniers nœuds aient terminé. On ne peut pas non plus tuer violemment tous les clients, car les résultats des blocs en cours seraient perdus. Il est donc permis aux programmes de calcul d'être interrompus au cours d'un bloc et de transmettre le résultat de ce bloc au client. Ainsi, la terminaison d'un client est presque immédiate et aucune seconde de calcul n'est perdue. Pour un calcul d'un millier de nœuds, on a mesuré une terminaison de l'ordre de 10 secondes.

Tous ces éléments mis ensemble permettent d'obtenir la courbe de la Figure 3.15. Pour un calcul donné, à nombre constant de nœuds le temps d'initialisation et de terminaison est constant, donc plus le temps de calcul est long, plus l'efficacité parallèle est bonne. De même que pour le Quantum Package, il est possible d'ajouter/supprimer des ressources à tout moment.

Optimisation mono-cœur

Chaque seconde gagnée dans le programme de calcul se répercutera sur la totalité de la simulation car aucune synchronisation n'est bloquante. Ainsi, j'ai entrepris un travail important d'optimisation mono-coeur de QMC=Chem en profitant de l'expérience et des outils développés par le groupe du Prof. W. Jalby au laboratoire Exascale Computing Research (Intel-CEA-UVSQ-GENCI) à l'Université de Versailles-Saint-Quentin.

Lors d'un calcul des milliards de pas sont effectués et on doit évaluer, à chaque pas Monte Carlo, la fonction d'onde, ses dérivées par rapport à chacune des coordonnées électroniques et son laplacien. Ces opérations font intervenir des produits de petites matrices ($< 1000 \times 1000$) dont l'une est dense et l'autre est creuse.

L'outil d'analyse statique MAQAO[74] développé par nos collègues de Versailles nous a aidé à écrire une routine de produit matrice dense $\times$ vecteur creux dont les boucles les plus internes atteignent théoriquement les 16 flops/cycle en simple précision sur les processeurs Sandy Bridge d'Intel de la machine CURIE. En pratique, étant donné que nous faisons de l'ordre de N^2 opérations (N, nombre d'électrons) pour N^2 accès à la mémoire, le calcul est inévitablement limité par les accès à la mémoire et nous avons mesuré jusqu'à 61% de la performance crête du processeur dans cette routine de produit matrice $\times$ vecteur.

Lors de l'installation de la machine CURIE en décembre 2011 et grâce au soutien des ingénieurs de Bull, nous avons pu réaliser notre tout premier calcul QMC à grande échelle sur l'ensemble des cœurs de de la machine. La machine étant en cours d'installation, les calculs ont subi des interruptions et des pannes. Ce qui initialement nous avait apparu comme une gêne est devenu finalement une chance puisque le fait que, malgré ces difficultés, nos simulations aient pu être menées à bien a validé en pratique la grande robustesse de notre schéma.

Chapitre 4

Perspectives

Le Monte Carlo quantique (QMC) est une méthode prometteuse car elle est présente naturellement un fort degré de parallélisme, et elle permet également d'implémenter facilement différentes stratégies de tolérance aux pannes (inévitables à l'échelle de l'*exascale*). Toutes ces conditions proviennent de la nature *stochastique* de la méthode. Ainsi on voit émerger depuis quelques années des variantes stochastiques des méthodes traditionnelles, parmi lesquelles on peut remarquer l'implémentation stochastique de la méthode MP2,[75] de la DFT stochastique,[76] ou les méthodes basées sur l'algorithme FCI-QMC développé par le groupe d'Ali Alavi.[4] Dans cette idée, nous avons développé l'algorithme stochastique qui permet le calcul de la correction perturbative de l'énergie dans le contexte multi-références. Cette implémentation nous a permis d'utiliser plus de 4 000 cœurs avec une efficacité parallèle de 90%, et permet également de résister à la panne de certains nœuds de calcul. Cela nous permet aujourd'hui de faire des calculs qui étaient encore impossibles récemment. Ainsi, j'envisage de poursuivre dans cette direction pour proposer une variante stochastique de l'algorithme CIPSI qui permettra donc l'utilisation efficace de tout type de méthode d'interaction de configurations sur des machines massivement parallèles, mais aussi des méthodes telles que le Coupled Cluster multi-références présenté dans ce manuscrit.

Un autre point fort des méthodes QMC est qu'elles permettent d'aller au-delà de l'approximation de base finie qu'on trouve dans toutes les méthodes post-Hartree-Fock. En effet, l'unique approximation du QMC est l'erreur des nœuds fixés, et les nœuds de la fonction d'onde sont paramétrés par une fonction d'essai issue d'un calcul préalable. Ainsi, plus la fonction d'essai est proche du résultat exact, plus le calcul QMC tend vers le résultat exact. Pour essayer de contrôler cette erreur, la tendance actuelle des différents groupes QMC dans le monde est de réoptimiser les paramètres de cette fonction d'essai en présence d'un facteur explicite de corrélation (terme de Jastrow). Mais le facteur de Jastrow a une forme telle que les intégrales biélectroniques ne peuvent pas être calculées analytiquement, et seul un calcul QMC permet d'obtenir l'énergie de la fonction d'essai. Bien que cette approche fonctionne bien pour de petites molécules, je ne pense pas que cette approche soit durable car elle nécessite de réaliser une optimisation de paramètres non-linéaires dans un contexte où les dérivées sont obtenues avec du bruit statistique. Il paraît plus raisonnable de définir un protocole *déterministe* d'optimisation de fonction d'onde afin d'assurer un meilleur contrôle de l'optimisation et une meilleure cohérence de la fonction d'essai aux différents points de la surface d'énergie potentielle. Ainsi, nous avons développé le couplage de fonctions d'ondes quasi-Full-CI avec le QMC, et nous avons vu que ces fonctions d'ondes permettent un véritable contrôle de l'erreur des nœuds fixés en QMC. Cependant, même si les calculs sont réalisables aujourd'hui sur de petites

molécules, nous n'avons pas encore atteint l'objectif de pouvoir proposer une méthode générale permettant de traiter des systèmes de taille arbitraire. En effet, lorsque le nombre de fonctions de bases devient trop important, la fonction d'onde variationnelle obtenue par CIPSI peut comporter plusieurs dizaines de millions de déterminants sans pour autant être suffisamment proche de la fonction d'onde Full-CI. Il convient donc de trouver des schémas plus approchés qui contiennent la physique du système.

La première voie que j'ai empruntée porte sur la réduction de la qualité de la fonction d'essai en utilisant des fonctions d'essai de type CAS+SD. Les premiers essais sur le dimère de molybdène ont montré que la courbe de dissociation en QMC était beaucoup plus homogène avec des nœuds issus d'un calcul MR-CISD convergé plutôt que ceux d'un calcul Full-CI mal convergé. L'utilisation de la méthode Coupled-Cluster multi-références présentée précédemment a montré une amélioration des résultats par rapport aux calcul MR-CISD. Ainsi, l'utilisation de fonctions d'ondes issues de calculs MR-CCSD sélectionnés permettra d'aller vers des systèmes plus étendus que ceux que nous traitons aujourd'hui. Pour aller encore plus loin, nous avons commencé à développer une version multi-références de la méthode $(SC)^2$. Cette méthode est une approximation de la méthode MR-CCSD avec un coût de calcul réduit d'un ordre de grandeur. Les premiers résultats ont montré que les fonctions d'ondes obtenues sont extrêmement proches des celles obtenues par le MR-CCSD, ce qui est très encourageant pour obtenir des nœuds de grande qualité avec un coût de calcul raisonnable.

Parallèlement, quand on regarde l'algorithme CIPSI on peut constater qu'un grand effort numérique est réalisé pour le calcul de la contribution perturbative de déterminants qui ne sont pas sélectionnés. Puisque seulement un tout petit sous-ensemble de ces déterminants est sélectionné, la majeure partie du travail n'est pas utilisée pour la suite du calcul. En utilisant les techniques d'habillage de l'hamiltonien, il est possible d'inclure dans l'hamiltonien l'effet de la corrélation apportée par chacun des déterminants perturbateurs non-sélectionnés, en s'inspirant de la méthode Shifted-B_k.[77] Ainsi, on espère accélérer énormément la convergence des coefficients de la fonction d'onde variationnelle vers leur valeur au Full-CI pour un coût de calcul quasi-nul, ce qui permettra d'avoir des nœuds de qualité quasi-Full-CI pour des systèmes qui sont aujourd'hui trop étendus. Je prévois également de coupler cet habillage avec celui provenant de la méthode MR-CCSD ou MR-$(SC)^2$ pour permettre d'introduire plusieurs niveaux d'approximations.

Un des problèmes fondamentaux des méthodes de structures électronique est leur lente convergence avec la taille de la base atomique. Ce problème a déjà été pointé il y a trente ans par Kutzelnigg[78] qui a proposé d'introduire explicitement la corrélation électronique à travers l'utilisation de la distance $r_{12} = |\mathbf{r}_1 - \mathbf{r}_2|$ comme fonction de base. Cela a pour effet d'accélérer considérablement la convergence en base. Cette idée a ensuite été généralisée à l'utilisation de meilleurs facteurs de corrélation $f_{12} = f(r_{12})$, et les méthodes incluant un tel facteur de corrélation sont capables d'atteindre la précision chimique avec des bases de gaussiennes de taille relativement faible pour de petites molécules. Par exemple, Tew a montré au niveau CCSD(T) qu'on peut obtenir des énergies de qualité quintuple-zeta avec des bases triple-zeta.[79] Cependant, ces méthodes sont encore limitées à des systèmes de petite taille à cause de leur coût de calcul important. Mais en utilisant les mêmes techniques d'habillage que celles utilisées pour le MR-CCSD ou le MR-$(SC)^2$, il est possible d'introduire la corrélation explicite comme un habillage de l'hamiltonien pour un coût

de calcul relativement faible. Ainsi, toutes les méthodes développées dans le *Quantum Package* pourront bénéficier de facteurs de corrélation explicite. Bien entendu, ces facteurs de corrélation sont extrêmement simples à ajouter dans le programme QMC=Chem, et il auront pour effet de nous permettre de nous rapprocher des fonctions d'ondes de type Slater-Jastrow, et réduira donc les fluctuations statistiques, mais sans qu'aucune optimisation stochastique ne soit nécessaire. Ce travail sera une véritable avancée dans le domaine du QMC.

L'objectif à long terme de mon activité de recherche est de permettre aux méthodes de fonction d'onde d'utiliser au mieux les supercalculateurs du futur, permettant ainsi la progression de la discipline en dépit des contraintes matérielles. Cela nécessite une veille technologique permanente. On a vu apparaître il y a une dizaine d'années l'utilisation d'accélérateurs graphiques (Graphical Processing Units, GPU) pour réaliser des calculs. Au départ, il s'agissait d'appeler des routines OpenGL pour réaliser des produits de matrices car le résultat pouvait être obtenu plus rapidement que sur CPU.[80] Cette idée amusante s'est développée, les constructeurs de GPUs on participé au développement d'un environnement de développement permettant de programmer ces accélérateurs à partir d'un langage proche du C, et les accélérateurs graphiques sont devenus aujourd'hui de véritables accélérateurs matériels pour le calcul scientifique. Je me suis intéressé au GPU vers 2007-2008, mais l'environnement de programmation était encore très immature et j'ai jugé que l'investissement en temps et en complexité était beaucoup trop important par rapport au gain que pouvaient avoir les applications que je développais. Puis, en 2013 le calcul sur GPU était devenu plus mûre, et j'ai encadré un stagiaire de M2 pour étudier la possibilité d'utiliser des GPUs pour accélérer QMC=Chem. La conclusion de ce travail était que pour obtenir un gain de performance par rapport au CPU, il fallait absolument réécrire tout le code QMC pour qu'il tourne intégralement sur GPU. En effet, les communications entre la mémoire du CPU et la mémoire du GPU ne pouvaient pas être masquées et présentaient un véritable goulot d'étranglement. Cela revenait donc à écrire et maintenir un deuxième code QMC, ce qui n'était pas dans mes priorités. Voila pourquoi je ne mentionne aucun travail en lien avec le calcul sur GPU dans ce manuscrit, bien que ce soit une technologie que j'ai continué à suivre en attendant une sérieuse évolution en ce qui concerne le transfert des données entre le CPU et le GPU. Aujourd'hui les choses sont véritablement en train de changer. Grâce au développement de nouveaux standards tels que OpenACC ou OpenMP4, la programmation sur GPU est devenue beaucoup plus facile et portable, et la majorité des algorithmes utilisés en algèbre linéaire sont disponibles dans des bibliothèques qui s'exécutent sur le GPU. On voit aussi arriver la mémoire unifiée entre le CPU et le GPU, et il devient maintenant difficile d'ignorer le GPU en tant qu'architecture cible pour le développement de codes scientifiques. Je vais donc consacrer une partie de mon temps à porter les codes que je développe sur GPU, mais je vais aussi intégrer ce nouvel aspect dans le processus de développement des codes mais aussi des algorithmes.

