UNIVERSITY OF CALIFORNIA,
IRVINE

Structure-Preserving Methods for Molecular Response Calculations

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Chemistry

by

Samuel Bekoe

Dissertation Committee:
Professor Filipp Furche, Chair
Professor Vladimir Mandelshtam
Professor Williams J. Evans

2023

DEDICATION

I dedicate this thesis to my mother Salomey Bekoe, my beloved Dr Iris Baffour Ansah, my uncles Kwame Quansah and Patrick Agyekum and to the memory of Dr Richard Tia. Their unwavering support have been a constant source of strength and motivation, and I am truly fortunate to have had such wonderful people in my life.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I am sincerely grateful to my advisor, Dr. Filipp Furche, for his unwavering support throughout my academic journey. Since joining the lab as a graduate student in the winter quarter of 2019, Dr. Furche has been an exceptional mentor and a constant source of inspiration. His expertise in quantum mechanics and computational chemistry has fueled my continued interest in these fields and has motivated me to pursue research in theoretical and computational chemistry.

I would like to express my deep gratitude to the members of the Furche research group. Their support, patience, and guidance have played a crucial role in shaping me as a researcher, mentor, and teacher. In particular, I am grateful for the expertise and experience of Dr. Dmitrij Rappoport and Dr. Luke Nambi on the Libkrylov project, which has been instrumental in my research. I am also thankful for the collaboration with Dr. Saswata Roy and Sourav Majumdar on computational studies of lanthanide and actinide complexes.

I would like to acknowledge the contributions of Dr. Brian D. Nguyen, Dr. Jason Yu, Gabriel Phun, Jeffrey Tsai, Ahmad Rajabi, George Naje, Dr Robin Grotjahn, Mikko Stulajter and Nicolas Lutfi who have all contributed to my scientific development, improved my public speaking skills, and pushed me to strive for more.

Throughout my graduate career, I have had the privilege of mentoring high school students and undergraduates. I am grateful to Dr. Furche for providing me with the opportunity to participate in the Furche High School Outreach program. Working with the program has allowed me to mentor talented students and expand their understanding of chemistry. I am particularly thankful for the opportunity to mentor Elizabeth Bulla, Aakrsh Misra, and Scott Le.

I would also like to express my gratitude to the experimental groups I have collaborated with, particularly the Evans group at UCI. I am thankful for the scientific discussions and collaborations with Dr. Justin Wedal, Dr. Tener Jenkins, Dr. Sierra R. Ciccone, and Dr. Daniel N. Huh. I also appreciate the valuable insights gained from discussions with Dr. Williams J. Evans.

I am grateful to professors: Prof. Kieron Burke, Prof. Shaul Mukamel, Prof. Ioan Andricioaei, Prof. Craig Martens, Prof. Filipp Furche, and Prof. Vladimir Mandelshtam to whom have provided me the support and prerequisite knowledge needed to succeed in the field of quantum chemistry. The Department of Chemistry staff has been very supportive during my PhD study. Thank you to Tenley Dunn, Jenny Du, Morgan Sibley, Cynthia A. Dennis and Dr Nathan Crawford.

This thesis would not have been possible without the funding agencies and fellowships that have supported my graduate research career. The National Science Foundation (NSF) under Grant OAC-1835909 has been a significant source of support for my research projects, and the Molecular Sciences Software Institute (MolSSI) under NSF grant CHE-2136142 provided

financial support for my last year.

I am incredibly fortunate to have crossed paths with the UC Irvine Bridges International team, especially Trey Harmon and Justin McDonald, and their families. When I arrived as a first-timer in the United States, they warmly welcomed me into their community and provided much-needed support and guidance in navigating the complexities of American culture and university life. Their kindness, generosity, and willingness to lend a helping hand made a significant impact on my experience as an international student. Special thanks to Pastors Catherine Smith and Madelyn Traylor for their unconditional love and prayers. I am deeply grateful for the family we formed and the sense of belonging they helped create.

The following chapters of the dissertation are reprinted:

Chapter 2 of this dissertation is a reprint of the material as it appears in (Huh, D. N.; Ciccone, S. R.; Bekoe, S; Roy, S.; Ziller, J. W.; Furche, F.; Evans, W. J. Synthesis of Ln(II)-in-Cryptand Complexes by Chemical Reduction of Ln(III)-in-Cryptand Precursors: Isolation of a Nd(II)-in-Cryptand Complex. *Angew. Chem. Int. Ed.*, **2020**, 59(37), 16141–16146), used with permission from John Wiley & Sons Inc. The co-authors listed in this publication are Daniel N. Huh, Sierra R. Ciccone, Samuel Bekoe, Saswata Roy, Joseph W. Ziller, Filipp Furche and William J. Evans.

Chapter 3 of this dissertation is a reprint of the material as it appears in (Rappoport, D.; Bekoe, S.; Mohanam, L. N.; Le, S.; George, N.; Shen, Z.; Furche, F. Libkrylov: A modular open-source software library for extremely large on-the-fly matrix computations. *J. Comput. Chem.*, **2023**, 44(11), 1105), used with permission from John Wiley & Sons Inc. The co-authors listed in this publication are Dmitrij Rappoport, Samuel Bekoe, Luke N. Mohanam, Scott Le, Naje' George, Ziyue Shen, Filipp Furche.

VITA

Samuel Bekoe

EDUCATION

Doctor of Philosophy in Chemistry **2023**
University of California, Irvine *Irvine, CA*

Bachelor of Science in Chemistry **2017**
Kwame Nkrumah University of Science and Technology *Kumasi, GH*

RESEARCH EXPERIENCE

Graduate Research **2018–2023**
University of California, Irvine *Irvine, CA*

Undergraduate Research **2016–2017**
Kwame Nkrumah University of Science and Technology *Kumasi, GH*

TEACHING EXPERIENCE

General Chemistry Teaching Assistant **2019-2020**
University of California, Irvine *Irvine, CA*

Sci. Math Compskills Teaching Assistant **2018**
University of California, Irvine *Irvine, CA*

Ions in Solution Teaching Assistant **2018**
Kwame Nkrumah University of Science and Technology *Kumasi, GH*

Group Theory and Inorganic Spectroscopy Teaching Assistant **2017**
Kwame Nkrumah University of Science and Technology *Kumasi, GH*

Basic Physical Chemistry Teaching Assistant **2017–2018**
Kwame Nkrumah University of Science and Technology *Kumasi, GH*

General Chemistry Teaching Assistant **2017–2018**
Kwame Nkrumah University of Science and Technology *Kumasi, GH*

MENTORING EXPERIENCE

Furche High School Outreach Program **2019-2023**
University of California, Irvine *Irvine, CA*

Undergraduate Research Mentorship **2021-2022**
University of California, Irvine *Irvine, CA*

Undergraduate Research Mentorship **2017-2018**
Kwame Nkrumah University of Science and Technology *Kumasi, GH*

REFEREED JOURNAL PUBLICATIONS

1. Rappoport, D.; **Bekoe**, **S**.; Mohanam, L. N.; Le, S.; George, N.; Shen, Z.; Furche, F. Libkrylov: A modular open-source software library for extremely large on-the-fly matrix computations. *J. Comput. Chem.*, **2023**, 44(11), 1105.

2. Goodwin, C. A. P.; Ciccone, S. R.; **Bekoe**, **S**; Majumdar, S.; Scott, B. L.; Ziller, J. W.; Furche, F.; Evans, W. J. 2.2.2-cryptand complexes of neptunium(iii) and plutonium(iii). *Chem. Commun.*, **2022**, 58, 997–1000.

3. Jenkins, T. F.; **Bekoe**, **S**; Ziller, J. W.; Furche, F.; Evans, W. J. . Synthesis of a Heteroleptic Pentamethylcyclopentadienyl Yttrium(II) Complex, [K(2.2.2-Cryptand)] $\{(C_5Me_5)_2Y^{II}[N(SiMe_3)_2]\}$, and Its C–H Bond Activated Y(III) Derivative. *Organometallics*, **2021**, 40(23), 3917–3925.

4. Wedal, J. C.; **Bekoe**, **S**; Ziller, J. W.; Furche, F.; Evans, W. J. C–H Bond Activation via U(II) in the Reduction of Heteroleptic Bis(trimethylsilyl)amide U(III) Complexes. *Organometallics*, **2020**, 39, 3425–3432.

5. Huh, D. N.; Ciccone, S. R.; **Bekoe**, **S**; Roy, S.; Ziller, J. W.; Furche, F.; Evans, W. J. Synthesis of Ln(II)-in-Cryptand Complexes by Chemical Reduction of Ln(III)-in-Cryptand Precursors: Isolation of a Nd(II)-in-Cryptand Complex. *Angew. Chem. Int. Ed.*, **2020**, 59(37), 16141–16146.

6. Wedal, J. C.; **Bekoe**, **S**; Ziller, J. W.; Furche, F.; Evans, W. J. In Search of Tris-(trimethylsilylcyclopentadienyl) Thorium. *Dalton Trans.*, **2020**, 59(37), 48, 16633–16640.

7. **Bekoe**, **S**; Osei, M. K.; Tia, R.; Adei, E. Density Functional Theory Studies on the Generation of Trimethylenemethanes from the Ring Opening of Dialkoxymethylenecyclopropanes and Methylenecyclopropanethioacetals and Follow-up Reactions. *J. Mol. Model.*, **2017**, 24(1), 24.