Annexe A

Autres travaux

A.1 IRPF90 : un générateur de code pour le calcul scientifique

Aujourd'hui les gros codes scientifiques en Fortran deviennent difficiles à maintenir. La complexité des programmes provient des dépendances entre entités du code. Plus les entités sont inter-dépendantes, plus le programme est complexe. Pour que le code reste sous contrôle, il faut que le programmeur puisse connaître les conséquences d'une modification du code source sur tous les chemins d'exécution qui passent par le code modifié. Lorsque le code a été écrit par plusieurs développeurs et qu'il comporte des centaines de milliers de lignes, cela devient extrêmement difficile pour le programmeur. En revanche, la machine est tout-à-fait capable de faire ce travail à notre place en gérant les dépendances entre variables à la manière d'un Makefile.

IRPF90 est un générateur de code Fortran. De façon schématique, l'utilisateur n'écrit que des noyaux de calcul et IRPF90 génère le code qui va faire la liaison entre ces noyaux pour produire le résultat attendu en tenant compte des relations de dépendance entre les variables du programme. Ainsi, même les gros codes demeurent entièrement sous contrôle.

A.1.1 Présentation d'IRPF90

Un programme (ou sous-programme) scientifique est une fonction complexe de ses données. On peut représenter le programme comme un arbre dont la racine est la sortie du programme, les feuilles sont les données, les nœuds sont les variables intermédiaires et les segments représentent la relation *a besoin de*. Par exemple, le programme qui calcule $t\Big(u(d_1,d_2),v(u(d_3,d_4),w(d_5))\Big)$ avec

$$\begin{aligned} u(x,y) &= x+y+1 \\ v(x,y) &= x+y+2 \\ w(x) &= x+3 \\ t(x,y) &= x+y+4 \end{aligned}$$

peut être représenté par l'arbre de la figure A.1. L'écriture du programme en Fortran nécessiterait au programmeur d'avoir cet arbre en tête :

```
program calcule_t
  implicit none
  integer :: d1, d2, d3, d4 d5   ! Donnees d'input
  integer :: u1, u2, v, w, t     ! Variables intermediaires
  call lecture(d1,d2,d3,d4,d5)
```

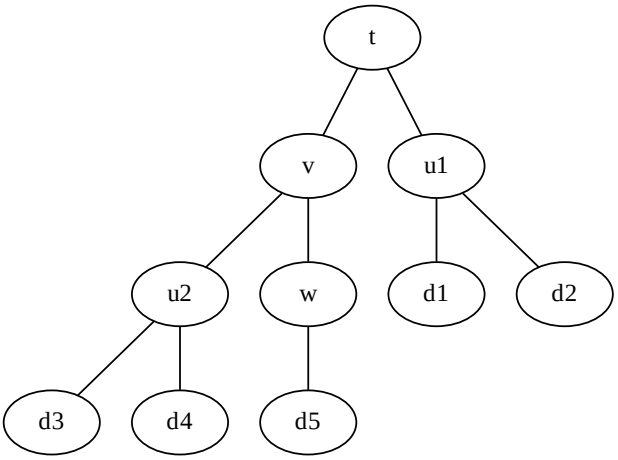


FIGURE A.1 – Représentation de $t\Big(u\big(d_1,d_2\big),v\big(u(d_3,d_4),w(d_5)\big)\Big)$ par un arbre.

```
  call calcule_u(d1,d2,u1)
  call calcule_u(d3,d4,u2)
  call calcule_w(d5,w)
  call calcule_v(u2,w,v)
  call calcule_t(u1,v,t)
  write(*,*), "t=", t
end program
```

De cette façon, le programmeur dit à la machine ce qu'elle doit faire étape par étape : on parle de programmation impérative. Si les étapes ne sont pas données dans la bonne séquence, le programme est faux. Ainsi à chaque ligne, il faut connaître l'état de l'ensemble du programme, et donc les dépendance entre les variables. Dans cette approche, la pensée du programmeur va des feuilles de l'arbre vers la racine.

Le même programme peut être réécrit en pensant différemment. Au lieu de dire à la machine ce qu'elle doit faire étape par étape, on peut plutôt lui dire ce que l'on veut. Cela revient à penser le programme en partant de la racine vers les feuilles.

```
program calcule_t
  implicit none
  integer :: d1, d2, d3, d4 d5     ! Donnees d'input
  integer :: u1, u2, v, w, t       ! Fonctions
  call lecture(d1,d2,d3,d4,d5)
  write(*,*), "t=", t( u(d1,d2), v( u(d3,d4), w(d5) ) )
end program
```

Ici, les relations de dépendances entre les variables sont exprimées à travers l'appel de la fonction `t`, et le programmeur n'a plus besoin de spécifier une séquence particulière. Il ne connaît d'ailleurs pas *a priori* l'ordre dans lequel vont s'effectuer les appels des fonctions `w(d5)` et `u(d3,d4)`, mais il doit toujours avoir l'arbre en tête pour écrire le programme.

Avec IRPF90, le programmeur n'a pas besoin de connaître l'arbre : il est automatiquement calculé. Ainsi, avec IRPF90 le programme serait tout simplement :

```
program calcule_t
  write(*,*) t
end program

BEGIN_PROVIDER [ integer, t ]
 t = u1+v+4
END_PROVIDER

BEGIN_PROVIDER [ integer, w ]
 w = d5+3
END_PROVIDER

BEGIN_PROVIDER [ integer, v ]
 v = u2+w+2
END_PROVIDER

BEGIN_PROVIDER [ integer, u1 ]
 call calcule_u(d1,d2,u1)
END_PROVIDER

BEGIN_PROVIDER [ integer, u2 ]
 call calcule_u(d3,d4,u2)
END_PROVIDER

subroutine calcule_u(x,y,u)
 integer, intent(in)  :: x,y
 integer, intent(out) :: u
 u = x+y+1
end

 BEGIN_PROVIDER [ integer, d1 ]
&BEGIN_PROVIDER [ integer, d2 ]
&BEGIN_PROVIDER [ integer, d3 ]
&BEGIN_PROVIDER [ integer, d4 ]
&BEGIN_PROVIDER [ integer, d5 ]
 call lecture(d1,d2,d3,d4,d5)
END_PROVIDER
```

De cette façon, le programmeur exprime facilement sa pensée : “Imprime `t` à l'écran.” Il n'a absolument pas besoin de savoir de quoi dépend `t`, comment `t` est calculé ou si `t` a déjà été calculé précédemment. Le programmeur veut `t` et rien de plus. Cela rend le développement collaboratif beaucoup plus simple.

Pour chaque nœud de l'arbre on écrit un *provider*, c'est-à-dire une routine dont le rôle est de construire la variable associée au nœud. Il est absolument nécessaire que la quantité soit construite correctement dans le provider, tel que lorsqu'un provider est exécuté on ait la garantie que la quantité est construite correctement. Il est possible d'ajouter des assertions qui seront vérifiées à l'exécution avec le mot-clé **ASSERT**.

```
BEGIN_PROVIDER [ integer, u2 ]
  call calcule_u(d3,d4,u2)
  ASSERT (u2 < d3)
```

```
END_PROVIDER
```

donne le résultat suivant en cas d'échec :

```
Stack trace:            0
-------------------------
provide_t
provide_v
provide_u2
u2
-------------------------
u2: Assert failed:
 file: uvwt.irp.f, line: 23
(u2 < d3)
u2 =            8
d3 =            3

STOP 1
```

IRPF90 analyse le code écrit dans tous les fichiers `*.irp.f` du répertoire courant et repère les dépendances entre les variables. On voit dans le code que le provider de `v` a besoin de `u2` et de `w‘` Ainsi, IRPF90 garantit qu'avant d'exécuter le provider de `v` les providers de `u2` et de `w` auront été exécutés, et donc que les variables `u2` et `w` seront *valides*. Cependant, le programmeur ne sait pas à quel moment exact le provider de tel ou tel nœud sera appelé. L'utilisation de `t` dans le programme principal déclenche l'exploration récursive de l'arbre avant l'impression de la valeur de `t` à l'écran. Cela revient exactement à utiliser la fonction $t\Big(u(d_1,d_2),v(u(d_3,d_4),w(d_5))\Big)$ où les paramètres des fonctions sont implicites (Implicit Reference to Parameters in Fortran 90 : IRPF90).

Dès qu'un provider a été exécuté, la variable associée est marquée comme valide. Elle ne sera donc pas reconstruite mais tout simplement réutilisée si un autre provider a besoin de la même variable.

Compilation

IRPF90 est un générateur de code Fortran 90. La gestion de l'arbre est réalisée avant la compilation et IRPF90 génère un code Fortran90 où l'exploration de l'arbre est écrite directement dans le code généré. Ainsi, toute la gestion de l'arbre est statique et ne nuit absolument pas à la vitesse d'exécution du programme qui reste du Fortran90 standard. L'arbre est recalculé avant chaque compilation car la modification du code peut induire de nouvelles dépendances entre les variables. Notons qu'il est bien entendu possible d'utiliser toutes les bibliothèques compatibles avec du Fortran (MPI, openMP, BLAS/Lapack, etc).

Il est possible d'écrire plusieurs programmes dans le même répertoire qui utilisent des providers communs. Cela est très utile pour construire des tests unitaires. Si l'on veut écrire un test pour la variable `u1`, il suffit d'écrire un programme principal qui imprime `u1` à l'écran, et seul le sous-arbre de `u1` sera généré à l'exécution.

Puisque IRPF90 connaît les dépendances entre les variables, il connaît également les dépendances entre les fichiers et peut donc écrire un `Makefile` automatiquement qui permettra de compiler tous les programmes du répertoire courant.

Modification dynamique des valeurs des nœuds

Les programmes scientifiques utilisent souvent des processus itératifs. Ceux-ci utilisent le même arbre de production à chaque itération, mais les valeurs des feuilles de l'itération $n+1$ dépendent de la valeur de la racine de l'itération n. Cela implique de pouvoir modifier la valeur d'une variable à l'extérieur de son provider, et d'informer le système de cette modification afin que les dépendances entre variables soient mises à jour et que la nouvelle racine soit construite correctement.

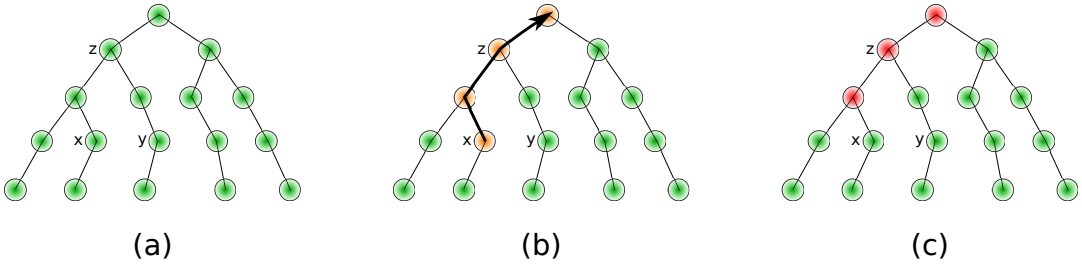


FIGURE A.2 – Invalidation de nœuds suite à l'utilisation de `TOUCH x`

Le mot-clé **TOUCH** a été introduit pour gérer les modifications en dehors des providers. Ce mot-clé rend valide la variable modifiée (touchée), mais invalide tous les nœuds qui ont besoin directement ou indirectement de cette variable. Dans l'exemple de la Figure A.2, la variable z a besoin de x et de y pour être construite. En (a), l'arbre est dans un état où tout est construit et valide : tous les nœuds sont représentés en vert. En (b), on modifie la valeur de x, et on exécute **TOUCH** x. En (c), on voit que x est valide, mais tous les nœuds entre lui et la racine ont été invalidés (en rouge). Ainsi, si l'on redemande la valeur de z, seul le nœud en rouge au dessous de z sera recalculé pour construire la nouvelle valeur de z.

Un exemple intéressant du **TOUCH** est le calcul d'une dérivée par différence finie, qui peut être utilisé pour vérifier l'implémentation de la dérivée d'une fonction. L'exemple suivant calcule la dérivée de F par rapport à x :

```
BEGIN_PROVIDER [ real, dF ]
  real :: F_p, F_m
  real, parameter :: delta_x = 0.001

  ! Calcul de F(x + 1/2.delta_x)
  x += 0.5*delta_x
  TOUCH x
  F_p = F

  ! Calcul de F(x - 1/2.delta_x)
  x -= delta_x
  TOUCH x
  F_m = F

  ! Calcul de dF
  dF = (F_p - F_m)/delta_x

  ! On remet x a sa position initiale
  x += 0.5*delta_x
  TOUCH x
END_PROVIDER
```

Variables tableaux

Un tableau est considéré comme valide lorsque toutes ses valeurs ont été calculées. Les dimensions des tableaux sont soit des variables qui ont des providers, soit des constantes, soit des intervalles.

```
BEGIN_PROVIDER [ integer, fact_max ]
  fact_max = 10
END_PROVIDER

BEGIN_PROVIDER [ double precision, fact, (0:fact_max) ]
  implicit none
  integer :: i
  fact(0) = 1.d0
  do i=1,fact_max
    fact(i) = fact(i-1)*dble(i)
  end do
END_PROVIDER
```

Dans cet exemple, puisque le tableau `fact` dépend de sa variable de dimensionnement `fact_max`, la modification de la dimension du tableau à travers **`TOUCH`** `fact_max` invalidera le tableau `fact`, et il sera recalculé avec la bonne dimension à sa prochaine utilisation. Toutes les allocations sont vérifiées et un message d'erreur apparaît à l'exécution en cas d'impossibilité d'allocation du tableau.

La mémoire réservée pour un tableau peut être libérée en utilisant le mot-clé `FREE`. Par exemple :

```
BEGIN_PROVIDER [ double precision, table2, (size(table1,1)) ]
  implicit none
  table2(:) = 2.d0 * table1(:)
  FREE table1
END_PROVIDER
```

Lorsque `table1` est libéré, le nœud correspondant est marqué comme invalide. Ainsi, si `table1` est redemandé plus tard, il sera préalablement réalloué et reconstruit.

Documentation du code

À l'intérieur de chaque provider, il est recommandé d'écrire quelques lignes de documentation pour décrire la variable construite.

```
BEGIN_PROVIDER [ double precision, fact, (0:fact_max) ]
  implicit none

  BEGIN_DOC
!  Computes an array of fact(n)
  END_DOC

  integer :: i
  fact(0) = 1.d0
  do i=1,fact_max
    fact(i) = fact(i-1)*dble(i)
  end do
END_PROVIDER
```

```
$ irpman fact
IRPF90 entities(1)                fact                IRPF90 entities(1)

Declaration
       double precision, allocatable :: fact   (0:fact_max)

Description
       !  Computes an array of fact(n)

File
       fact.irp.f

Needs
       fact_max

Needed by
       binom
       exponential_series

IRPF90 entities                   fact                IRPF90 entities(1)
```

FIGURE A.3 – Exemple de sortie de la commande `irpman`.

Lors de la compilation, la liste de toutes les variables est écrite dans un fichier nommé `irpf90_entities`, et une *man page* est crée pour chaque variable. Cette page contient la documentation présente dans le bloc de documentation, mais aussi quelles sont les variables nécessaires et quelles variables ont besoin de la variables courante. Ces pages sont accessibles avec l'outil `irpman` (figure A.3).

Templates

Il arrive parfois que l'on doive écrire plusieurs morceaux de code qui utilisent un même schéma. En C++ par exemple on utiliserait des patrons de fonctions, de classes ou d'expressions. Prenons un exemple où l'on veut créer plusieurs providers et fonctions semblables :

```
BEGIN_TEMPLATE

BEGIN_PROVIDER [ $type , $name ]
 call find_in_input('$name', $name)
END_PROVIDER

logical function $name_is_zero()
  $name_is_zero = ($name == 0)
end function

SUBST [ type, name ]

  integer     ;   size_tab1 ;;
  integer     ;   size_tab2 ;;
  real        ;   distance  ;;
```

```
  real        ;   x          ;;
  real        ;   y          ;;
  real        ;   z          ;;

END_TEMPLATE
```

Cet exemple génère automatiquement un provider et une fonction `*_is_zero` pour chaque couple apparaissant au dessous du mot-clé **SUBST**.

Interaction avec le shell et metaprogrammation

On peut avoir envie d'insérer au milieu de son programme le résultat d'une commande du shell exécutée à la compilation. Prenons un cas typique où l'on souhaite que le programme imprime à l'écran la date de compilation et la version de git à laquelle il correspond :

```
subroutine print_git_log
  write(*,*) , '----------------'
  BEGIN_SHELL [ /bin/bash ]
     echo "write(*,*)" \'Compiled␣by␣$(whoami)␣on␣$(date)\'
  END_SHELL
  write(*,*) , 'Last␣git␣commit:'
  BEGIN_SHELL [ /bin/bash ]
    git log -1 | sed "s/'//g" | \
     sed "s/^/␣␣␣␣write(*,*)␣'/g" | sed "s/$/'/g"
  END_SHELL
  write(*,*) , '----------------'
end
```

L'insertion de sorties de scripts dans le code permet également d'aller au delà des templates, et de générer du code de façon mécanique. Voici un exemple qui génère des fonctions particulières pour calculer x^n en utilisant un minimum de multiplications :

```
BEGIN_SHELL [ /usr/bin/python ]

POWER_MAX = 20

def compute_x_prod(n,d):
  if n == 0:
    d[0] = None
    return d
  if n == 1:
    d[1] = None
    return d
  if n in d:
    return d
  m = n/2
  d = compute_x_prod(m,d)
  d[n] = None
  d[2*m] = None
  return d

def print_function(n):
```

```
  keys = compute_x_prod(n,{}).keys()
  keys.sort()
  output = []
  print "real␣function␣power_%d(x1)"%n
  print "␣real,␣intent(in)␣::␣x1"
  for i in range(1,len(keys)):
    output.append( "x%d"%keys[i] )
  print "␣real␣::␣"+',␣'.join(output)
  for i in range(1,len(keys)):
    ki = keys[i]
    ki1 = keys[i-1]
    if ki == 2*ki1:
      print "␣x%d"%ki + "␣=␣x%d␣*␣x%d"%(ki1,ki1)
    else:
      print "␣x%d"%ki + "␣=␣x%d␣*␣x1"%(ki1)
  print "␣power_%d␣=␣x%d"%(n,n)
  print "end"

for i in range(POWER_MAX):
  print_function(i+1)
  print ''

END_SHELL
```

Voici un échantillon du code généré :

```
real function power_1(x1)
  real, intent(in) :: x1
  real ::
  power_1 = x1
end

real function power_2(x1)
  real, intent(in) :: x1
  real :: x2
  x2 = x1 * x1
  power_2 = x2
end
   ...
real function power_20(x1)
  real, intent(in) :: x1
  real :: x2, x4, x5, x10, x20
  x2 = x1 * x1
  x4 = x2 * x2
  x5 = x4 * x1
  x10 = x5 * x5
  x20 = x10 * x10
  power_20 = x20
end
```

A.1.2 IRPF90 pour le HPC

Vectorisation

IRPF90 offre certaines possibilités pour améliorer facilement les performances des programmes. Il peut générer automatiquement des directives du compilateur Intel pour aider la vectorisation. En effet, pour que le compilateur génère une instruction vectorielle, il est nécessaire que les opérandes soient alignées sur une frontière de 16, 32, ou 64 octets selon le jeu d'instructions. Une boucle Fortran traditionnelle va générer trois boucles :

— une boucle d'"épluchage" (peel loop)
— une boucle vectorielle
— une boucle de reste (tail loop)

La peel loop est une boucle qui traite les éléments de départ jusqu'à la frontière d'alignement, et la boucle de reste traite les derniers éléments.

Prenons la boucle suivante que l'on voudra vectoriser avec des instructions AVX (32 octets), où `nmax = 26`, et le tableau `a` est de type double précision.

```
double precision :: a(nmax)
do i=1,nmax
  a(i) = 2. * a(i)
end do
```

La boucle scalaire va faire 26 tours. Si l'alignement du tableau est tel que le premier élément du tableau est à 8 octets après la frontière, la peel loop va effectuer trois tours pour atteindre la frontière de 32 octets. Puis, la boucle vectorisée va prendre cinq fois les éléments suivants de `a` quatre par quatre, jusqu'au 23ième élément inclus. Ensuite, la boucle de reste va traiter les trois éléments restants. Avec la vectorisation, le nombre de tours devient $3 + 5 + 3 = 11$ tours. Dans ce cas, la vectorisation fait gagner un facteur 2.36.

Si maintenant le tableau est aligné sur 32 octets avec la directive suivante (pour le compilateur Intel) :

```
!DIR$ ATTRIBUTES ALIGN : 32 :: a
```

la peel loop ne sera pas générée, la boucle vectorielle fera 6 tours, et les la éléments 25 et 26 seront traités avec boucle de reste. On aura don $6 + 2 = 8$ la tours, ce qui correspond a un gain de 3.25.

Enfin, on peut aussi forcer la vectorisation complète de la boucle de la façon suivante :

```
double precision :: a(nmax+4)
!DIR$ ATTRIBUTES ALIGN : 32 :: a
do i=1, 1+(nmax-1)/4, 4
  !DIR$ VECTOR ALIGNED
  a(i:i+3) = 2. * a(i:i+3)
end do
```

Ici, les boucles implicites `a(i:i+3)` seront vectorisées, et on aura au total 7 tours, soit un gain de 3.71.

Cet exemple montre deux aspects importants (pour les boucles qui font peu de tours) :

1. L'alignement des tableaux ne peut qu'améliorer l'efficacité de la vectorisation

2. Pour pouvoir vectoriser à 100% une boucle, il faut connaître la largeur des vecteurs pour les instructions vectorielles (16 octets pour SSE, 32 octets pour AVX, etc), et donc écrire un code spécifique pour l'architecture visée.

Il est possible de demander à IRPF90 de générer les directives d'alignement pour *toutes* les variables tableaux en utilisant par exemple :

```
irpf90 --align=32
```

Génération de code spécifique

Plus le compilateur Fortran a d'informations, plus il peut produire un code efficace. IRPF90 possède une option qui permet de substituer des variables du programme par des valeurs dans les bornes des boucles et dans les conditions (**if**). Ce type d'optimisation fait gagner en général entre 5 et 15% sur un vrai code, mais bien entendu le programme donnera des résultats faux si les données ne correspondent pas a ce qui a été spécifié à la compilation.

Reprenons la boucle scalaire de l'exemple précédent :

```
double precision :: a(nmax)
!DIR$ ATTRIBUTES ALIGN : $IRP_ALIGN :: a

if (calculate_loop) then
  do i=1,nmax
    a(i) = 2. * a(i)
  end do
end if
```

et utilisons :

```
irpf90 --align=32 --substitute nmax:26 --substitute calculate_loop:.True.
```

le code produit sera

```
double precision :: a(nmax)
!DIR$ ATTRIBUTES ALIGN : 32 :: a

if (.True.) then
  do i=1,26
    a(i) = 2. * a(i)
  end do
end if
```

Le compilateur Fortran va supprimer le test qui est toujours vrai, et produire le binaire optimal pour une boucle a 26 tours. Si l'on utilise :

```
irpf90 --align=32 --substitute nmax:26 --substitute calculate_loop:.False.
```

la condition étant toujours fausse le compilateur Fortran va supprimer tout le code contenu dans le if.

Profiling

Quand on fait développement pour le HPC, il est absolument nécessaire de pouvoir isoler la portion de code que l'on cherche à optimiser. Étant donné que l'arbre du programme est géré par IRPF90, il peut générer automatiquement des *codelets* pour chacun des nœuds de l'arbre :

```
irpf90 --codelet=tab2:10000
```

va construire un programme principal qui va appeler le provider de `tab2` 10000 fois, et mesurer le nombre de cycles et le temps en secondes nécessaire pour construire `tab2`. Si l'on souhaite profiler tout le code, il suffit d'ajouter l'option `-g` et un rapport de profiling apparaîtra dans la sortie standard après l'exécution du programme.

A.1.3 Résumé

Écrire un programme revient à écrire les providers des variables impliquées. Lorsqu'on écrit un provider pour une variable `x`, les seules questions que l'on se pose sont :

1. Comment est-ce que je construis `x` ?
2. Quelles sont les noms des variables dont j'ai besoin ?
3. Suis-je bien sûr que `x` est valide à la sortie du provider ?

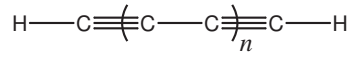
La programmation quotidienne est facilitée :
— La séquence d'exécution n'a pas besoin d'être connue
— Les `makefiles` sont construits automatiquement
— Lorsqu'on utilise une quantité, on est absolument sûr qu'elle est a déjà été construite correctement
— Écrire un nouveau provider ne va jamais nuire au reste du code : on ne risque pas de "casser" un programme qui fonctionnait
— Plusieurs personnes peuvent travailler simultanément sur le même code avec un minimum d'effort
— Si une variable a déjà été construite, elle ne sera jamais reconstruite si cela n'est pas nécessaire.
— La construction de tests unitaires et codelets pour le travail d'optimisation consiste simplement à créer un nouveau programme principal.