CONFERENCE PRESENTATIONS

Libkrylov: A Modular Open-Source Software Library for extremely large on-the-fly matrix computation **Mar 2023**
Spring ACS National Meeting - Oral

SPECIAL TALKS

Libkrylov: A Modular Open-Source Software Library for extremely large on-the-fly matrix computation **Oct 2022**
SoCal TheoChem Symposium - Oral

Libkrylov: A Modular Open-Source Software Library for extremely large and dense eigenvalue and linear problems **Feb 2020**
NSF CSSI PI meeting - Oral/Poster

AWARDS AND HONORS

Molecular Sciences Software Institute (MolSSI) Software Fellowship **Jul 2022**
University of California, Irvine

SOFTWARE

Libkrylov `https://gitlab.com/libkrylov/libkrylov-stable`
A Modular Open-Source Software Library for extremely large on-the-fly matrix computation

ABSTRACT OF THE DISSERTATION

Structure-Preserving Methods for Molecular Response Calculations

By

Samuel Bekoe

Doctor of Philosophy in Chemistry

University of California, Irvine, 2023

Professor Filipp Furche, Chair

Time-dependent density functional theory (TDDFT) is a powerful and efficient method for calculating excitation energies and properties of electronic excited states. It has found wide applications in various scientific fields due to its accuracy and computational efficiency. However, solving the TDDFT equations involves large eigenvalue and linear problems, which can be computationally challenging. To address this, matrix-free iterative subspace algorithms have been developed. The first half of the thesis demonstrates the use of density functional theory (DFT) to elucidate the electronic structure of the first synthesis of Neodymium(II) encapsulated in a 2.2.2-cryptand ligand. The comparison between experimental results and the calculated molecular structure, as well as the UV-Vis spectrum obtained through TDDFT, provided strong evidence supporting the discovery of the traditional $4f^4$ electron configuration. In the second part of this thesis, I introduce *libkrylov*, which is a versatile and open-source Krylov subspace library designed for performing large-scale matrix computations on-the-fly. The main goals of libkrylov are to provide a versatile application programming interface (API) design and a modular structure that allows seamless integration with specialized matrix-vector evaluation "engines." The library is designed to offer pluggable preconditioning, orthonormalization, and tunable convergence control, making it highly flexible and easily adaptable to various computational scenarios and requirements. By providing these features, libkrylov enables users to customize and optimize their calculations, thereby

enhancing the efficiency and accuracy of large-scale matrix computations in computational chemistry and other related fields. I extend libkrylov to Hamiltonian structured problems often encountered in TDDFT. The implementation is based on preserving the full SO(1, 1) symmetry of the TDDFT response equations, which in the absence of magnetic fields reduces to the use of split-complex numbers. In Krylov subspace methods, this preservation is achieved by utilizing symmetry-adapted basis vectors that maintain the orthonormality condition of the symplectic problem. The calculation of excitation energies for some test molecules show improved convergence over the Olsen algorithm and the TURBOMOLE implementation. The improved convergence highlights the importance of the correct algebraic approach to the problem.

Chapter 1

Introduction

Recent developments in the field of organometallic chemistry have expanded our understanding of lanthanide and actinide series elements, particularly in relation to their variable oxidation states. These advancements have created opportunities for potential applications in various areas, including catalysis[1,2], magnetism[3–5], and energy production. These complexes have witnessed significant advancements in synthetic techniques in recent years[6–8]. However, due to their inherent instability, traditional spectroscopic methods struggle to characterize them effectively.

Understanding the electronic structure of these complexes is crucial for unraveling their unique properties and their implications in various chemical processes. Computational methods offer valuable insights into the properties and chemistry of these compounds. As part of my studies, I have delved into the world of computational chemistry, leveraging these methods to gain a deeper understanding of the electronic structure and behavior of lanthanide and actinide organometallic complexes.

Modeling the linear response properties of lanthanide and actinide complexes requires solving large-scale eigenvalue and linear problems involving dense coefficient matrices, often reaching

dimensions of the order of $N \simeq 10^7$ or greater. Calculating the matrix elements is at least quadratic in the dimension of the matrix, or even higher. This means that the computational cost of direct linear algebra methods, such as direct matrix inversion or eigendecomposition, would be cubic, resulting in a significant increase in computational complexity. As the size of the matrix grows, the computational burden of these direct methods becomes prohibitively large.[9,10].

One of the key applications of linear response property calculations in this context is the computation of excitation energies within the framework of time-dependent density functional theory (TDDFT), which involves solving symplectic eigenvalue problems,

$$\begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B} & \mathbf{A} \end{pmatrix} \begin{pmatrix} \mathbf{X} & \mathbf{Y} \\ \mathbf{Y} & \mathbf{X} \end{pmatrix} = \begin{pmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & -\mathbf{1} \end{pmatrix} \begin{pmatrix} \mathbf{X} & \mathbf{Y} \\ \mathbf{Y} & \mathbf{X} \end{pmatrix} \begin{pmatrix} \mathbf{\Omega} & \mathbf{0} \\ \mathbf{0} & -\mathbf{\Omega} \end{pmatrix} \tag{1.1}$$

Running TDDFT calculations becomes prohibitively expensive with direct methods. Equation 1.1 also arise in time-dependent Hatree-Fock (TDHF)[11], widely known as random phase approximation.

To address this challenge, matrix-free iterative subspace algorithms, like the Krylov subspace methods, have been developed. These methods avoid explicitly forming the entire matrix and instead focus on computing matrix-vector products, which are more efficient for large matrices. By using iterative techniques and constructing appropriate subspace vectors, these methods reduce the computational complexity to linear or even less than linear, making them more suitable for handling extremely large matrices. Starting from an initial vector space $\mathbf{V}$, the desired eigenvectors or solution vectors are expanded into a Krylov subspace[12]:

$$\mathcal{K}_m(\mathbf{A}, \mathbf{V}) = span\{\mathbf{V}, \mathbf{AV}, \mathbf{A}^2\mathbf{V}, ..., \mathbf{A}^{m-1}\mathbf{V}\}. \tag{1.2}$$

where $\mathcal{K}_m$ is the Krylov subspace of dimension m, $\mathbf{A}$ is a subspace operator and $\mathbf{V}$ is a

collection of subspace vectors. The subspace is constructed by iteratively applying the matrix to the vectors in $\mathbf{V}$. By doing so, the method avoids the need to explicitly form the entire matrix, which would be computationally demanding and memory-intensive for large-scale problems. Instead, it focuses on generating the relevant subspace that captures the essential information needed to approximate the solutions efficiently. The general optimization of the problem corresponds to minimizing a functional, F, of the problem with respect to the solution vectors $\mathbf{X}$;

$$\frac{\delta F[\mathbf{X}]}{\delta \mathbf{X}} = 0 \tag{1.3}$$

where the gradient of the functional represent the residual. Residuals above the convergence threshold are preconditioned (using standard preconditioners) and used to extend the subspace in the next iteration until convergence is reached.

Furche *et al.*[13] illustrate the use and performance of nonorthonormal Krylov space (nKs) algorithms to accelerate calculations of these molecular response properties for TDHF and hybrid TDDFT similar to recursive Fock matrix build. This work is based on decreasing residual norms to reduce the cost of the matrix-vector products by using nonorthonormal basis of residual vectors to construct the Krylov space.

Almost all quantum chemistry codes contain at least one implementation of Krylov space methods for computing excitation energies, molecular properties, and/or electron correlation energies. However these are usually hard-coded, resulting in varying convergence behaviour across implementation and poor portability. Additionally, the lack of well-separated matrix-vector product (the most expensive step) steps can hinder computational efficiency.

During my PhD, I have focused on developing, implementing, testing and deploying libkrylov - a robust and efficient open-source library of Krylov space methods suitable for extremely large and dense problems. By creating this library, I aimed to provide a unified and reliable

framework for researchers and developers in the quantum chemistry community. As of part of the libkrylov library I implemented an algorithm that preserve the inherent symmetry in eq 1.1.

In Chapter 2 of my research, I have focused on investigating the accuracy of modeling electronic states using density functional theory (DFT). Through the application of DFT, I have successfully determined the electronic structure of the first Neodymium(II) encapsulated in a 2.2.2-cryptand ligand. The comparison between the experimental results and the calculated molecular structure, as well as the UV-Vis spectrum obtained through TDDFT, provided strong evidence supporting my discovery of the traditional $4f^4$ electron configuration.

Chapter 3 of my research presents the design and implementation of libkrylov. The main objectives of libkrylov are to provide a flexible application programming interface (API) design and a modular structure that allows integration with specialized matrix-vector evaluation "engines." The library offers various features, including pluggable preconditioning, orthonormalization, and tunable convergence control. This allows users to choose different preconditioners such as diagonal (conjugate gradient), Davidson, and Jacobi-Davidson, as well as orthonormal and nonorthonormal (nKs) schemes. The functionality of libkrylov is made accessible through FORTRAN and C APIs.

I focus on exploring structure-preserving methods in Chapter 4, inspired by the successful work of Furche *et al.*, which aimed to accelerate molecular response calculations using a nonorthonormal Krylov subspace method. The objective was to develop and investigate methods that preserve the underlying structure of the TDDFT response problem while improving the efficiency of the calculations.

By leveraging the modular architecture of libkrylov, I sought to enhance the performance of response calculations by incorporating structure-preserving techniques into the Krylov subspace framework. These methods aim to exploit the inherent properties and symmetries of

the problem to reduce computational cost and enhance numerical stability. Through numerical experiments and comparisons with existing approaches, I evaluated the effectiveness of the proposed structure-preserving methods in terms of convergence rate, computational efficiency, and accuracy. Current structure preserving methods like the Olsen[14] and Kauczor[15] algorithms require twice the computational cost compared to the conventional approach. The method presented here is comparable to the conventional approach or even better.

Chapter 2

Explorative DFT Calculations of Neodymium and Samarium (II) in Cyptand Complex

This chapter contains verbatim excerpts from Huh, D. N.; Ciccone, S. R.; Bekoe, S; Roy, S.; Ziller, J. W.; Furche, F.; Evans, W. J. Synthesis of Ln(II)-in-Cryptand Complexes by Chemical Reduction of Ln(III)-in-Cryptand Precursors: Isolation of a Nd(II)-in-Cryptand Complex. *Angew. Chem. Int. Ed.*, **2020**, 59(37), 16141–16146. This work was supported by the National Science Foundation (NSF) under Grant OAC-1835909

Contributions: Ran electronic structure calculations and related analysis, presenting and discussing results with other authors. I also provided interpretation of the computational results and contributed to manuscript and computational details section of supporting information.

2.1 Introduction

Complexation of alkali and alkaline-earth metal ions using the 2.2.2-cryptand (crypt) ligand was first reported in 1973.[16,17] This crypt encapsulation chemistry was extended to lanthanide ions in 1979.[18] However, in the ensuing four decades, only seventeen crystallographically-characterized Ln-in-crypt complexes have been reported in the literature.[18–28] The majority of these complexes involve lanthanides in the common +3 oxidation state and only recently have Ln^{2+}-in-crypt complexes been identified, but only for Ln=Sm, Eu, and Yb.[22–27] These crystallographically-characterized Ln^{2+}-in-crypt complexes were all formed from Ln^{2+} precursors, as shown in the examples in Equation(1) and (2) of Figure 2.1, rather than by reduction of Ln^{3+}-in-crypt precursors.