A.2 Travail de thèse : Réactivité en milieu atmosphérique et analyse Monte Carlo quantique de la localisation électronique

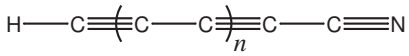
J'ai réalisé ma thèse au Laboratoire de Chimie Théorique (Paris VI) sous la direction de Patrick Chaquin. À l'origine, ce travail de thèse, réalisé dans le cadre de la mission Cassini-Huygens pour l'étude de l'atmosphère de Titan, avait une vocation principalement applicative. Lex expérimentateurs effectuant des mesures spectroscopiques au Laboratoire Interuniversitaire des Systèmes Atmosphériques de l'Université Paris XII (LISA), nous ont proposé d'étudier les polyynes (figure A.4), car leur manipulation expérimentale est extrêmement délicate sur Terre. En effet, ces molécules sont très fortement réactives à l'air, même lorsque l'oxygène est présent en quantité infinitésimale.

Dans ce travail, une partie des calculs a été effectuée avec les methodes usuelles pour traiter la corrélation (DFT, CAS-SCF, Coupled-Cluster, ...). En revanche, pour l'étude de certains systèmes ces méthodes se sont révélées insuffisantes. L'utilisation de méthodes Monte Carlo quantique (QMC) s'est avérée nécessaire, et une grande partie du travail a été consacrée au développement de méthodes de localisation électronique adaptées aux données QMC pour permettre la visualisation de la prise en compte des effets de corrélation électronique.

Polyynes :



Monocyanopolyynes :



Dicyanopolyynes :

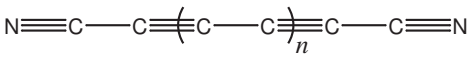


FIGURE A.4 – Représentation des polyynes, monocyanopolyynes et dicyanopolyynes.

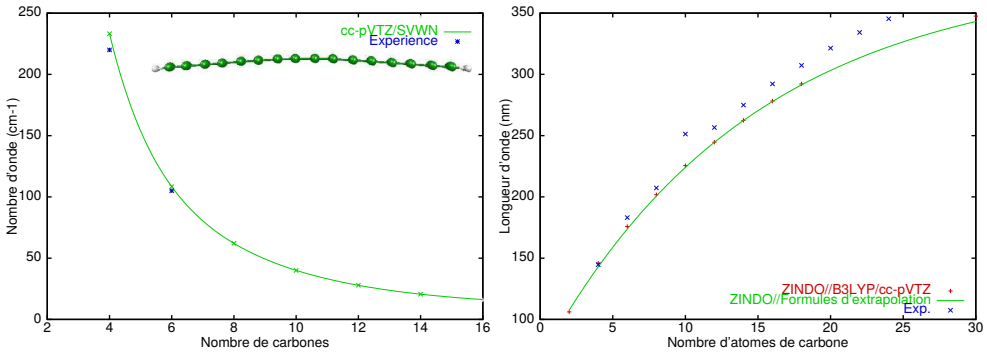


FIGURE A.5 – Nombre d'onde du mode de déformation antisymétrique en fonction du nombre de carbones (à gauche), et longueur d'onde d'absorption en fonction du nombre de carbones (à droite).

A.2.1 Propriétés chimiques des polyynes

Structure

J'ai commencé par proposer des formules d'extrapolation pour les longueurs de liaisons. Les longueurs de liaisons simples et triples ne tendent pas vers la même limite, donc l'alternance de longueur de liaison est préservée pour une chaîne de longueur infinie. Ce résultat a été complété par une analyse topologique de la fonction de localisation électronique (ELF), qui a montré une différence qualitative significative entre les liaisons simples et triples dans les polyynes jusqu'à $C_{16}H_2$.[81]

Spectroscopie

Une conclusion importante a été tirée de nos calculs en spectroscopie infrarouge : le mode de vibration le plus approprié pour identifier les polyynes dans les atmosphères planétaires ou dans le milieu interstellaire n'est pas le mode qui possède l'absorption la plus intense, mais plutot le mode de déformation antisymétrique de la molécule (figure A.5). La convergence de la valeur du nombre d'onde en fonction du nombre d'atomes de carbone n'est toujours pas atteinte pour $C_{16}H_2$, donc il doit être possible de différencier expérimentalement $C_{14}H_2$ de $C_{16}H_2$ en observant ce mode.[82]

Nos formules d'extrapolation pour les longueurs de liaison nous ont permis de calculer des énergies d'excitation électronique avec la méthode semi-empirique ZINDO sur des géométries très proches des géométries optimisées avec des calculs DFT (figure A.5) pour des chaînes possédant jusqu'à 30 atomes de carbone.[83] J'ai pu montrer que l'énergie d'excitation HOMO-LUMO tend vers une limite non-nulle. Ainsi, le polyyne en tant que forme allotropique du carbone est un isolant.

Réactivité

La présence de composés cycliques dans les atmosphères simulées par le LISA et la population du mode de déformation de très basse fréquence à basse température ont orienté le projet vers une étude systématique des produits de cyclisation des polyynes. J'ai commencé par étudier la structure électronique de la molécule C_4H_2 cyclique. Cette molécule n'est pas un cyclobutatriène mais un dicarbène singulet, relativement stable.[84] Puis nous avons étudié ses réactions hypothétiques de formation et d'isomérisation (figure A.6) à partir de molécules de l'atmosphère de Titan. Sa formation à partir du butadiyne n'est pas la voie privilégiée à cause de la forte énergie d'activation. En revanche, sa formation à partir de l'addition de C_2 sur l'acétylène peut se faire sans activation.

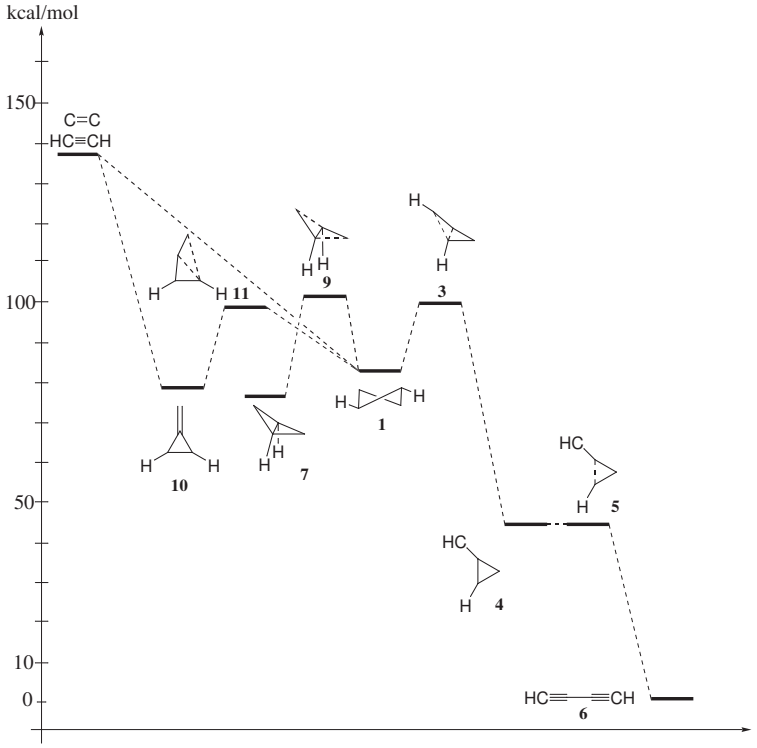


FIGURE A.6 – Réactions de formation du butadiyne cyclique.

Par la suite, nous avons essayé d'étendre notre étude à l'homologue supérieur C_6H_2, mais les méthodes usuelles ne nous permettaient pas d'identifier clairement l'état fondamental. Michel Caffarel et Roland Assaraf de l'équipe Monte Carlo quantique (QMC) du laboratoire nous ont donc proposé d'utiliser les méthodes QMC.

A.2.2 Développement de méthodes de localisation électronique en QMC

Ne possédant pas d'outil adapté à la visualisation des distributions électroniques en QMC, l'interprétation des résultats s'avérait difficile. J'ai donc entrepris le développement d'une méthode qualitative permettant de décrire l'appariement électronique (EPLF), et d'une méthode quantitative basée sur les distributions de probabilités à partir de densités issues de calculs QMC.

La fonction de localisation de paires électroniques (EPLF)

La *fonction de localisation de paires électroniques*[85] (EPLF) est positive dans les régions où les électrons de spins opposés sont appariés, négative dans les régions où les électrons de même spin sont appariés, et proche de zéro où les électrons ne sont pas appariés. Pour la définir, introduisons d'abord deux quantités locales $d_{\sigma\sigma}(\mathbf{r})$ et $d_{\sigma\bar{\sigma}}(\mathbf{r})$:

$$d_{\sigma\sigma}(\mathbf{r}) = \left\langle \sum_{i=1}^{N} \delta(\mathbf{r}-\mathbf{r}_i) \min_{j;\sigma_j=\sigma_i} |\mathbf{r}_i - \mathbf{r}_j| \right\rangle \tag{A.1}$$

$$d_{\sigma\bar{\sigma}}(\mathbf{r}) = \left\langle \sum_{i=1}^{N} \delta(\mathbf{r}-\mathbf{r}_i) \min_{j;\sigma_j\neq\sigma_i} |\mathbf{r}_i - \mathbf{r}_j| \right\rangle \tag{A.2}$$

$$\tag{A.3}$$

où $\{\mathbf{r}_k\}_{k=1,N}$ sont les positions des N électrons pour une configuration donnée, σ_i est le spin de l'électron i (α ou β), et $\langle \ldots \rangle$ est la moyenne sur les configurations Monte Carlo. Comme on peut le voir d'après les définitions, $d_{\sigma\sigma}(\mathbf{r})$ (respectivement $d_{\sigma\bar{\sigma}}(\mathbf{r})$) est la distance moyenne entre un électron positionné en $\mathbf{r}$ et l'électron de même spin (respectivement de spin opposé) le plus proche. On définit l'EPLF comme :

$$\mathrm{EPLF}(\mathbf{r}) = \frac{d_{\sigma\sigma}(\mathbf{r}) - d_{\sigma\bar{\sigma}}(\mathbf{r})}{d_{\sigma\sigma}(\mathbf{r}) + d_{\sigma\bar{\sigma}}(\mathbf{r})} \tag{A.4}$$

Nous avons d'abord vérifié que la structure de couches des atomes est bien décrite, puis nous avons illustré cette nouvelle méthode avec des calculs sur des molécules très simples (figure A.7).

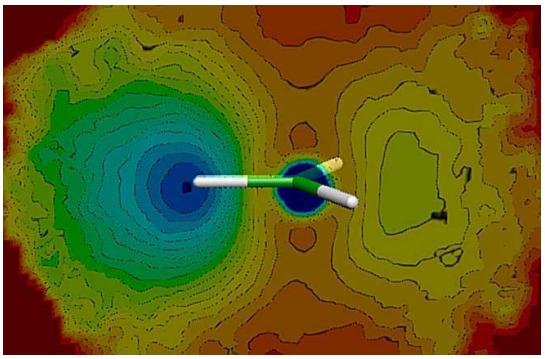


FIGURE A.7 – Fonction EPLF dans le plan du radical CH_3 : les valeurs sont élevées dans les liaisons C–H et dans le cœur, et les valeurs sont proches de zéro où l'électron célibataire est localisé.

Localisation électronique avec les distributions de probabilités

Andreas Savin travaillait au laboratoire en collaboration avec Eric Cancès du CERMICS à Marne-la-vallée sur la recherche de volumes maximisant la probabilité trouver ν électrons dans le volume et $N-\nu$ électrons hors du volume :

$$P_\Omega(\nu) = \frac{N!}{\nu!\,(N-\nu)!} \int_\Omega \mathrm{d}1\mathrm{d}2\ldots\mathrm{d}\nu \int_{\bar{\Omega}} \mathrm{d}(\nu+1)\ldots\mathrm{d}N\,|\Psi|^2 \tag{A.5}$$

Leurs applications étant limité aux molécules linéaires et aux fonctions d'ondes monodéterminantales (Hartree-Fock ou DFT), A. Savin m'a proposé de développer la méthode pour des densités issues de calculs Monte Carlo quantique, et de concevoir un algorithme de recherche du volume dans l'espace à trois dimensions afin d'étudier tout type de molécules avec une densité très corrélée.[86, 87]

Cette méthode nous a permis de proposer une nouvelle partition de l'espace (figure A.8), mais aussi d'analyser la densité électronique à N particules. Nous avons également pu donner des poids aux structures covalentes et ioniques dans la molécule de difluor après un calcul Diffusion Monte Carlo, prenant en compte près de 90% de l'énergie de corrélation. Ces poids sont en parfait accord avec les calculs de type Valence Bond de la littérature.