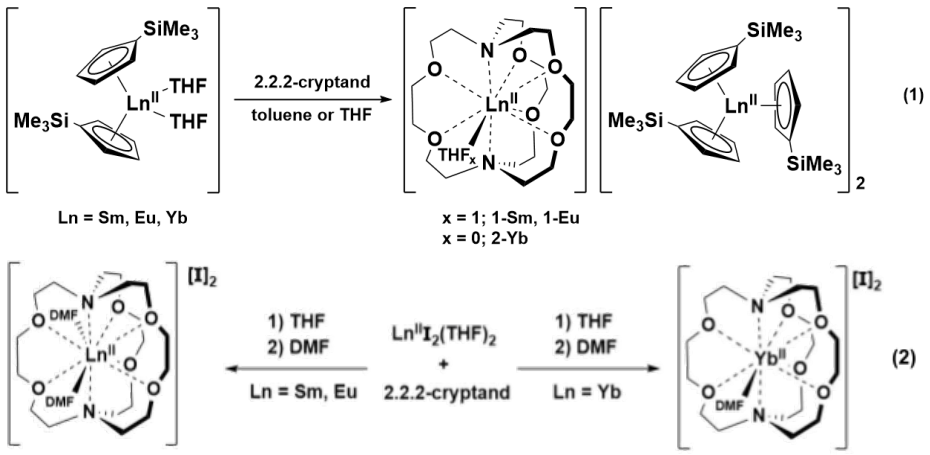

Figure 2.1: Crytand encapsulation of Ln^{2+}

It should be noted that lanthanide ions have been incorporated into encapsulating ligands other than crypt and that the synthetic chemistry is more varied than that described above specifically for 2.2.2-cryptand.[29]

Recently, crypt-encapsulated lanthanide complexes have become of interest because of the discovery of a more extensive Ln^{2+} oxidation state chemistry than was previously thought possible.[30–38] Since previous electrochemical reports by Gansow[39] and Allen[22,27] suggest that

crypt supports lower oxidation states, the crypt coordination environment could allow the isolation of new classes of Ln^{2+} ions[38] or even Ln^{+} compounds.

Ln-in-crypt complexes are of interest in reductive rare-earth chemistry for another reason. The complexes of the new Ln^{2+} ions for La, Ce, Pr, Gd, Tb, Ho, Er, Lu, and Y were often made by alkali metal reduction in the presence of crypt to sequester the alkali metal.[34–38] However, since lanthanides can be encapsulated by crypt, and sometimes unexpectedly, Equation 1,[26] it is important to know the relevant Ln-in-crypt chemistry especially in the presence of other alkali metals.

We were interested in exploring further the reduction chemistry of Ln-in-crypt complexes, but unfortunately the complexes in Equation 2 are insoluble in the ethereal solvents commonly used in the synthesis of reduced lanthanide compounds.[34–38,40–55] The compounds dissolve in dimethylformamide (DMF) to form DMF adducts,[25] but this complicates the reduction chemistry since alkali metal reducing agents can react with DMF.[56] Thus, THF soluble Ln-in-crypt complexes were sought.

2.2 Density Functional Calculation

Density functional theory (DFT) calculations using the TPSSh[57] functional with Grimme's D3 dispersion correction[58] were used to study the $[Nd(crypt)]^{2+}$ complex contained in **1-Nd** and its Sm analogue, $[Sm(crypt)]^{2+}$ in **1-Sm**. Scalar relativistic effective core potentials (ECPs)[59] along with def2-TZVP[60] basis sets were used for Nd and Sm, and polarized split-valence def2-SV(P)[61] basis sets were used for all other atoms. DFT quadrature grids of size 4 were chosen.[62] Solvent effects were accounted for using the continuum solvent model COSMO[63] with a dielectric constant of 7.52[64] and a refractive index of 1.3 to model THF solution. All structures were initially optimized starting from the X-ray structures without

symmetry constraints and with geometry convergence thresholds of 10^{-4} a.u and energy convergence of 10^{-8} a.u. Optimized structures were confirmed to be minima on the potential energy surface by vibrational analysis using finite differences of gradients. Point group symmetry was inferred by re-optimization of the symmetrized structures with symmetry constraints and comparison of the resulting total energy to the one obtained in C_1. Larger augmented def2-SVPD basis sets[65] for ligands and def-TZVP basis sets for Nd and Sm were used for time-dependent density functional theory (TDDFT) calculations of vertical excitation energies and oscillator strengths. UV-Vis spectra were simulated using Gaussian line profiles with a root mean-square width of 0.16 eV, and excitation energies were empirically shifted by 0.2 eV to account for systematic errors inherent in the functional, basis sets, and solvation model. Molecular orbitals and electronic transitions and states were analyzed with VMD[66] and Mulliken population analysis (MPA). Excitations between 400 and 600 nm are largely f-d metal-metal transitions. All calculations were carried out with the TURBOMOLE program suite, Version V-7.4.1.[67]

2.3 Results

$[Sm(crypt)]^{2+}$

Structure optimization of $[Sm(crypt)]^{2+}$ yielded a D_3 symmetric minimum, which is a relatively high symmetry for a 63-atom molecule. The structure is in qualitative agreement with the one obtained from X-ray analysis, except for Sm-O and Sm-N distances that are shorter by 10 and 20 pm, respectively, Table 2.1. This shortening of bond lengths may be due the removal of the OTf-ligands in the computational model, as discussed for the Nd compound below. Inspection of the molecular orbitals revealed a $^7A_2$ ground state with predominant f^6 character of the metal. As shown in the molecular orbital diagram in Figure 2.2, the D_3-

symmetric ligand field splitting is compatible with a stable, non-degenerate ground state. Occupation of the 22 a2 orbital, which has mostly f_z^3 character, is energetically unfavorable due to repulsive interactions with the nitrogen lone pairs pointing along the threefold symmetry axis, Figure 2.3. The 24 a1 lowest unoccupied molecular orbital (LUMO) and the degenerate 40 e LUMO+1 are three-lobe diffuse (Rydberg) orbitals mainly localized in the xy plane as shown in Figure 2.3. The 40e orbital exhibits significant $5d$ character.

Complex	Bond	Bond length
$[Sm(crypt)]^{2+}$	Sm-N	2.704
	Sm-O	2.654
$[Sm(crypt)(OTf)_2]$	Sm-N(crypt)	2.843
	Sm-O1(crypt)	2.651
	Sm-O2(crypt)	2.738
	Sm-O3(crypt)	2.819
	Sm-OTf	2.531

Table 2.1: Relevant metal-ligand distances from the structures optimized in D_3 symmetry for $[Sm(crypt)]^{2+}$.

The computed excitation spectrum for $[Sm(crypt)]^{2+}$ shows no bands in the lower-energy visible part of the spectrum. In agreement with the experimental data, the first peak is predicted around 500 nm. This relatively intense transition with an oscillator strength of 0.005 is mostly due to a $4f \rightarrow 5d$ excitation from the occupied 39 e into the unoccupied 40 e orbitals, compare Figures 2.3-2.5. Higher energy transitions involve excitations from increasingly lower-lying $4f$ metal orbitals.

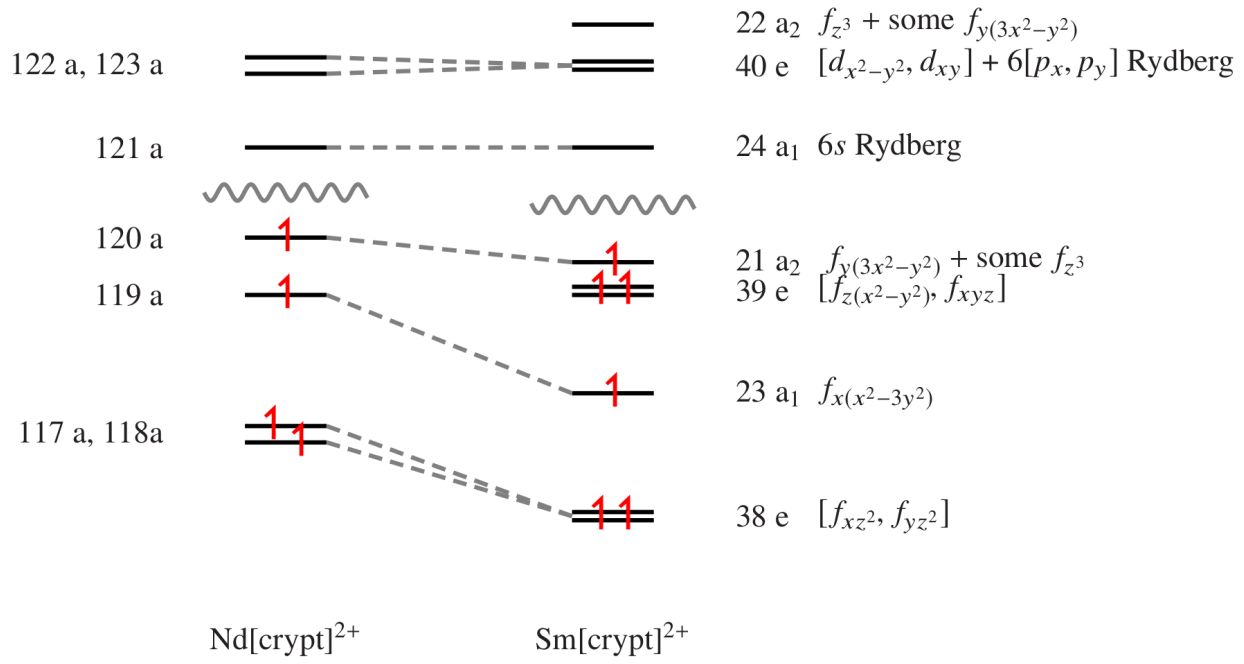

Figure 2.2: Qualitative molecular orbital diagram for the ^{5}A ground state of $[Nd(crypt)]^{2+}$ and the $^7A_2$ ground state of $[Sm(crypt)]^{2+}$ showing the α spin frontier orbitals (not to scale). Other unoccupied orbitals with Nd 4f character are higher in energy and are not displayed here. The corresponding molecular orbital labels in C_1 (Nd) and D_3 (Sm) energy are included on the left and right, respectively

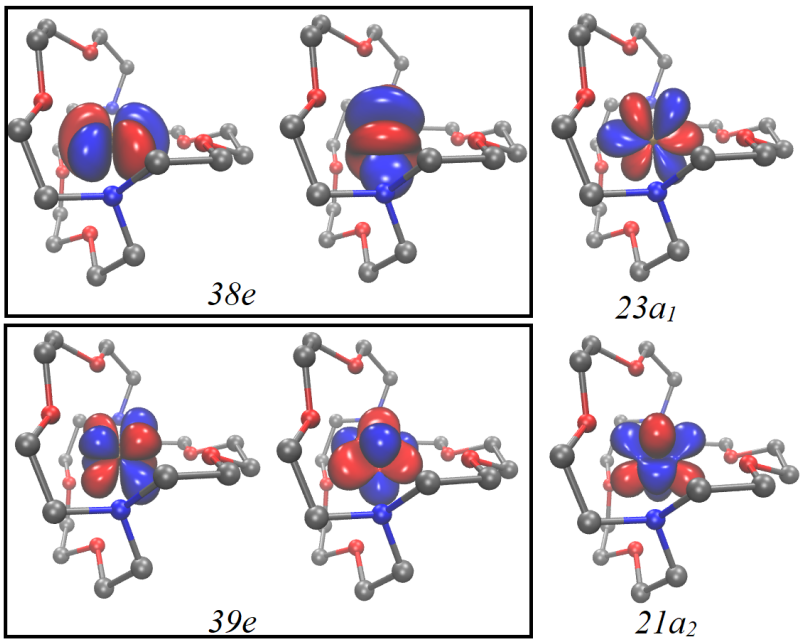

Figure 2.3: Highest occupied molecular orbitals of $[Sm(crypt)]^{2+}$ plotted with contour values of ± 0.03.

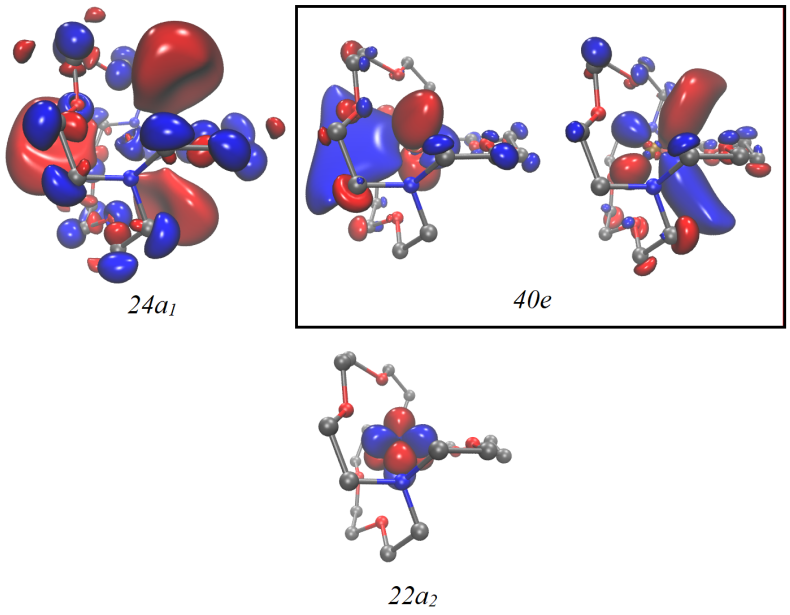

Figure 2.4: Lowest virtual molecular orbitals of $[Sm(crypt)]^{2+}$ plotted with contour values of ± 0.02.

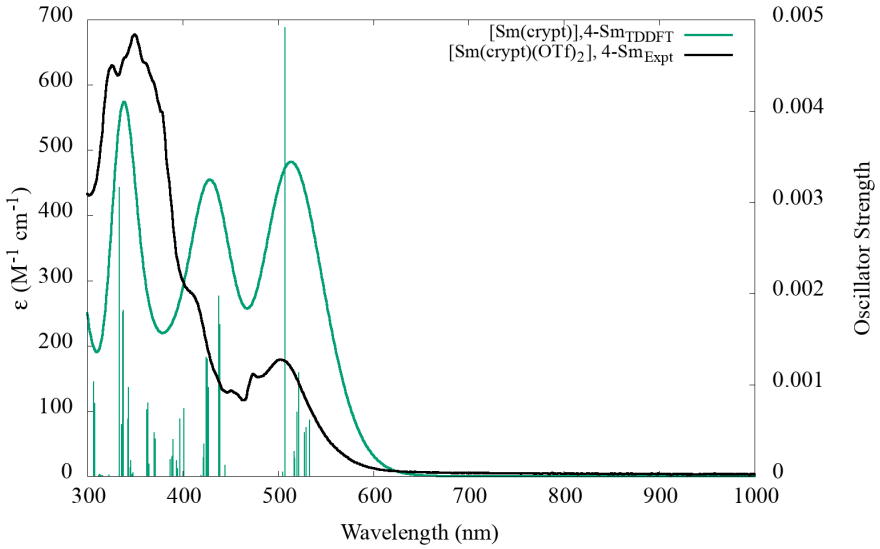

Figure 2.5: Simulated UV-visible spectrum of $[Sm(crypt)]^{2+}$ (dashed) with computed TDDFT oscillator strengths shown as vertical lines. The experimental spectrum is shown as a black solid trace for comparison. A Gaussian line broadening of 0.16 eV was applied and the computed excitation energies were empirically blue-shifted by 0.2 eV. The computed intensities were scaled by a factor 0.3 to ease comparison with the experimental spectrum.

$[Nd(crypt)]^{2+}$

Structure optimization of $[Nd(crypt)]^{2+}$ resulted in a D_3-symmetric structure exhibiting a small second-order Jahn-Teller distortion into C_2 and an even smaller additional distortion

into C_1 symmetry when solvation effects were included. Since the energy gain due to distortion is only 3 kcal/mol, the electron configuration of $[Nd(crypt)]^{2+}$ can still be qualitatively understood starting in D_3 symmetry, see Figure 2.2: Compared to the Sm complex, the 39 e orbitals remain unoccupied in $[Nd(crypt)]^{2+}$, corresponding to a $^5A_2$ ground state in D_3 symmetry with an f^4 electronic configuration of the metal. Second-order Jahn-Teller mixing of the now empty 39 e with the HOMO causes further distortion into a C_1 symmetric minimum corresponding to a ^5A ground state term.

Similar to the Sm case, the optimized structure shows good agreement with the X-ray analysis, except for the Nd-O and Nd-N distances, which are shorter by 8 and 16 pm respectively, Table 2.2. To confirm that this shortening of metal-ligand bond lengths is caused by the removal of OTf- ligands in the computational model, an additional geometry optimization including the OTf^- ions was performed starting from the X-ray structure. This led to a minimum with Nd-O and Nd-N bond lengths within 10 pm of the crystal structure data Table S10. The shortening of the Nd-O and Nd-N distances in $[Nd(crypt)]^{2+}$ are consistent with a decrease in coordination number of the complex.[68] However, the OTf- ligands may partially dissociate in solution, and the more symmetric $[Nd(crypt)]^{2+}$ ion may be a better model for the average solution structure. Moreover,charged ligands may give rise to charge-transfer transitions with spuriously low energy in a TDDFT treatment[69,70] contaminating the simulated spectra. Explorative TDDFT calculations including the OTf- ligands indeed showed much larger intensities in the visible region than experimentally observed.

Compared to the UV-visible spectrum of $[Sm(crypt)]^{2+}$, the UV-visible spectrum of $[Nd(crypt)]^{2+}$ shows transitions throughout the visible region, see Figure 2.8 and Table 2.4. The lower-energy onset of the characteristic $4f \rightarrow 5d$ transitions is consistent with the higher energy of occupied $4f$ orbitals in Nd vs. Sm. Moreover, the relatively intense 39 e$\rightarrow$40 e transition is absent in the Nd compound because 39 e is unoccupied. The broader, less structured spectrum of $[Nd(crypt)]^{2+}$ is consistent with lower C_1 symmetry compared to the D_3-symmetric

Complex	Bond	Bond length
$[Nd(crypt)]^{2+}$	Nd-N	2.758
	Nd-O1	2.669
	Nd-O2	2.670
$[Nd(crypt)(OTf)_2]$	Nd-N(crypt)	2.857
	Nd-O1(crypt)	2.668
	Nd-O2(crypt)	2.737
	Nd-O3(crypt)	2.825
	Nd-OTf	2.574

Table 2.2: Relevant metal-ligand distances from the structures optimized in C_1 symmetry for $[Nd(crypt)]^{2+}$ and $[Nd(crypt)(OTf)_2]$.

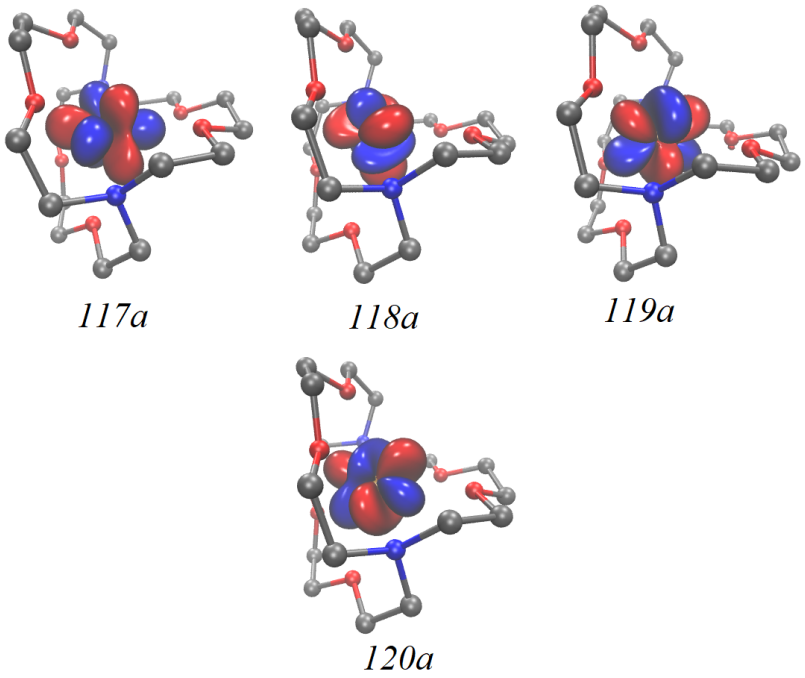

Figure 2.6: Highest occupied molecular orbitals of $[Nd(crypt)]^{2+}$ plotted with contour values of ± 0.03.

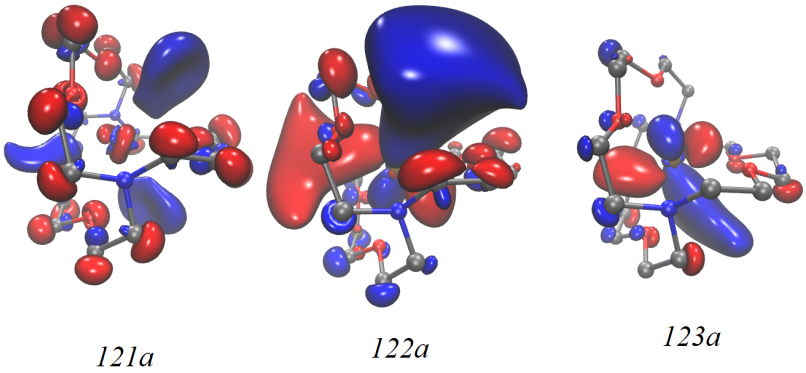

Figure 2.7: Lowest virtual molecular orbitals of $[Nd(crypt)]^{2+}$ plotted with contour values of ± 0.02.