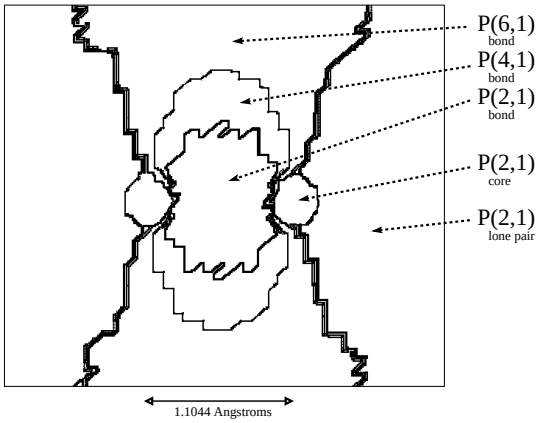


FIGURE A.8 – Coupe dans le plan des volumes maximisant la probabilité de trouver 2, 4 et 6 électrons autour de la molécule de diazote.

Résultats

Ces deux nouvelles méthodes sont définies quelque soit la fonction d'onde (couches ouvertes, ondes planes, VMC, DMC, DFT, MCSCF, Valence Bond, Coupled Cluster, semi-empiriques, *etc.*...) puisque toute densité peut être échantillonnée par l'algorithme de Metropolis. L'utilisation des ces deux méthodes m'a permis d'élucider la structure électronique du produit de cyclisation de l'hexatriyne C_6H_2. Les quatre atomes ne portant pas d'hydrogène possèdent chacun un électron célibataire, et ces électrons ont un couplage anti-ferromagnétique relativement faible.

A.3 Travail post-doctoral : MCSCF en présence d'une fonction de corrélation de Jastrow

J'ai effectué mon premier séjour post-doctoral dans le groupe de Claudia Filippi à Leiden (Pays-Bas). Son travail de recherche se focalise sur l'étude de chromophores biologiques car ce sont des systèmes qui se placent à la fois dans le domaine des systèmes de grande taille et dans celui du calcul d'états excités ayant un fort caractère multi-configurationnel. Les méthodes standard (CAS-PT2, MR-CI, *etc*) demandent des calculs extrêmement lourds pour obtenir une précision de qualité suffisante pour ces types de systèmes pouvant contenir une centaine d'électrons de valence. Nous avons proposé une approche[52] construite en alliant les avantages des méthodes *ab initio* (prise en compte de la corrélation non-dynamique, calculs MCSCF) et ceux des méthodes de type Monte Carlo quantique (corrélation dynamique).

A.3.1 Nature de la fonction d'onde

De manière générale, les méthodes *ab-initio* traitent la corrélation électronique par une expansion de la fonction d'onde sur une base de déterminants de Slater. Cette expansion doit traiter à la fois la corrélation statique et la corrélation dynamique. Nous avons choisi de traiter explicitement la corrélation dynamique par un facteur de corrélation de Jastrow. Ainsi, l'expansion en déterminants est réduite de plusieurs ordres de grandeur, car elle ne décrit que les phénomènes de quasi-dégénérescence.

Nous partons d'une fonction d'essai Ψ de la forme suivante :

$$\Psi = \mathcal{J}\Phi \tag{A.6}$$

où $\mathcal{J}$ est le facteur de Jastrow décrivant explicitement la corrélation et Φ est une expansion sur une base de déterminants de Slater :

$$\Phi = \sum_{i}^{N_{\text{det}}} c_i D_i \tag{A.7}$$

D_i sont les déterminants et c_i sont les coefficients de l'interaction de configurations (IC). Une forme minimale typique pour le facteur de Jastrow est :

$$\mathcal{J} = \exp\left[\sum_{i}^{N_{\text{elec}}}\sum_{j>i}^{N_{\text{elec}}} U(r_{ij}) - \sum_{i}^{N_{\text{elec}}}\sum_{M}^{N_{\text{nucl}}} U_M(r_{iM})\right] \tag{A.8}$$

U_M est le terme de pénétration électron-noyau qui permet essentiellement de corriger la densité, et $U(r_{ij})$ est le terme de cusp électron-électron :

$$U(r_{ij}) = a_\sigma \frac{r_{ij}}{1 + b_\sigma r_{ij}} \tag{A.9}$$

Au niveau diffusion Monte Carlo (DMC), la seule erreur commise par rapport au résultat exact non-relativiste est due à l'approximation des nœuds fixés. Le facteur de Jastrow $\mathcal{J}$ étant une fonction positive, la surface nodale est imposée par la composante déterminantale Φ de la fonction d'onde Ψ. La prise en compte de la corrélation dynamique par le facteur de Jastrow nous permet de réoptimiser la partie déterminantale au niveau Monte Carlo variationnel, dans le but d'améliorer

les nœuds. Remarquons qu'il s'agit d'une optimisation non-linéaire dans un contexte stochastique.

A.3.2 Optimisation des coefficients de l'IC

La fonction d'onde à optimiser est

$$\Psi = \mathcal{J}\Phi = \sum_{i}^{N_{\text{det}}} c_i \mathcal{J} C_i \tag{A.10}$$

où les C_i sont des configurations de spin adapté (CSFs) qui sont des combinaisons linéaires de déterminants de Slater. Les coefficients de l'IC sont déterminés variationnellement par minimisation de l'énergie :

$$E = \frac{\langle \Psi | \widehat{H} | \Psi \rangle}{\langle \Psi | \Psi \rangle} = \frac{\langle \mathcal{J}\Phi | \widehat{H} | \mathcal{J}\Phi \rangle}{\langle \mathcal{J}\Phi | \mathcal{J}\Phi \rangle} \tag{A.11}$$

Cette minimisation est la résolution usuelle du système d'équations de l'interaction de configurations, mais la présence du facteur de Jastrow implique que la matrice Hamiltonienne et la matrice de recouvrement soient échantillonnés :

$$\sum_{j=1}^{N_{\text{CSF}}} c_j \langle \mathcal{J} C_i | \mathcal{H} | \mathcal{J} C_j \rangle = E \sum_{j=1}^{N_{\text{CSF}}} c_j \langle \mathcal{J} C_i | \mathcal{J} C_j \rangle \tag{A.12}$$

$$\sum_{j=1}^{N_{\text{CSF}}} c_j \mathbf{H}_{ij} = E \sum_{j=1}^{N_{\text{CSF}}} c_j \mathbf{S}_{ij} \tag{A.13}$$

puis la matrice $\mathbf{S}^{-1}\mathbf{H}$ est diagonalisée et les vecteurs propres sont les coefficients optimaux pour chaque état. Notons que la qualité du résultat dépend de la statistique de l'échantillonnage des matrices.

A.3.3 Optimisation des orbitales moléculaires

L'optimisation des orbitales moléculaires est réalisée en utilisant un développement super-CI de la fonction d'onde : l'espace de référence est augmenté avec toutes les simples excitations possibles de l'espace actif vers un ensemble d'orbitales externes, engendrant l'expansion suivante de la composante déterminantale,

$$\Phi' = \sum_i c_i \left(C_i + \sum_{j}^{\text{occ}} \sum_{l}^{\text{virt}} t_{jl} C_i^{j \to l} \right) \tag{A.14}$$

où $\sum_i c_i C_i$ est une petite expansion MCSCF et les $C_i^{j \to l}$ sont les simples excitations. Le théorème de Brillouin-Levy-Berthier nous indique que lorsque les orbitales sont optimales, les amplitudes t_{jl} sont nulles.

En pratique, le nombre de simples excitations croît très rapidement avec la taille de l'espace de référence et avec le nombre d'orbitales externes. L'échantillonnage de la matrice d'IC devient trop coûteuse en temps de calcul, alors nous utilisons une méthode itérative, basée sur une simplification de la méthode du potentiel de fluctuation de l'énergie.

Méthode du potentiel de fluctuation de l'énergie (EFP)

L'équation de Schrödinger $\widehat{H}\Psi = E\Psi$ peut être ré-exprimée comme

$$\widehat{H}\mathcal{J}\Phi = E\mathcal{J}\Phi \qquad \longrightarrow \qquad [\mathcal{J}^{-1}\widehat{H}\mathcal{J}]\Phi = E\Phi \tag{A.15}$$

L'idée principale de la méthode EFP[48, 49, 50] est la suivante : si Ψ est un état propre de l'hamiltonien, la composante déterminantale Φ est un état propre de l'opérateur $\mathcal{J}^{-1}\widehat{H}\mathcal{J}$ avec la même valeur propre. En partant d'un hamiltonien effectif d'essai $\widehat{H}_{\text{eff}}$ qui approxime $\mathcal{J}^{-1}\widehat{H}\mathcal{J}$, on modifie itérativement cet hamiltonien effectif pour améliorer Φ.

Considérons les variations infinitésimales de Ψ par rapport aux paramètres variationnels α :

$$\Psi' = \Psi + \sum_k \delta_k \frac{\partial\Psi}{\partial\alpha_k} = \Psi\Big(1 + \sum_k \delta_k O_k\Big) \tag{A.16}$$

où

$$O_k = \frac{1}{\Psi}\frac{\partial\Psi}{\partial\alpha_k} \tag{A.17}$$

L'énergie est stationnaire si ces dérivées sont nulles :

$$\frac{\partial E}{\partial\delta_k}\bigg|_{\delta=0} = 2\langle(E_L - \bar{E})(O_k - \bar{O}_k)\rangle_{\Psi^2} = 0 \tag{A.18}$$

avec $\bar{E} = \langle E\rangle_{\Psi^2}$ et $\bar{O}_k = \langle O_k\rangle_{\Psi^2}$. Lorsque l'énergie est stationnaire, les fluctuations de E_{L} et O_k sont décorrélées : on ne peut pas rendre l'énergie 'plus constante' en ajoutant une combinaison de fonctions O_k.

Le problème de la minimisation de l'énergie peut être reformulé comme un ajustement aux moindres carrés des fluctuations de l'énergie locale E_L par les fonctions O_k :

$$\chi^2 = \left\langle\left(E_L - \bar{E} - \sum_k V_k O_k\right)^2\right\rangle_{\Psi^2} = \left\langle\left(\frac{\widehat{H}_{\text{eff}}\Phi}{\Phi} - E_0 - \sum_k V_k O_k\right)^2\right\rangle_{\Psi^2} \tag{A.19}$$

et la minimisation de χ^2 par rapport aux paramètres V_k nous amène à la résolution d'un système d'équations linéaires :

$$\langle\Delta E\Delta O_m\rangle_{\Psi_T'^2} = \sum_k V_k\langle\Delta O_k\Delta O_m\rangle_{\Psi_T'^2} \tag{A.20}$$

où $\Delta E = E_L - \bar{E}$ et $\Delta O_m = O_m - \bar{O}_m$.

L'hamiltonien effectif de départ, développé sur les états propres :

$$\widehat{H}_{\text{eff}} = \sum_k E_k|\Phi_k\rangle\langle\Phi_k| \tag{A.21}$$

peut être amélioré en utilisant les paramètres V_k :

$$\widehat{H}'_{\text{eff}} = \widehat{H}_{\text{eff}} + \sum_k V_k\left(|\Phi_k\rangle\langle\Phi_0| + |\Phi_0\rangle\langle\Phi_k|\right) \tag{A.22}$$

Le nouvel hamiltonien $\widehat{H}'_{\text{eff}}$ est diagonalisé et l'on obtient de nouveaux états propres $\{\Phi'_k\}$. Cette procédure est itérée jusqu'à ce que $\{V_k = 0\}$.

Approche perturbative

La méthode EFP, suivie de calculs DMC, s'est montrée très efficace pour des petits systèmes tels que la formaldimine, reproduisant une surface d'energie potentielle d'un état excité avec une qualité égale à celle des calculs MR-CI[50]. Mais le calcul devient trop coûteux pour des systèmes de grande taille, donc nous proposons une approche permettant d'éviter le calcul, stockage et la diagonalisation de l'hamiltonien effectif.