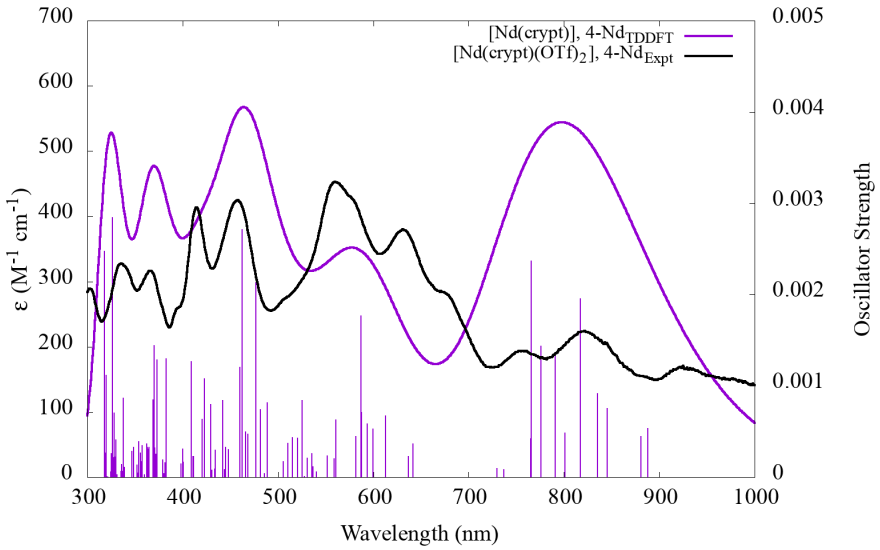

Figure 2.8: Simulated UV-visible spectrum of $[Nd(crypt)]^{2+}$ (dashed) with computed TDDFT oscillator strengths shown as vertical lines. The experimental spectrum is shown as a black solid trace for comparison. A Gaussian line broadening of 0.16 eV was applied and the computed excitation energies were empirically blue-shifted by 0.2 eV. The computed intensities were scaled by a factor 0.3 to ease comparison with the experimental spectrum.

Sm compound.

In conclusion, the agreement between the experimental and the computed spectra, both regarding intensities and positions and shape of the bands, corroborates the assignment of f^6 and f^4 electronic ground state configurations for the 1-Sm and 1-Nd compounds.

2.4 Supplementary Information

Term Symbol	Wavelength (nm)	Oscillator (len)	Occupied	Virtual	% Weight	Assignment
5 ^{7}E	507.5	0.005	39e	40e	27.5	$4f_{xyz}\rightarrow(6p_y+5d_{xy})$RB
				40e	20.9	$4f_{z(x^2-y^2)}\rightarrow(6p_y+5d_{(x^2-y^2)})$RB
				40e	18.1	$4f_{xyz}\rightarrow(6p_y+5d_{(x^2-y^2)})$RB
				41e	10.3	$4f_{xyz}\rightarrow(5d_{yz}+5d_{(x^2-y^2)})$RB
				41e	7.5	$4f_{z(x^2-y^2)}\rightarrow(5d_{yz}+5d_{(x^2-y^2)})$RB
				40e	6.7	$4f_{z(x^2-y^2)}\rightarrow(6p_y+5d_{xy})$RB
7 ^{7}E	437.7	0.002	23a_1	40e	54.4	$4f_{x(x^2-3y^2)}\rightarrow(6p_y+5d_{(x^2-y^2)})$RB
				41e	13.9	$4f_{x(x^2-3y^2)}\rightarrow(5d_{yz}+5d_{(x^2-y^2)})$RB
			21a_2	40e	7.1	$4f_{y(3x^2-y^2)}\rightarrow(6p_y+5d_{xy})$RB
8 ^{7}E	424.6	0.001	21a_2	41e	59.8	$4f_{y(3x^2-y^2)}\rightarrow(5d_{yz}+5d_{(x^2-y^2)})$RB
				40e	14.8	$4f_{y(3x^2-y^2)}\rightarrow(6p_y+5d_{xy})$RB
18 ^{7}E	333.7	0.003	38e	41e	47.0	$4f_{yz^2}\rightarrow(5d_{yz}+5d_{(x^2-y^2)})$RB
				40e	13.6	$4f_{yz^2}\rightarrow(6p_y+5d_{xy})$RB
			39e	42e	8.0	$4f_{xyz}\rightarrow$(some 5d)RB

Table 2.3: Summary of $[Sm(crypt)]^{2+}$. The TPSSh functional along with def2-SVPD basis set for ligand atoms were used. Oscillator strengths are reported in the length gauge. All excitations obtained in this spectral range are alpha spin to alpha spin transitions. RB stands for Rydberg orbitals.

Term Symbol	Wavelength (nm)	Oscillator (len)	Occupied	Virtual	% Weight	Assignment
13 ^{5}A	817.1	0.002	118a	122a	39.1	$4f_{yz^2}\rightarrow(6p_x+5d_{xy})$RB
				126a	17.6	$4f_{x(x^2-3y^2)}\rightarrow(6p_x)$RB
				123a	6.4	$4f_{x(x^2-3y^2)}\rightarrow(6p_y+5d_{(x^2-y^2)})$RB
17 ^{5}A	765.9	0.002	120a	121a	36.4	$4f_{y(3x^2-y^2)}\rightarrow(6s)$RB
			119a	121a	15.2	$4f_{x(x^2-3y^2)}\rightarrow(6s)$RB
			118a	123a	9.4	$4f_{yz^2}\rightarrow(6p_y+5d_{(x^2-y^2)})$RB
			117a	122a	9.0	$4f_{xz^2}\rightarrow(6p_x+5d_{xy})$RB
27 ^{5}A	586.8	0.002	117a	126a	12.7	$4f_{xz^2}\rightarrow(6p_x)$RB
			117a	122a	12.0	$4f_{xz^2}\rightarrow(6p_x+5d_{xy})$RB
			120a	126a	9.8	$4f_{y(3x^2-y^2)}\rightarrow(6p_x)$RB
			118a	126a	6.7	$4f_{yz^2}\rightarrow(6p_x)$RB
			119a	126a	6.4	$4f_{x(x^2-3y^2)}\rightarrow(6p_x)$RB
45 ^{5}A	477.0	0.002	120a	129a	18.9	$4f_{y(3x^2-y^2)}\rightarrow(5d_{yz}+5d_{xz})$RB
				128a	16.8	$4f_{y(3x^2-y^2)}\rightarrow(5d_{xy})$RB
				123a	14.1	$4f_{y(3x^2-y^2)}\rightarrow(6p_y+5d_{(x^2-y^2)})$RB
				149a	9.2	$4f_{y(3x^2-y^2)}\rightarrow$ligand
				134a	6.8	$4f_{y(3x^2-y^2)}\rightarrow(5d_{yz})$RB
48 ^{5}A	462.6	0.003	118a	128a	22.3	$4f_{yz^2}\rightarrow(5d_{xy})$RB
				133a	10.7	$4f_{yz^2}\rightarrow(5d_{z^2})$RB
				134a	7.2	$4f_{yz^2}\rightarrow(5d_{yz})$RB
				129a	6.0	$4f_{yz^2}\rightarrow(5d_{yz}+5d_{xz})$RB
70 ^{5}A	383.0	0.001	120a	147a	61.9	$4f_{y(3x^2-y^2)}\rightarrow$ligand
78 ^{5}A	373.3	0.001	118a	147a	57.5	$4f_{yz^2}\rightarrow$ligand
113 ^{5}A	326.1	0.003	120a	145a	49.9	$4f_{yz^2}\rightarrow$ligand
				146a	21.3	$4f_{yz^2}\rightarrow$ligand
				149a	14.8	$4f_{yz^2}\rightarrow$ligand

Table 2.4: Summary of $[Nd(crypt)]^{2+}$. The TPSSh functional along with def2-SVPD basis set for ligand atoms were used. Oscillator strengths are reported in the length gauge. All excitations obtained in this spectral range are alpha spin to alpha spin transitions. RB stands for Rydberg orbitals.

Chapter 3

Libkrylov, a Modular Open-Source Software Library for Extremely Large On-the-Fly Matrix Computations

This chapter contains verbatim excerpts from Rappoport, D., Bekoe, S., Mohanam, L. N., Le, S., George, N., Shen, Z., Furche, F. Libkrylov: A modular open-source software library for extremely large on-the-fly matrix computations. J. Comput. Chem. 2023, 44(11), 1105. This work was supported by the National Science Foundation (NSF) under Grant OAC-1835909 and The Molecular Sciences Software Institute (MolSSI) under NSF grant CHE-2136142.

Contribution: I was involved in discussion and design of library interface. I implemented the preconditioning objects. Additionally I contributed to implementing the global and linear algebra utility functions in the library and testing library functions. I wrote user guide and gitlab wiki.

3.1 Introduction

Linear equations and eigenvalue problems with large dimensionality, on the order of $> 10^6$ degrees of freedom, and no special structure or sparsity arise in many scientific and engineering applications, for example, quantum chemical and materials science simulations, partial differential equation solvers, signal reconstruction, and machine learning applications.[71–73] Density functional calculations of light-induced processes in organic and semiconductor nanostructures have pushed the limit of quantum chemical studies to systems with 1000 or more atoms. Molecular and material property calculations in these systems amount to solving eigenvalue problems of dimension one billion or larger.[74,75] In many critical applications, the coefficient matrices are extremely large, dense, and full rank (due to strong coupling/interaction), precluding the use of specialized techniques such as sparse solvers or factorizations.

Matrix-free iterative methods eliminate the bottleneck of explicitly computing and storing extremely large and dense coefficient matrices; instead, products of the coefficient matrix with trial vectors are computed "on the fly". These solvers are based on *Krylov subspace* methods and solve the linear problems such as eigenvalue problems, linear equations, or related problems involving the coefficient matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ or $\in \mathbb{C}^{n\times n}$ on the sequence of Krylov subspaces $\mathcal{K}^{(k)}$ of increasing dimension q_k, $k = 1, 2, \ldots$.[76–82] For many linear problems the iteration converges after K iterations with $q_K \ll n$ and is very economic in its CPU and memory usage even in the case of very large matrices $\mathbf{A}$. The key component of Krylov subspace algorithms is the matrix–vector multiplication "engine", which is capable of efficiently computing $\mathbf{Av}$, given a vector $\mathbf{v}$. These engines are highly application-dependent and often exploit special intermediate representations and/or domain-specific physical properties.[83–92]

The importance of Krylov subspace methods for the development of efficient quantum chemical implementations is hard to overstate. Implementations of these methods for molecu-

lar property calculations, convergence acceleration of ground-state energy calculations, and other computationally intensive tasks are part of nearly all molecular and solid-state electronic structure codes, many of which were included in a recent software overview.[93] In particular, the Davidson algorithm[94–96] and its variants are behind some of the largest quantum chemical simulations.[97–99] Matrix-free methods are particularly attractive on massively parallel computing architectures, where (re)computing matrix elements is preferable to the storage and communication of the full coefficient matrix **A**.

While the theory and algorithms of Krylov subspace methods have a wealth of literature[76–82,100,101], the corresponding matrix-free implementations are scattered over many software packages and are often narrowly tailored to the solution of a specific problem.