Les corrections de l'hamiltonien effectif V_k peuvent être estimées par un développement en perturbations au premier ordre. Si les énergies des états peuvent être approximées, la composante déterminantale de la fonction d'onde peut être directement améliorée de la façon suivante :

$$\Phi_0' = \Phi_0 - \sum_{k\neq 0} \frac{V_k}{E_k - E_0}\Phi_k \tag{A.23}$$

Exemple : Un déterminant à couches fermées

La fonction d'onde est $\Psi_T = \mathcal{J}D_0$. L'hamiltonien effectif de départ $\widehat{H}_{\text{eff}}^{(0)}$ est choisi comme l'Hamiltonien de Hartree-Fock :

$$\widehat{H}_{\text{eff}}^{(0)} D_0 = E_0^{(0)} D_0 \tag{A.24}$$

En utilisant l'expansion super-CI, les M orbitales occupées φ_i peuvent être améliorées en les combinant avec les $N - M$ orbitales virtuelles φ_l grâce aux déterminants simplement excités $D_0^{i\to l}$:

$$\begin{aligned}\Psi' &= \mathcal{J}\left(D_0 + \sum_j^{\text{occ}}\sum_l^{\text{virt}} t_{jl} D_0^{j\to l}\right) \\ &= \Psi\left(1 + \sum_j^{\text{occ}}\sum_l^{\text{virt}} t_{jl} O^{j\to l}\right)\end{aligned} \tag{A.25}$$

avec $O^{j\to l} = D_0^{j\to l}/D_0$. Si les orbitales sont canoniques (vecteurs propres de la matrice de Fock), la différence d'énergie dans le développement en perturbations peut être estimée en utilisant l'approche Møller-Plesset :

$$E_k - E_0 \sim \epsilon_l - \epsilon_i \tag{A.26}$$

Les $\langle \Delta E \Delta O^{j\to l}\rangle_{\Psi_T^2}$, et les $\langle \Delta O^{j\to l}\Delta O^{j\to m}\rangle_{\Psi_T^2}$ sont échantillonnées par VMC, et les paramètres $V^{j\to l}$ sont trouvés par l'ajustement aux moindres carrés des fluctuations de l'énergie locale. La fonction d'onde est mise à jour comme

$$\Psi' = \Psi\left(1 - \sum_j^{\text{occ}}\sum_l^{\text{virt}} \frac{V^{j\to l}}{\epsilon_l - \epsilon_i}\right) \tag{A.27}$$

et l'optimisation est itérée jusqu'à la convergence en gardant les énergies orbitalaires ϵ_i constantes au cours des itérations. Nous utilisons la procédure de Löwdin pour rendre les orbitales orthonormales après chaque itération.

Estimation des dénominateurs

Dans le cas où Φ est une expansion MCSCF, les configurations de la fonction d'onde d'essai sont construites sur les orbitales pseudo-canoniques. Afin d'obtenir une estimation correcte des différences d'énergies, les énergies associées aux orbitales (éléments diagonaux de la matrice de Fock) doivent être ajustées comme[88] :

$$E^{j\to l} - E_0 = \epsilon_l - \epsilon_j + \frac{\lambda}{2}\left(\Gamma_{ll} + 2 - \Gamma_{jj}\right) \tag{A.28}$$

où Γ est la matrice de densité à une particule, et λ est un paramètre de décalage en énergie, choisi égal à 0,5 Hartree. On peut remarquer que dans le cas d'un déterminant Hartree-Fock, on retombe bien sur la différence des valeurs propres.

A.3.4 États excités

Si l'on optimise les orbitales et les coefficients de l'IC d'un état excité de même symétrie que le fondamental par minimisation de l'énergie, on risque bien souvent de retomber sur l'état fondamental. De plus, les orbitales optimales pour l'état excité ne décrivent pas bien le fondamental, donc l'état excité considéré est orthogonal à un état fondamental mal décrit. Ces problèmes peuvent être réduits par la méthode *state-average*, où les orbitales sont optimisés pour plusieurs états en minimisant une moyenne pondérée des énergie des états :

$$E_{\mathrm{SA}} = \sum_i w_i \frac{\langle\Psi_i|\mathcal{H}|\Psi_i\rangle}{\langle\Psi_i|\Psi_i\rangle} \tag{A.29}$$

où les w_i sont les poids de chaque état i. L'expression du χ^2 dans l'ajustement aux moindre carrés devient :

$$\chi^2 = \sum_i w_i \left\langle E_L^{(i)} - \sum_k V_{ki} O_{ki} \right\rangle_{\Psi_i^2} \tag{A.30}$$

La minimisation du χ^2 engendre un ensemble d'équations linéaires pour chaque état i, produisant des fonctions d'ondes améliorées en utilisant le développement en perturbations. Ainsi, pour chaque état les orbitales ont été améliorées à partir des orbitales de départ $\{\varphi\}$, puis orthonormalisées. Les matrices densité sont projetées sur les orbitales de départ, et moyennées pour donner la matrice de densité *state-average* :

$$\Gamma^{SA} = \sum_i w_i \Gamma^{(i)} \tag{A.31}$$

qui est diagonalisée pour donner les orbitales naturelles moyennes améliorées. Ces orbitales sont retransformées en orbitales pseudo-canoniques, meilleures que les orbitales canoniques de départ.

A.3.5 Application : Optimisation simultanée de trois états de N_2

Un facteur de Jastrow reproduisant seulement le cusp inter-electronique ($\mathcal{J}_1$) a été ajouté à une fonction d'onde CAS, dont l'espace actif était composé de 10 électrons et 8 orbitales. Le poids de chacun des trois états dans le calcul state-average était de 1/3. Puis, la composante déterminantale Φ a été optimisée avec un *cut-off* de 0,08 appliqué aux coefficients des configurations. La convergence de l'énergie VMC

pour les deux premiers états de symétrie Σ_g^+ et le premier état Σ_u^+ est montrée sur la figure A.9.

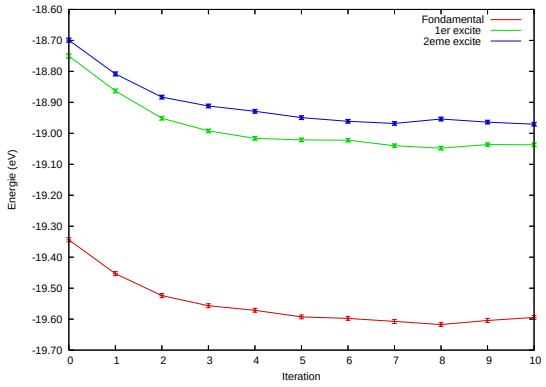


FIGURE A.9 – N_2 : convergence des deux premiers états Σ_g^+ et du premier état Σ_u^+ (le plus haut en énergie).

Puis les autres paramètres du facteur de Jastrow ont été optimisés ($\mathcal{J}_2$), et Φ a été optimisée à nouveau en présence de $\mathcal{J}_2$, avec un *cut-off* de 0,15 appliqué aux coefficients des configurations. Les énergies totales DMC sont présentées dans le tableau A.1 avec et sans optimisation de Φ.

D'après les résultats DMC du tableau A.1, on peut constater que les énergies des trois états sont abaissées par l'optimisation de Φ. L'erreur des nœuds fixés est donc réduite pour chacun des états. On peut aussi remarquer que les fonctions d'ondes optimisées sont extrêmement compactes avec seulement 1, 6 et 5 configurations.

	Σ_g^+	Σ_g^+	Σ_u^+
CAS-SCF			
N_{CSF}	176	176	176
E(DMC) (ua)	-19.585	-18.950	-18.901
ΔE (DMC) (eV)	0	17.27(3)	18.61(3)
$\mathcal{J}_1$			
N_{CSF}	1	25	25
E(VMC) (ua)	-19.350(8)	-18.777(8)	-18.705(8)
σ^2 (ua)	1.47	1.43	1.52
ΔE (VMC) (eV)	0	15.60(3)	17.55(3)
E(DMC) (ua)	-19.854(3)	-19.213(3)	-19.179(3)
σ^2 (ua)	1.89	1.70	1.63
ΔE (DMC) (eV)	0	17.449(4)	18.367(4)
$\mathcal{J}_1$ et optimisation			
E(VMC) (ua)	-19.605(7)	-19.041(6)	-18.978(6)
σ^2 (ua)	1.05	0.90	0.90
ΔE (VMC) (eV)	0	15.34(2)	17.05(2)
E(DMC) (ua)	-19.864(2)	-19.222(2)	-19.167(2)
σ^2 (ua)	1.17	0.77	1.04
ΔE (DMC) (eV)	0	17.442(3)	18.970(3)
$\mathcal{J}_2$ et optimisation			
N_{CSF}	1	6	5
E(VMC) (ua)	-19.756(3)	-19.172(3)	-19.118(3)
σ^2 (ua)	0.57	0.51	0.51
ΔE (VMC) (eV)	0	15.897(9)	17.371(9)
E(DMC) (ua)	-19.879(2)	-19.254(2)	-19.184(2)
σ^2 (ua)	0.64	0.54	0.63
ΔE (DMC) (eV)	0	16.998(3)	18.884(3)

TABLE A.1 – énergies obtenues avec les facteurs de Jastrow $\mathcal{J}_1$ et $\mathcal{J}_2$, suivi de l'optimisation de Φ.

A.4 Travail post-doctoral : Amélioration de l'échantillonnage en QMC

J'ai réalisé un second séjour post-doctoral dans l'équipe d'Eric Cancès au CERMICS (Champs-sur-Marne). Cette équipe vise à rationaliser et améliorer les méthodes de la chimie théorique grâce à leur expertise en mathématiques appliquées.

Nous avons proposé un nouvel algorithme d'échantillonnage pour le VMC,[89] permettant d'obtenir l'état ergodique plus rapidement qu'avec la diffusion brownienne et la dérive. Cet algorithme fait intervenir une dynamique fictive de Langevin pour les électrons dans l'espace des positions et des vitesses. Ce qui motive ce choix est tout d'abord qu'en dynamique moléculaire classique, ce type de dynamique est connu pour être plus performant que les marches aléatoires dans l'espace des positions. Deuxièmement, à l'approche d'une surface nodale le terme de dérive $\frac{\nabla\Psi(\mathbf{R})}{\Psi(\mathbf{R})}$ diverge et une distance artificiellement trop grande sera parcourue. Avec la dynamique de Langevin, cet effet interviendra dans les vitesses et non les positions, ce qui donnera une meilleur densité dans l'espace des positions.

A.4.1 Dynamique de Langevin

La dynamique de Langevin d'un système de N particules de masse m dans un potentiel V est donnée par

$$\mathrm{d}\mathbf{R}(t) = \frac{1}{m}\mathbf{P}(t)\,\mathrm{d}t \tag{A.32}$$

$$\mathrm{d}\mathbf{P}(t) = -\nabla V(\mathbf{R}(t))\,\mathrm{d}t - \gamma\mathbf{P}(t)\,\mathrm{d}t + \sigma\,\mathrm{d}\mathbf{W}(t) \tag{A.33}$$

avec $\sigma^2 = \frac{2m\gamma}{\beta}$. $\mathbf{R}(t)$ et $\mathbf{P}(t)$ sont les vecteurs des positions est des vitesses au temps t des N particules, $W(t)$ est le processus de Wiener (mouvement Brownien) à $3N$ dimensions. La dynamique de Langevin peut être interprétée comme une dynamique Newtonienne ($\gamma = 0$ et $\sigma = 0$) perturbée par une dérive $-\gamma\mathbf{P}(t)\,\mathrm{d}t$ et une force aléatoire $\sigma\,\mathrm{d}\mathbf{W}(t)$. En choisissant $\beta = 1$ et $V = -\ln|\Psi|^2$, on peut montrer que la dynamique de Langevin projetée dans l'espace des positions échantillonne bien $\frac{|\Psi(\mathbf{R})|^2}{\langle\Psi|\Psi\rangle}$.

Pour implémenter cette dynamique, nous avons choisi la discrétisation de Ricci et Ciccotti[90] car c'est celle qui donnait la plus petite erreur :

$$\mathbf{R}_{n+1} = \mathbf{R}_n + \frac{\tau}{m}\mathbf{P}_n e^{-\gamma\tau/2} + \frac{\tau}{2m}\big[-\nabla V(\mathbf{R}_n)\tau + \mathbf{G}_n\big]e^{-\gamma\tau/4} \tag{A.34}$$

$$\mathbf{P}_{n+1} = \mathbf{P}_n e^{-\gamma\tau} - \frac{\tau}{2}\big[\nabla V(\mathbf{R}_n) + \nabla V(\mathbf{R}_{n+1})\big]e^{-\gamma\tau/2} + \mathbf{G}_n e^{-\gamma\tau/2} \tag{A.35}$$

où $\mathbf{G}$ est un vecteur aléatoire gaussien de variance $\sigma^2 = \frac{2\gamma m}{\beta}\tau$.

A.4.2 Introduction du pas de Metropolis

Pour avoir une densité de transition, il est nécessaire de modifier l'algorithme en introduisant deux vecteurs aléatoires corrélés :

$$\langle (G_i^1)^2 \rangle = \sigma_1^2 = \frac{\tau}{\beta m \gamma} \left(2 - \frac{3 - 4e^{-\gamma\tau} + e^{-2\gamma\tau}}{\gamma\tau} \right) \tag{A.36a}$$

$$\langle (G_i^2)^2 \rangle = \sigma_2^2 = \frac{m}{\beta} \left(1 - e^{-2\gamma\tau} \right) \tag{A.36b}$$

$$\frac{\langle G_i^1 G_i^2 \rangle}{\sigma_1 \sigma_2} = c_{12} = \frac{(1 - e^{-\gamma\tau})^2}{\beta\gamma\sigma_1\sigma_2}. \tag{A.36c}$$

La nouvelle discrétisation est donc

$$\mathbf{R}_{n+1} = \mathbf{R}_n + \frac{\tau}{m}\mathbf{P}_n e^{-\gamma\tau/2} - \frac{\tau^2}{2m}\nabla V(\mathbf{R}_n) e^{-\gamma\tau/4} + \mathbf{G}_n^1 \tag{A.37}$$

$$\mathbf{P}_{n+1} = \mathbf{P}_n e^{-\gamma\tau} - \frac{\tau}{2}\left[\nabla V(\mathbf{R}_n) + \nabla V(\mathbf{R}_{n+1})\right] e^{-\gamma\tau/2} + \mathbf{G}_n^2 \tag{A.38}$$

avec la densité de transition

$$T_\tau((\mathbf{R}_n, \mathbf{P}_n) \to (\mathbf{R}_{n+1}, \mathbf{P}_{n+1})) = Z^{-1} \exp\left[-\frac{1}{2(1 - c_{12}^2)} \left(\left(\frac{|\mathbf{d}_1|}{\sigma_1} \right)^2 + \left(\frac{|\mathbf{d}_2|}{\sigma_2} \right)^2 - 2c_{12} \frac{\mathbf{d}_1}{\sigma_1} \cdot \frac{\mathbf{d}_2}{\sigma_2} \right) \right], \tag{A.39a}$$

et

$$\mathbf{d}_1 = \mathbf{R}_{n+1} - \mathbf{R}_n - \tau \frac{\mathbf{P}_n}{m} e^{-\gamma\tau/2} + \frac{\tau^2}{2m}\nabla V(\mathbf{R}_n) e^{-\gamma\tau/4}, \tag{A.39b}$$

$$\mathbf{d}_2 = \mathbf{P}_{n+1} - \mathbf{P}_n e^{-\gamma\tau} + \frac{1}{2}\tau\left[\nabla V(\mathbf{R}_n) + \nabla V(\mathbf{R}_{n+1})\right] e^{-\gamma\tau/2}. \tag{A.39c}$$

L'utilisation de cette densité de transition dans l'algorithme de Metropolis engendre un très fort taux de rejet. Cela vient du fait que la probabilité que les forces aléatoires soient suffisantes pour permettre de faire le déplacement inverse est en général très faible. Pour pallier ce problème, nous utilisons la propriété que la moyenne du vecteur des vitesses est nulle. En effet, la densité de transition $\tilde{T}_\tau((\mathbf{R}, \mathbf{P}) \longrightarrow (\mathbf{R}', \mathbf{P}') = T_\tau((\mathbf{R}, \mathbf{P}) \longrightarrow (\mathbf{R}', -\mathbf{P}')$ échantillonne aussi $\frac{|\Psi(\mathbf{R})|^2}{\langle\Psi|\Psi\rangle}$ et vérifie le bilan détaillé, nécessaire à l'algorithme de Metropolis. On introduit donc l'algorithme suivant :

1. On propose un déplacement $(\mathbf{R}_n, \mathbf{P}_n) \longrightarrow (\mathbf{R}_{n+1}, \mathbf{P}_{n+1})$ (Eq A.37)
2. On calcule le taux d'acceptation en utilisant la densité de transition $\tilde{T}\tau$
3. Si le pas est accepté, la dynamique continue en $(\mathbf{R}_{n+1}, \mathbf{P}_{n+1})$. Si le pas est rejeté, la dynamique continue en $(\mathbf{R}_n, -\mathbf{P}_n)$

A.4.3 Résultats

Pour mesurer d'efficacité de l'algorithme, nous avons calculé les temps d'autocorrélation de l'énergie dans plusieurs systèmes : les atomes de Lithium, Fluor et Cuivre, et aussi dans la molécule de phénol. Dans tous les cas nous avons observé un temps d'autocorrélation environ 75% plus faible environ avec notre algorithme qu'avec la diffusion brownienne et la dérive. D'autre part, le régime où l'algorithme

est le plus efficace est celui où le taux d'acceptation est de l'ordre de 0.90, contrairement à l'algorithme standard où le taux d'acceptation est plutôt de l'ordre de 0.50. Ainsi, on bénéficie d'une meilleure exploration de l'espace des phases.

Bibliographie

[1] A. M. Turing, "On computable numbers, with an application to the entscheidungsproblem," *Proceedings of the London Mathematical Society*, vol. s2-42, pp. 230–265, jan 1937.

[2] R. Krithivasan, Y. Lu, J. Cressler, J.-S. Rieh, M. Khater, D. Ahlgren, and G. Freeman, "Half-terahertz operation of SiGe HBTs," *IEEE Electron Device Letters*, vol. 27, pp. 567–569, jul 2006.

[3] H. Sutter and J. Larus, "Software and the concurrency revolution," *Queue*, vol. 3, p. 54, sep 2005.

[4] G. H. Booth, A. J. W. Thom, and A. Alavi, "Fermion monte carlo without fixed nodes : A game of life, death, and annihilation in slater determinant space," *The Journal of Chemical Physics*, vol. 131, no. 5, p. 054106, 2009.

[5] A. J. W. Thom and A. Alavi, "Stochastic perturbation theory : A low-scaling approach to correlated electronic energies," *Physical Review Letters*, vol. 99, oct 2007.

[6] R. S. T. Franklin, J. S. Spencer, A. Zoccante, and A. J. W. Thom, "Linked coupled cluster monte carlo," *The Journal of Chemical Physics*, vol. 144, p. 044111, jan 2016.

[7] A. Scemama, T. Applencourt, E. Giner, and M. Caffarel, "Quantum Monte Carlo with very large multideterminant wavefunctions," *J. Comput. Chem.*, vol. 37, pp. 1866–1875, jun 2016.

[8] E. Giner, *Méthodes d'interaction de configurations et Monte Carlo quantique : marier le meilleur des deux mondes.* Theses, Université de Toulouse, Oct. 2014.

[9] Y. Garniron, *Thèse en cours.* Theses, Université Paul Sabatier - Toulouse III, 2018.

[10] M. W. Schmidt, K. K. Baldridge, J. A. Boatz, S. T. Elbert, M. S. Gordon, J. H. Jensen, S. Koseki, N. Matsunaga, K. A. Nguyen, S. Su, T. L. Windus, M. Dupuis, and J. A. Montgomery, "General atomic and molecular electronic structure system," *Journal of Computational Chemistry*, vol. 14, pp. 1347–1363, nov 1993.

[11] H.-J. Werner, P. J. Knowles, G. Knizia, F. R. Manby, and M. Schütz, "Molpro : a general-purpose quantum chemistry program package," *WIREs Comput Mol Sci*, vol. 2, pp. 242–253, 2012.

[12] T. Applencourt, *Calcul haute performance & chimie quantique.* Theses, Université Paul Sabatier - Toulouse III, Nov. 2015.

[13] P. Hintjens, *ZeroMQ : connecting your code.* O'Reilly Media, 2012.

[14] F. Matsen, "Spin-free quantum chemistry," in *Advances in Quantum Chemistry*, pp. 59–114, Elsevier BV, 1964.

[15] F. A. Matsen and A. A. Cantu, "Spin-free quantum chemistry. VII. the slater determinant," *The Journal of Physical Chemistry*, vol. 73, pp. 2488–2494, aug 1969.

[16] B. O. Roos, "The complete active space SCF method in a fock-matrix-based super-CI formulation," *International Journal of Quantum Chemistry*, vol. 18, pp. 175–189, jun 1980.

[17] K. Andersson, P. A. Malmqvist, B. O. Roos, A. J. Sadlej, and K. Wolinski, "Second-order perturbation theory with a CASSCF reference function," *The Journal of Physical Chemistry*, vol. 94, pp. 5483–5488, jul 1990.

[18] C. Angeli, R. Cimiraglia, S. Evangelisti, T. Leininger, and J.-P. Malrieu, "Introduction of n-electron valence states for multireference perturbation theory," *The Journal of Chemical Physics*, vol. 114, pp. 10252–10264, jun 2001.

[19] E. Giner, C. Angeli, Y. Garniron, A. Scemama, and J.-P. Malrieu, "A Jeziorski-Monkhorst fully uncontracted multi-reference perturbative treatment I : principles, second-order versions and tests on ground state potential energy curves," *ArXiv e-prints*, p. 1702.03133, Feb. 2017.

[20] H. P. Kelly and A. M. Sessler, "Correlation effects in many-fermion systems : Multiple-particle excitation expansion," *Physical Review*, vol. 132, pp. 2091–2095, dec 1963.

[21] W. Meyer, "Ionization energies of water from PNO-CI calculations," *International Journal of Quantum Chemistry*, vol. 5, pp. 341–348, jun 1971.

[22] F. Coester, "Bound states of a many-particle system," *Nuclear Physics*, vol. 7, pp. 421–424, jun 1958.

[23] J. Čížek, "On the correlation problem in atomic and molecular systems. calculation of wavefunction components in ursell-type expansion using quantum-field theoretical methods," *The Journal of Chemical Physics*, vol. 45, pp. 4256–4266, dec 1966.

[24] J.-P. Daudey, J.-L. Heully, and J.-P. Malrieu, "Size-consistent self-consistent truncated or selected configuration interaction," *The Journal of Chemical Physics*, vol. 99, pp. 1240–1254, jul 1993.

[25] A. Scemama, M. Caffarel, R. Chaudret, and J.-P. Piquemal, "Electron pair localization function (EPLF) for density functional theory and ab initio wave function-based methods : A new tool for chemical interpretation," *J. Chem. Theory Comput.*, vol. 7, pp. 618–624, mar 2011.

[26] M. Caffarel, T. Applencourt, E. Giner, and A. Scemama, "Communication : Toward an improved control of the fixed-node error in quantum Monte Carlo : The case of the water molecule," *The Journal of Chemical Physics*, vol. 144, p. 151103, apr 2016.

[27] L. S. Blackford, *ScaLAPACK user's guide*. Philadelphia : SIAM, 1997.

[28] A. V. Akimov and O. V. Prezhdo, "Large-scale computations in chemistry : A bird's eye view of a vibrant field," *Chemical Reviews*, vol. 115, pp. 5797–5890, jun 2015.

[29] C. Bender, "Integral transformations. a bottleneck in molecular quantum mechanical calculations," *Journal of Computational Physics*, vol. 9, pp. 547–554, jun 1972.

[30] P. Pendergast and E. F. Hayes, "A partial transformation for application to perturbation theory configuration interaction," *Journal of Computational Physics*, vol. 26, pp. 236–242, feb 1978.

[31] V. R. Saunders and J. H. V. Lenthes, "The direct CI method. a detailed analysis," *Molecular Physics*, vol. 100, pp. 167–187, jan 2002.

[32] E. R. Davidson, "The iterative calculation of a few of the lowest eigenvalues and corresponding eigenvectors of large real-symmetric matrices," *Journal of Computational Physics*, vol. 17, pp. 87–94, jan 1975.

[33] W. Gropp, E. Lusk, N. Doss, and A. Skjellum, "A high-performance, portable implementation of the MPI message passing interface standard," *Parallel Computing*, vol. 22, pp. 789–828, sep 1996.

[34] L. Dagum and R. Menon, "OpenMP : an industry standard API for shared-memory programming," *IEEE Computational Science and Engineering*, vol. 5, no. 1, pp. 46–55, 1998.

[35] C. F. Bender and E. R. Davidson, "Studies in configuration interaction : The first-row diatomic hydrides," *Phys. Rev.*, vol. 183, pp. 23–30, jul 1969.

[36] J. L. Whitten, "Configuration interaction studies of ground and excited states of polyatomic molecules. i. the CI formulation and studies of formaldehyde," *The Journal of Chemical Physics*, vol. 51, no. 12, p. 5584, 1969.