The integration of the optimized matrix–vector multiplication "engines" with the Krylov subspace algorithms and efficient data interchange are crucial to the efficiency of the overall implementation. These requirements rule out general-purpose linear algebra environments such as MATLAB[102] and its open-source clone Octave[103] due to the computational overhead of repeated data transformations. The existing ecosystem of open-source libraries for iterative linear algebra primarily targets problems with an explicit but sparse coefficient matrix **A** and does not fit the needs of matrix-free linear algebra problems, for example those arising in molecular property calculations, with regard to their functionality or their data interface.

The seminal ARPACK library[104] lacks the implementation of the Davidson algorithm or its variants and has not seen significant development in many years. The popular and efficient BLOPEX library is limited to solving ordinary and generalized eigenvalue problems using the LOBPCG algorithm.[105,106] The SLEPc library is primarily designed for sparse matrix computations but can also accept matrix–vector functions[107,108] instead of explicit matrices. The PETSc library also provides this functionality for solving large partial differential equations.[109]

The many custom implementations of Krylov subspace algorithms with essentially overlapping functionalities resulted in considerable duplication of programming, debugging, and testing effort. Moreover, each implementation is somewhat different in its functionality, configuration options, and numerical thresholds. This results in difficulties in comparing and reproducing results from different codes. Finally, in the absence of well-defined application programming interfaces (APIs), even the existing open-source implementations cannot easily interoperate.

A further complicating factor is that the speed of convergence of iterative methods depends on specific properties of the coefficient matrix, in particular diagonal dominance and the distribution of its eigenvalues. Suitably chosen preconditioning techniques can thus drastically improve convergence of Krylov subspace methods by taking advantage of these properties. In addition, the non-orthonormal Krylov subspace (nKs) approach[75] can exploit the decreasing norms of the vectors added to Krylov subspaces as the iteration progresses to further reduce the computational cost of evaluating matrix–vector products. A related "balancing" approach for the Krylov subspace methods has been proposed.[110] Because the performance characteristics of Krylov subspace methods are structure-dependent, it is desirable to provide the user code with the flexibility to choose the preconditioning and orthonormalization methods best suited for the specific numerical problem.

In this contribution, we describe *libkrylov*[111], an extensible software library for orthogonal and non-orthogonal matrix-free Krylov subspace methods, which features flexible APIs and enables pluggable preconditioning, orthonormalization, and tunable convergence control. The key design objective of libkrylov is to balance the integration of the optimized matrix–vector multiplication and the Krylov subspace algorithms with the simplicity and flexibility of use.Libkrylov is designed as a framework, which inverts the control flow compared to that of typical libraries. While library functions are designed to be called from a driver procedure, the libkrylov solver function is responsible for executing the control flow, calling

the user-provided matrix–vector product function in each iteration via a fixed API. This approach ensures the necessary flexibility of the libkrylov components without compromising the efficiency of the implementation, while the user code is only responsible for implementing a matrix–vector multiplication function and can focus on domain-specific optimizations. The libkrylov library is implemented in portable Fortran 2003/C99 and aims to be an off-the-shelf library component for large-scale scientific and engineering applications. To this end, libkrylov is designed with maximal modularity, structured Fortran and C interfaces, and integration with platform-optimized BLAS[112–114] and LAPACK[115] linear algebra primitives in mind. Libkrylov is distributed under the open source 3-clause BSD license.

The version 1.1.0 of libkrylov treats symmetric eigenvalue problems, linear equations, and shifted linear equations. Diagonal (conjugate-gradient, CG) preconditioning, Davidson preconditioner[94–96], and Jacobi–Davidson preconditioning due to Sleijpen and van der Vorst[116,117] can be selected for convergence acceleration. Both the most commonly used orthonormal Krylov subspace algorithm and the computationally efficient nKs approach of Furche and co-workers[75] are available. Moreover, libkrylov supports blocked algorithms for simultaneous iteration of multiple equations with a shared coefficient matrix for linear problems with block-diagonal structure. *A posteriori* error bounds and dynamic restart capability cover the most common application scenarios.

This paper describes the design strategy of libkrylov and outlines its approach to solving a wide range of matrix-free linear problems via a common function-based interface. The structure of the paper is as follows. We first introduce the notation and give a brief overview of Krylov subspace algorithms implemented in libkrylov. After that, we describe the design and implementation of libkrylov. Finally, using an illustrative set of linear problems related to molecular property calculations, we review the suite of preconditioning and orthonormalization techniques available in libkrylov and give examples of their convergence.

3.2 Krylov Subspace Methods

3.2.1 Background and Notation

In this section we establish our notation for Krylov subspace methods for eigenvalue problems, linear equations, and shifted linear equations for symmetric coefficient matrices $\mathbf{A} \in \mathbb{R}^{n\times n}$ or Hermitian matrices in $\mathbb{C}^{n\times n}$. Since many applications seek $p \geq 1$ simultaneous solutions involving the same coefficient matrix $\mathbf{A}$, for example, the p lowest eigenvalues, right-hand sides (RHS) and/or shifts, we present the Krylov subspace algorithms in their blocked form and use matrix notation throughout. Table 3.1 gives an overview of the linear problems treated here. $\mathbf{X}$ denotes the $n \times p$ matrix having solution vectors as columns. For eigenvalue problems, $\mathbf{X}$ is the matrix of eigenvectors. The corresponding eigenvalues are represented by the diagonal matrix $\mathbf{\Omega} = \mathrm{diag}(\Omega_1, \ldots, \Omega_p)$. The RHS of the (shifted) linear equations are combined into the $n \times p$ matrix $\mathbf{P}$. Shifted linear equations additionally contain a matrix of diagonal shifts $\boldsymbol{\omega} = \mathrm{diag}(\omega_1, \ldots, \omega_p)$. These equations can be considered as the diagonal variant of the Sylvester equations.[118]

In Krylov subspace methods, the k-th iterates ($k \geq 1$) are written as vectors in the Krylov subspace $\mathcal{K}^{(k)}$ of dimension q_k,

$$\mathbf{X}^{(k)} = \mathbf{V}^{(k)}\mathbf{x}^{(k)}, \tag{3.1}$$

where $\mathbf{V}^{(k)}$ is the matrix of basis vectors of $\mathcal{K}^{(k)}$ and $\mathbf{x}^{(k)}$ are the expansion coefficients. The residual (error) vectors of the k-th iterates are collected in the matrix $\mathbf{R}^{(k)}$, see Table 3.1 for definitions. The k-th iterates are chosen according to the Ritz–Galerkin condition, that is, the residual vectors are required to be orthogonal to the Krylov subspace $\mathcal{K}^{(k)}$,

$$\mathbf{V}^{(k)\dagger}\mathbf{R}^{(k)} = \mathbf{0}. \tag{3.2}$$

Note that superscript † indicates the matrix transpose for real matrices and the Hermitian conjugate in the complex case. In this case, $\mathbf{X}^{(k)}$ are the Ritz vectors, and the corresponding expansion coefficients $\mathbf{x}^{(k)}$ follow from solving the projected linear problem on $\mathcal{K}^{(k)}$, as shown in Table 3.1. The nKs method[75] uses a nonorthonormal subspace basis, in which case the Gram (overlap) matrix of the subspace basis is given by

$$\mathbf{s}^{(k)} = \mathbf{V}^{(k)\dagger}\mathbf{V}^{(k)} \tag{3.3}$$

The Rayleigh matrix

$$\mathbf{a}^{(k)} = \mathbf{V}^{(k)\dagger}\mathbf{A}\mathbf{V}^{(k)} \tag{3.4}$$

is the projection of the matrix $\mathbf{A}$ onto the Krylov subspace. In eigenvalue problems, the diagonal matrix $\mathbf{\Omega}^{(k)}$ contains the k-th eigenvalue iterates. The projection of the RHS matrix $\mathbf{P}$ of (shifted) linear equations is denoted as

$$\mathbf{p}^{(k)} = \mathbf{V}^{(k)\dagger}\mathbf{P} \tag{3.5}$$

The columns of the residual matrix $\mathbf{R}^{(k)}$ are used to construct the basis of the subsequent Krylov subspace $\mathcal{K}^{(k+1)}$ of dimension $q_{k+1} = q_k + p$. The basic Krylov subspace iteration in libkrylov (no preconditioner) adds the residuals to the Krylov subspace basis, similar to Arnoldi iteration.[119] However, most applications apply a suitably chosen invertible $n \times n$ matrix preconditioning matrix $\mathbf{K}^{(k)}$ to the residuals,

$$\tilde{\mathbf{R}}^{(k)} = \left(\mathbf{K}^{(k)}\right)^{-1}\mathbf{R}^{(k)}\,, \tag{3.6}$$

prior to expanding the Krylov subspace. Preconditioning techniques are crucial to the performance of Krylov subspace methods for linear equations.[76,81,120–123] In eigenvalue problems,

	Equation	Projected Equation	Residuals
Eigenvalue problem	$\mathbf{AX} = \mathbf{\Omega X}$	$\mathbf{a}^{(k)}\mathbf{x}^{(k)} = \mathbf{s}^{(k)}\mathbf{x}^{(k)}\mathbf{\Omega}^{(k)}$	$\mathbf{R}^{(k)} = \mathbf{AX}^{(k)} - \mathbf{X}^{(k)}\mathbf{\Omega}^{(k)}$
	$\mathbf{X}^\dagger\mathbf{X} = \mathbf{1}$	$\mathbf{x}^{(k)\dagger}\mathbf{s}^{(k)}\mathbf{x}^{(k)} = \mathbf{1}$	
Linear equation	$\mathbf{AX} = \mathbf{P}$	$\mathbf{a}^{(k)}\mathbf{x}^{(k)} = \mathbf{p}^{(k)}$	$\mathbf{R}^{(k)} = \mathbf{AX}^{(k)} - \mathbf{P}$
Shifted linear equation	$\mathbf{AX} - \mathbf{X}\boldsymbol{\omega} = \mathbf{P}$	$\mathbf{a}^{(k)}\mathbf{x}^{(k)} - \mathbf{s}^{(k)}\mathbf{x}^{(k)}\boldsymbol{\omega} = \mathbf{p}^{(k)}$	$\mathbf{R}^{(k)} = \mathbf{AX}^{(k)} - \mathbf{X}^{(k)}\boldsymbol{\omega} - \mathbf{P}$

Table 3.1: Linear problems solved by libkrylov, the corresponding projected equations on Krylov subspace $\mathcal{K}^{(k)}$, and definitions of the residual matrices. See text for definitions.

Davidson[94,96] and Jacobi–Davidson (JD) preconditioning[116,117] are widely used convergence acceleration techniques.