[37] B. Huron, "Iterative perturbation calculations of ground and excited state energies from multiconfigurational zeroth-order wavefunctions," *The Journal of Chemical Physics*, vol. 58, no. 12, p. 5745, 1973.

[38] G. K.-L. Chan and M. Head-Gordon, "Exact solution (within a triple-zeta, double polarization basis set) of the electronic schrödinger equation for water," *The Journal of Chemical Physics*, vol. 118, no. 19, p. 8551, 2003.

[39] P.-O. Widmark, B. J. Persson, and B. O. Roos, "Density matrix averaged atomic natural orbital (ANO) basis sets for correlated molecular wave functions," *Theoretica Chimica Acta*, vol. 79, no. 6, pp. 419–432, 1991.

[40] D. E. Woon and T. H. Dunning, "Gaussian basis sets for use in correlated molecular calculations. v. core-valence basis sets for boron through neon," *The Journal of Chemical Physics*, vol. 103, no. 11, p. 4572, 1995.

[41] J. A. Pople, M. Head-Gordon, D. J. Fox, K. Raghavachari, and L. A. Curtiss, "Gaussian-1 theory : A general procedure for prediction of molecular energies," *The Journal of Chemical Physics*, vol. 90, no. 10, p. 5622, 1989.

[42] I. Nebot-Gil, J. Sanchez-Marin, J. Heully, J. Malrieu, and D. Maynau, "Eigenvalue problem formulation of coupled-cluster expansions through intermediate hamiltonians," *Chemical Physics Letters*, vol. 234, pp. 45–49, mar 1995.

[43] E. Giner, G. David, A. Scemama, and J. P. Malrieu, "A simple approach to the state-specific MR-CC using the intermediate Hamiltonian formalism," *The Journal of Chemical Physics*, vol. 144, p. 064101, feb 2016.

[44] B. Jeziorski and H. J. Monkhorst, "Coupled-cluster method for multideterminantal reference states," *Phys. Rev. A*, vol. 24, pp. 1668–1681, oct 1981.

[45] J. Meller, J. P. Malrieu, and R. Caballol, "State-specific coupled cluster-type dressing of multireference singles and doubles configuration interaction matrix," *The Journal of Chemical Physics*, vol. 104, no. 11, p. 4068, 1996.

[46] Y. Garniron, E. Giner, J.-P. Malrieu, and A. Scemama, "Alternative definition of excitation amplitudes in multi-reference state-specific coupled cluster," *J. Chem. Phys*, p. in press, Apr. 2017.

[47] A. Scemama and E. Giner, "An efficient implementation of Slater-Condon rules," *arXiv :1311.6244 [physics.comp-ph]*, 2013.

[48] C. Filippi and S. Fahy, "Optimal orbitals from energy fluctuations in correlated wave functions," *The Journal of Chemical Physics*, vol. 112, no. 8, p. 3523, 2000.

[49] F. Schautz and S. Fahy, "Optimization of configuration interaction coefficients in multideterminant jastrow–slater wave functions," *The Journal of Chemical Physics*, vol. 116, no. 9, p. 3533, 2002.

[50] F. Schautz and C. Filippi, "Optimized jastrow–slater wave functions for ground and excited states : Application to the lowest states of ethene," *The Journal of Chemical Physics*, vol. 120, no. 23, p. 10931, 2004.

[51] C. J. Umrigar and C. Filippi, "Energy and variance optimization of many-body wave functions," *Phys. Rev. Lett.*, vol. 94, apr 2005.

[52] A. Scemama and C. Filippi, "Simple and efficient approach to the optimization of correlated wave functions," *Phys. Rev. B*, vol. 73, jun 2006.

[53] J. Toulouse and C. J. Umrigar, "Optimization of quantum monte carlo wave functions by energy minimization," *The Journal of Chemical Physics*, vol. 126, no. 8, 2007.

[54] J. Toulouse and C. J. Umrigar, "Full optimization of jastrow–slater wave functions with application to the first-row atoms and homonuclear diatomic molecules," *The Journal of Chemical Physics*, vol. 128, no. 17, p. 174101, 2008.

[55] B. K. Clark, M. A. Morales, J. McMinis, J. Kim, and G. E. Scuseria, "Computing the energy of a water molecule using multideterminants : A simple efficient algorithm," *J. Chem. Phys.*, vol. 135, no. 24, p. 244105, 2011.

[56] W. Klopper, "Highly accurate coupled-cluster singlet and triplet pair energies from explicitly correlated calculations in comparison with extrapolation techniques," *Molecular Physics*, vol. 99, pp. 481–507, mar 2001.

[57] M. Casula, C. Attaccalite, and S. Sorella, "Correlated geminal wave function for molecules : An efficient resonating valence bond approach," *The Journal of Chemical Physics*, vol. 121, no. 15, p. 7110, 2004.

[58] I. G. Gurtubay and R. J. Needs, "Dissociation energy of the water dimer from quantum monte carlo calculations," *The Journal of Chemical Physics*, vol. 127, no. 12, p. 124306, 2007.

[59] A. Lüchow, J. B. Anderson, and D. Feller, "Improved estimates of the total correlation energy in the ground state of the water molecule," *The Journal of Chemical Physics*, vol. 106, no. 18, p. 7706, 1997.

[60] C. X. Almora-Díaz, "Highly correlated configuration interaction calculations on water with large orbital bases," *The Journal of Chemical Physics*, vol. 140, p. 184302, may 2014.

[61] B. H. Muller and W. Kutzelnigg, "A CCSD(t)-r12 study of the ten-electron systems ne, f-, HF, h2o, NH3, NH4+ and CH4," *Molecular Physics*, vol. 92, pp. 535–546, oct 1997.

[62] T. Helgaker, W. Klopper, H. Koch, and J. Noga, "Basis-set convergence of correlated calculations on water," *The Journal of Chemical Physics*, vol. 106, no. 23, p. 9639, 1997.

[63] A. Halkier, T. Helgaker, P. Jørgensen, W. Klopper, H. Koch, J. Olsen, and A. K. Wilson, "Basis-set convergence in correlated calculations on ne, n2, and h2o," *Chemical Physics Letters*, vol. 286, pp. 243–252, apr 1998.

[64] L. Bytautas and K. Ruedenberg, "Correlation energy extrapolation by intrinsic scaling. v. electronic energy, atomization energy, and enthalpy of formation of water," *The Journal of Chemical Physics*, vol. 124, no. 17, p. 174304, 2006.

[65] E. Giner, A. Scemama, and M. Caffarel, "Fixed-node diffusion Monte Carlo potential energy curve of the fluorine molecule F_2 using selected configuration interaction trial wavefunctions," *The Journal of Chemical Physics*, vol. 142, p. 044115, jan 2015.

[66] L. Bytautas, N. Matsunaga, T. Nagata, M. S. Gordon, and K. Ruedenberg, "Accurate ab initio potential energy curve of f[sub 2]. II. core-valence correlations, relativistic contributions, and long-range interactions," *The Journal of Chemical Physics*, vol. 127, no. 20, p. 204301, 2007.

[67] A. Scemama, T. Applencourt, E. Giner, and M. Caffarel, "Accurate nonrelativistic ground-state energies of 3d transition metal atoms," *The Journal of Chemical Physics*, vol. 141, p. 244110, dec 2014.

[68] E. Buendía, F. Gálvez, P. Maldonado, and A. Sarsa, "Quantum monte carlo ionization potential and electron affinity for transition metal atoms," *Chemical Physics Letters*, vol. 559, pp. 12–17, feb 2013.

[69] A. Monari, A. Scemama, and M. Caffarel, "Large-scale quantum Monte Carlo electronic structure calculations on the EGEE grid," in *Remote Instrumentation for eScience and Related Aspects*, pp. 195–207, Springer Science + Business Media, 2012.

[70] A. Scemama, M. Caffarel, E. Oseret, and W. Jalby, "QMC=Chem : A quantum Monte Carlo program for large-scale simulations in chemistry at the petascale level and beyond," in *Lecture Notes in Computer Science*, pp. 118–127, Springer Science + Business Media, 2013.

[71] A. Scemama, M. Caffarel, E. Oseret, and W. Jalby, "Quantum Monte Carlo for large chemical systems : Implementing efficient strategies for petascale platforms and beyond," *J. Comput. Chem.*, vol. 34, pp. 938–951, jan 2013.

[72] R. Assaraf, M. Caffarel, and A. Khelif, "Diffusion monte carlo methods with a fixed number of walkers," *Phys. Rev. E*, vol. 61, 2000.

[73] A. B. Yoo, M. A. Jette, and M. Grondona, "SLURM : Simple linux utility for resource management," in *Job Scheduling Strategies for Parallel Processing*, pp. 44–60, Springer Science + Business Media, 2003.

[74] L. Djoudi, J. Noudohouenou, and W. Jalby, "The design and architecture of MAQAOAdvisor : A live tuning guide," in *High Performance Computing - HiPC 2008*, pp. 42–56, Springer Science + Business Media, 2880.

[75] S. Y. Willow, K. S. Kim, and S. Hirata, "Stochastic evaluation of second-order many-body perturbation energies," *The Journal of Chemical Physics*, vol. 137, p. 204122, nov 2012.

[76] Y. Cytter, D. Neuhauser, and R. Baer, "Metropolis evaluation of the hartree–fock exchange energy," *Journal of Chemical Theory and Computation*, vol. 10, pp. 4317–4323, oct 2014.

[77] D. C. Rawlings and E. R. Davidson, "The rayleigh—schrödinger BK method applied to the lower electronic states of pyrrole," *Chemical Physics Letters*, vol. 98, pp. 424–427, jul 1983.

[78] W. Kutzelnigg, "r 12-dependent terms in the wave function as closed sums of partial wave amplitudes for large l," *Theoretica Chimica Acta*, vol. 68, pp. 445–469, dec 1985.

[79] D. P. Tew, W. Klopper, and T. Helgaker, "Electron correlation : The many-body problem at the heart of chemistry," *Journal of Computational Chemistry*, vol. 28, no. 8, pp. 1307–1320, 2007.

[80] E. S. Larsen and D. McAllister, "Fast matrix multiplies using graphics hardware," in *Proceedings of the 2001 ACM/IEEE conference on Supercomputing (CDROM) – Supercomputing 01*, ACM Press, 2001.

[81] A. Scemama, P. Chaquin, M.-C. Gazeau, and Y. Bénilan, "Theoretical study of the structure and properties of polyynes and monocyano- and dicyanopolyynes : Predictions for long chain compounds," *J. Phys. Chem. A*, vol. 106, pp. 3828–3837, apr 2002.

[82] V. Vuitton, A. Scemama, M.-C. Gazeau, P. Chaquin, and Y. Bénilan, "IR and UV spectroscopic data for polyynes : Predictions for long carbon chain compounds in Titan's atmosphere," *Advances in Space Research*, vol. 27, pp. 283–288, jan 2001.

[83] A. Scemama, P. Chaquin, M.-C. Gazeau, and Y. Bénilan, "Semi-empirical calculation of electronic absorption wavelengths of polyynes, monocyano- and dicyanopolyynes. predictions for long chain compounds and carbon allotrope carbyne," *Chemical Physics Letters*, vol. 361, pp. 520–524, aug 2002.

[84] P. Chaquin and A. Scemama, "Theoretical study of the electrocyclization product of butadiyne : structure, stability and possible formations," *Chemical Physics Letters*, vol. 394, pp. 244–249, aug 2004.

[85] A. Scemama, P. Chaquin, and M. Caffarel, "Electron pair localization function : A practical tool to visualize electron localization in molecules from quantum Monte Carlo data," *The Journal of Chemical Physics*, vol. 121, no. 4, p. 1725, 2004.

[86] A. Scemama, "Investigating the volume maximizing the probability of finding ν electrons from variational Monte Carlo data," *Journal of Theoretical and Computational Chemistry*, vol. 04, pp. 397–409, jun 2005.

[87] A. Scemama, M. Caffarel, and A. Savin, "Maximum probability domains from quantum Monte Carlo calculations," *J. Comput. Chem.*, vol. 28, no. 1, pp. 442–454, 2007.

[88] G. Ghigo, B. O. Roos, and P.-Å. Malmqvist, "A modified definition of the zeroth-order hamiltonian in multiconfigurational perturbation theory (CASPT2)," *Chemical Physics Letters*, vol. 396, pp. 142–149, sep 2004.

[89] A. Scemama, T. Lelièvre, G. Stoltz, E. Cancès, and M. Caffarel, "An efficient sampling algorithm for variational Monte Carlo," *The Journal of Chemical Physics*, vol. 125, no. 11, p. 114105, 2006.

[90] A. Ricci and G. Ciccotti, "Algorithms for brownian dynamics," *Molecular Physics*, vol. 101, pp. 1927–1931, jun 2003.