3.2.2 Krylov Subspace Algorithm

Given the $q_1 \times n$ matrix of starting vectors $\mathbf{V}^{(1)}$ and matrix–vector multiplication function $f(\mathbf{V}) = \mathbf{AV}$, the Krylov subspace algorithm takes the steps outlined in Fig. 3.1. Note that matrix–vector products and other quantities involving the columns $\mathbf{V}^{(k)}$ can be reused from previous iterations so that $f(\mathbf{V})$ is only evaluated for new vectors. The step denoted by (*) is only needed for nonorthonormal subspace bases, see below. In the orthonormal case, the modified Gram–Schmidt (MGS) method[118] is applied in the step (‡). The projection (†) is performed for (shifted) linear equations only.

3.2.3 Preconditioning

The convergence of the basic Krylov subspace iteration is often unsatisfactory for practical calculations. The choice of the preconditioning matrix $\mathbf{K}$ can significantly affect the speed of convergence.[76–78,80,81,100,101] The diagonal approximation $\mathbf{D} = \mathrm{diag}(A_{11}, \ldots, A_{nn})$ to the coefficient matrix,

$$\mathbf{K}_{\mathrm{CG}}^{(k)} = \mathbf{D}\,, \tag{3.7}$$

Choose $q_1 \geq p$ starting vectors $\mathbf{V}^{(1)}$
for $k = 1, 2, \ldots$
$\mathbf{W}^{(k)} = f(\mathbf{V}^{(k)}) = \mathbf{A}\mathbf{V}^{(k)}$
$\mathbf{s}^{(k)} = \mathbf{V}^{(k)\dagger}\mathbf{V}^{(k)}$ (*)
$\mathbf{a}^{(k)} = \mathbf{V}^{(k)\dagger}\mathbf{W}^{(k)}$
$\mathbf{p}^{(k)} = \mathbf{V}^{(k)\dagger}\mathbf{P}$ (†)
Compute $\mathbf{x}^{(k)}$ from projected equation (see Table 3.1)
$\mathbf{X}^{(k)} = \mathbf{V}^{(k)}\mathbf{x}^{(k)}$
Compute residuals $\mathbf{R}^{(k)}$ (see Table 3.1)
if $\max_{1\leq i\leq p}|\mathbf{R}_i^{(k)}| \leq \tau$ **then** quit
$\tilde{\mathbf{R}}^{(k)} = \mathbf{K}^{(k)-1}\mathbf{R}^{(k)}$
Orthogonalize $\tilde{\mathbf{R}}^{(k)}$ against $\mathbf{V}^{(k)}$ (‡)
$\mathbf{V}^{(k+1)} = [\mathbf{V}^{(k)} \;\; \tilde{\mathbf{R}}^{(k)}]$
end

Figure 3.1: Schematic Krylov subspace iteration algorithm including preconditioning and orthonormalization. $f(\mathbf{V}^{(k)})$ is the matrix–vector product evaluation by the user-supplied function. The step denoted by (*) is only used in nonorthonormal Krylov subspace algorithm. The orthonormal case includes the vector orthonormalization step (‡). The projection (†) is performed for (shifted) linear equations only. τ is the convergence threshold.

is often used in the preconditioned CG algorithm.[76,77,81] For eigenvalue problems, the Davidson algorithm corresponds to choosing the preconditioner as

$$\mathbf{K}_{\mathrm{D}}^{(k)} = \mathbf{D} - \boldsymbol{\Omega}^{(k)}\,, \tag{3.8}$$

where $\boldsymbol{\Omega}^{(k)}$ are the k-th eigenvalue iterates.[94,96] For shifted linear equations, the equivalent of the Davidson preconditioner includes the diagonal shifts $\boldsymbol{\omega}$ instead of eigenvalues.

The JD algorithm ensures by projection that the preconditioned residuals are orthogonal to the solution,[116,117]

$$\mathbf{K}_{\mathrm{JD}}^{(k)} = (\mathbf{1} - \Pi^{(k)})\mathbf{K}_{\mathrm{D}}^{(k)}(\mathbf{1} - \Pi^{(k)})\,, \tag{3.9}$$

where $\Pi^{(k)}$ is the projector onto the set of k-th iterates. By imposing orthogonality con-

straints, the preconditioned residuals are computed as

$$\tilde{\mathbf{R}}^{(k)} = \left(\mathbf{K}_{\mathrm{D}}^{(k)}\right)^{-1} \mathbf{R}^{(k)} - \varepsilon^{(k)} \left(\mathbf{K}_{\mathrm{D}}^{(k)}\right)^{-1} \mathbf{X}^{(k)} \quad \text{with} \quad \varepsilon^{(k)} = \frac{\mathbf{X}^{(k)\dagger} \left(\mathbf{K}_{\mathrm{D}}^{(k)}\right)^{-1} \mathbf{R}^{(k)}}{\mathbf{X}^{(k)\dagger} \left(\mathbf{K}_{\mathrm{D}}^{(k)}\right)^{-1} \mathbf{X}^{(k)}} . \quad (3.10)$$

In the case of $p > 1$, two variants of the JD preconditioner can be formulated. Each residual vector can be preconditioned by projecting out only the corresponding solution (JD variant 1) or all solutions (JD variant 2).

3.2.4 Nonorthonormal Subspace Bases

The nKs method takes advantage of the fact that the residual norms $\mathbf{R}^{(k)}$ usually decrease as the approximate solutions $\mathbf{X}^{(k)}$ converge to the true solution $\mathbf{X}$. If the function $f(\mathbf{V}) = \mathbf{AV}$ can be made to execute more efficiently for small $\|\mathbf{V}\|$, for example, by prescreening the matrix elements of $\mathbf{A}$, then the cost of each iteration may be reduced by up to 80%[75]. In order to preserve the decreasing residual norms, the orthonormalization step (‡) is omitted in the nKs method, and instead the projected equations are solved in their generalized form with a Gram matrix $\mathbf{s}^{(k)} \neq \mathbf{1}$ (Table 3.1).

The projected equations are solved in two steps. First, the equation is scaled by the diagonal $\mathbf{d}$ of $\mathbf{s}^{(k)}$,

$$\mathbf{d} = \mathrm{diag}(s_{11}^{(k)}, \ldots, s_{q_k q_k}^{(k)}) \quad (3.11)$$

to reduce the condition number of $\mathbf{s}$. Using the Cholesky decomposition of the scaled Gram matrix

$$\mathbf{L}\mathbf{L}^{\dagger} = \mathbf{d}^{-1/2} \mathbf{s}^{(k)} \mathbf{d}^{-1/2} \quad (3.12)$$

the following substitutions can be made

$$\begin{aligned}\tilde{\mathbf{a}}^{(k)} &= \mathbf{L}^{-1}\mathbf{d}^{-1/2}\mathbf{a}^{(k)}\mathbf{d}^{-1/2}(\mathbf{L}^{-1})^{\dagger},\\ \tilde{\mathbf{x}}^{(k)} &= \mathbf{L}^{\dagger}\mathbf{d}^{1/2}\mathbf{x}^{(k)},\\ \tilde{\mathbf{p}}^{(k)} &= \mathbf{L}^{-1}\mathbf{d}^{-1/2}\mathbf{p}^{(k)}.\end{aligned} \tag{3.13}$$

The resulting linear problem can be then solved using standard methods. For more details, see Ref.[75].

If $p > 1$ equations are simultaneously iterated, the (preconditioned) residuals corresponding to different solutions may become linearly dependent. To improve the condition of the Gram matrix while maintaining the advantage of decreasing residual norms of the nKs approach, we test a variant of the nKs approach. In this *semiorthonormal* variant, instead of the full orthonormalization (‡) in Fig. 3.1, we limit ourselves to enforcing orthogonality between the columns of $\tilde{\mathbf{R}}^{(k)}$ using a singular-value decomposition (SVD),

$$\begin{aligned}\mathbf{u}^{(k)\dagger}\boldsymbol{\sigma}^{(k)}\mathbf{v}^{(k)} &= \left(\mathbf{K}^{(k)}\right)^{-1}\mathbf{R}^{(k)},\\ \tilde{\mathbf{R}}^{(k)} &= \boldsymbol{\sigma}^{(k)}\mathbf{v}^{(k)}.\end{aligned} \tag{3.14}$$

For $p = 1$, the semiorthonormal variant is identical to the nKs method.[75]

Error Bounds and Convergence of the Algorithm

The norm of the residual matrix $\mathbf{R}^{(k)}$ provides a strict upper bound for the error of the approximate eigenvalues in the k-th iteration[118],

$$|\Omega_i - \Omega_i^{(k)}| \leq \sqrt{2}\|\mathbf{R}^{(k)}\|_2 \quad \text{for } 1 \leq i \leq q_k\,, \tag{3.15}$$

where the 2-norm of the residual matrix is used. Because

$$\|\mathbf{R}^{(k)}\|_2 \leq \|\mathbf{R}^{(k)}\|_F \leq p\, r_{\max}^{(k)} \quad \text{with} \quad r_{\max}^{(k)} = \max_{1\leq i\leq p}|\mathbf{R}_i^{(k)}|\,, \tag{3.16}$$

the maximum residual norm yields a convenient convergence criterion, $r_{\max}^{(k)} \leq \tau$, where τ is the convergence threshold.

An alternative convergence measure for Krylov subspace algorithms monitors the change in the corresponding Lagrangian functional F. The latter is a quadratic form whose stationary points are the solutions of the linear problem.[75] For eigenvalue problems, the Lagrangian takes the form

$$F[\mathbf{X}, \boldsymbol{\Omega}] = \left\langle \mathbf{X}^\dagger \mathbf{A}\mathbf{X} - \boldsymbol{\Omega}\left(\mathbf{X}^\dagger\mathbf{X} - \mathbf{1}\right)\right\rangle\,, \tag{3.17}$$

where $\langle\cdot\rangle$ denotes the trace operation. The minimum of $F[\mathbf{X}, \boldsymbol{\Omega}]$ over all $n \times p$ matrices is given by the matrix containing the lowest p eigenvectors as columns. The Lagrangian functional for (shifted) linear equations is given by

$$F[\mathbf{X}] = \left\langle \mathbf{X}^\dagger \mathbf{A}\mathbf{X} - \boldsymbol{\omega}\left(\mathbf{X}^\dagger\mathbf{X} - \mathbf{1}\right) - \mathbf{X}^\dagger\mathbf{P} - \mathbf{P}^\dagger\mathbf{X}\right\rangle\,, \tag{3.18}$$

in which linear equation is obtained by settings the shifts to zero, $\boldsymbol{\omega} = \mathbf{0}$. If all shifts are smaller than lowest eigenvalue of $\mathbf{A}$, the quadratic form is convex, and the stationary point is a minimum.

In the case of a restart after k iterations, the Krylov subspace basis is replaced by the set of the p solution iterates $\mathbf{X}^{(k)}$. In (shifted) linear problems, the columns of $\mathbf{X}^{(k)}$ must be first orthonormalized before proceeding.

3.3 Libkrylov Design and Implementation

The principal objective of libkrylov is serving as a flexible computational framework, which supports a wide variety of matrix-free linear problems including eigenvalue problems, linear equations, and shifted linear equations via a simple, uniform API. The user should be empowered to evaluate different preconditioning and orthonormalization techniques and experiment with hybrid approaches as demanded by the application domain. Composability is thus a central consideration. Since scientific software development is split between Fortran and C/C++ ecosystems, libkrylov aims to be interoperable with both language families and to provide for simple cross-platform build and installation procedures.

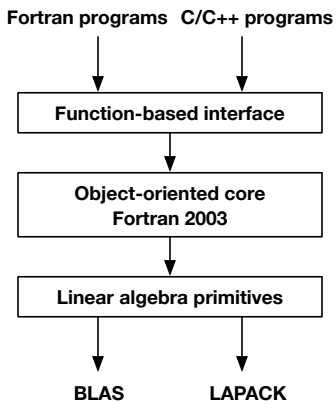

Figure 3.2: Libkrylov architecture. Invocations are indicated by arrows, composition by inclusion.

To make good on these promises, libkrylov uses a layered architecture, in which an object-oriented core is wrapped in a function-based interface layer, which allows calling from both Fortran and C/C++ (Fig. 3.2). The library is implemented in portable Fortran 2003 and takes advantage of the runtime polymorphism capabilities afforded by this standard. However, in the design of libkrylov, composition is preferred over inheritance as it induces looser coupling between components and allows for better composability. The simplified class diagram of libkrylov is shown in Fig. 3.3. The central component of libkrylov is the `space_t` abstract class, which represents a general Krylov subspace and provides an interface to the

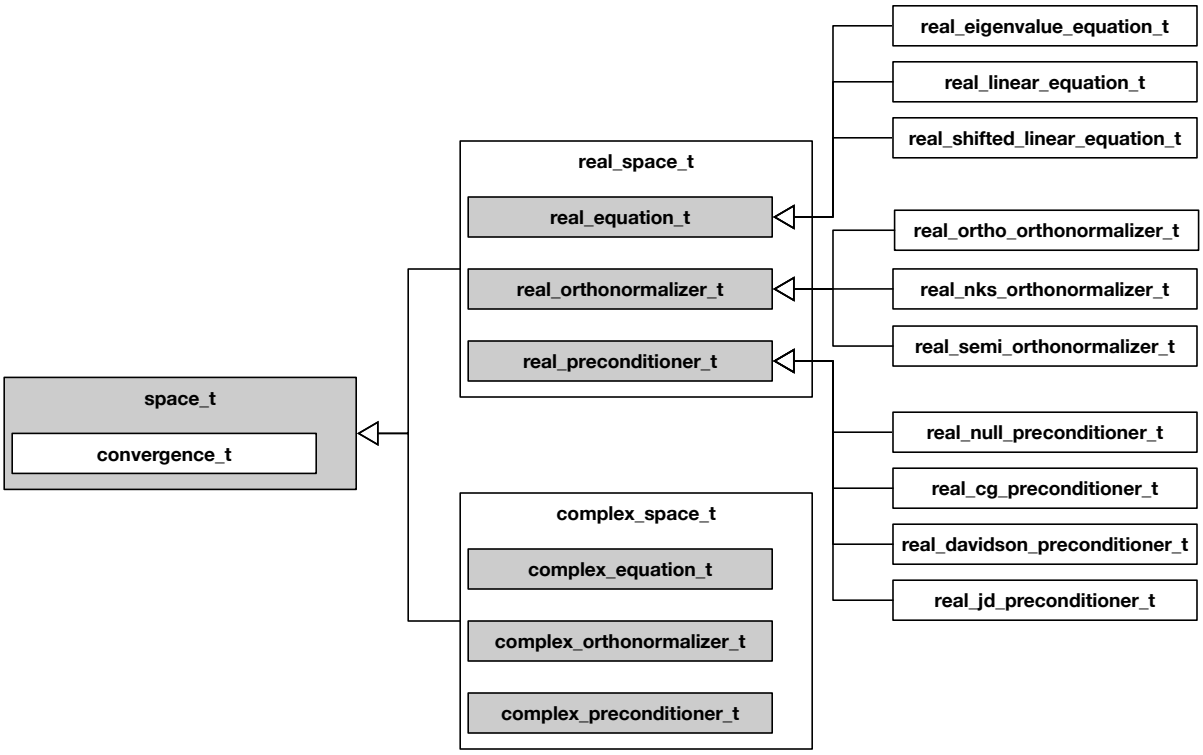

Figure 3.3: Simplified class diagram of libkrylov components. Abstract classes are shown in gray, concrete classes in white. Inheritance relationships are indicated by arrows.

steps of the iterative algorithm of Fig. 3.1. The implementations of the Krylov subspace algorithm for the real symmetric and complex Hermitian cases largely follow analogous paths. It would thus be desirable to use generic programming techniques to abstract over the underlying arithmetic. However, due to the lack of support for generic programming in Fortran 2003, one has to emulate generic classes by polymorphic types such as `real_space_t` and `complex_space_t` inheriting from `space_t`. Such *ad hoc* polymorphism might seem an unappealing alternative because it requires a significant amount of code duplication. But polymorphic classes also offer several advantages as they make extensions to other cases easy, for example for structured problems. For simplicity, we focus on the function of the `real_space_t` class and its components in the following. However, the details apply analogously to the complex case. The generalization of the libkrylov code to different floating-point precisions (single, double, and potentially quadruple) is implemented using code preprocessing, which is not part of the Fortran 2003 standard but is nevertheless almost universally

supported by Fortran compilers.

The internal structure of the `real_space_t` class is designed to maximize the separation of concerns between the equation type, orthonormalization, preconditioning, and convergence control. Therefore, the `real_space_t` class contains polymorphic components for the equation type, orthonormalizer, and preconditioner, see Fig. 3.3. The current implementation supports eigenvalue problems, linear equations, and shifted linear equations. The orthonormal Krylov subspace method, the nKs method, and the semiorthonormal variant are implemented via a common orthonormalizer interface. The preconditioning choices include the null preconditioner (no preconditioning, analogous to Arnoldi iteration), diagonal preconditioning (CG preconditioner), Davidson and Jacobi–Davidson (JD) preconditioning techniques.

This design of the `real_space_t` class allows the user to freely mix and match the components of the Krylov subspace algorithm when constructing a new Krylov subspace object. Flexible convergence control and transparent error handling enable experimentation to determine the best setup depending on the properties of the specific problem. The convergence control object tracks the norm of the residual matrix, the condition number of the overlap matrix (for a nonorthonormal basis), and the Lagrangian of the linear problem over the course of the iterative process. Different combinations of convergence and restart criteria can thus be specified by configuration, while sensible default settings are provided by the library.

The requirement of interoperability with both Fortran and C/C++ code bases significantly constrains the structure of the libkrylov interface. A simple function-based API was chosen for user interaction from both Fortran and C/C++. All `space_t` objects are created as global module variables inside the libkrylov library and addressed using numerical indices. Data exchange between user code and library code is done incrementally with separate function calls, for example, for starting vectors, RHS, or diagonal shifts. This approach has the advantage that new equation types can be added without breaking backwards compatibility.

Examples of libkrylov calls from Fortran and C are shown in Figs. 3.4 and 3.5.

```
program use_krylov
    integer :: err, ind
    real :: v(4, 1), s(4, 1), res
    err = krylov_initialize()
    ind =  krylov_add_space('r', 's', 'e', 4, 1, 1)
    v = reshape([1.0, 0.0, 0.0, 0.0], [4, 1])
    err = krylov_set_real_space_vectors(ind, 4, 1, v)
    err = krylov_solve_real_equation(ind, multiply)
    if (err /= 0) stop 1
    err =  krylov_get_real_space_solutions(ind, 4, 1, s)
    res = krylov_get_space_last_residual_norm(ind)
    err = krylov_finalize()
contains
    function multiply(n, m, v, p)
        integer :: n, m
        real :: v(n, m), p(n, m), m(4, 4)
        mat = reshape([5.0, 4.0, 1.0, 1.0, 4.0, 5.0, 1.0, 1.0, &
                       1.0, 1.0, 4.0, 2.0, 1.0, 1.0, 2.0, 4.0], [4, 4])
        p = matmul(mat, v)
        multiply = 0
    end function multiply
end program use_krylov
```

Figure 3.4: Example of calling libkrylov from Fortran. Program variables: `n`: dimension of coefficient matrix, `m`: current subspace dimension, `t`: number of solutions, `v(n, m)`: basis vectors, `p(n, m)`: matrix–vector products, `s(n, t)`: solutions, `res`: residual norm, `ind`: space index, `err`: error code. An explicit coefficient matrix `mat` is used for simplicity. In real implementations, the matrix–vector products are formed "on the fly".

The user interaction with libkrylov consists of only a few function calls. At minimum, the library must be initialized by the `(c)krylov_initialize` function, the problem type and the relevant dimensions must be specified in the `(c)krylov_add_space` function call, and the initial subspace basis must be passed to the `(c)krylov_set_real_space_vectors` function. The type of arithmetic (real or complex), equation type, and matrix structure are specified by character constants, in analogy to BLAS. In our example, `'r'` indicates real arithmetic, `'e'` means eigenvalue problem, and `'s'` stands for a symmetric coefficient matrix. The bulk of the computation takes place within the `(c)krylov_solve_real_equation` call. The matrix–vector product function `multiply` is passed as the first argument. For the sake of simplicity, the evaluation of matrix–vector products is simulated in Figs. 3.4 and 3.5 by multiplication with an explicit coefficient matrix. In real-life applications, the corresponding matrix–vector

```
#include "ckrylov.h"
int main() {
    long err, ind;
    double v[4] = {1.0, 0.0, 0.0, 0.0}, s[4], res;
    err = ckrylov_initialize();
    ind = ckrylov_add_space("r", 1, "s", 1, "e", 1, 4, 1, 1);
    err = ckrylov_set_real_space_vectors(ind, 4, 1, v);
    err = ckrylov_solve_real_equation(ind, multiply);
    if (err != CKRYLOV_OK) exit(1);
    err = ckrylov_get_real_space_solutions(ind, 1, s);
    res = ckrylov_get_space_last_residual_norm(ind);
    err = ckrylov_finalize();
    exit(0);
}
int multiply(const long *n, const long *m, const double *v, double *p) {
    double mat[] = {5.0, 4.0, 1.0, 1.0, 4.0, 5.0, 1.0, 1.0,
                    1.0, 1.0, 4.0, 2.0, 1.0, 1.0, 2.0, 4.0};
    for (long i = 0; i < *n * *m; ++i) p[i] = 0.0
    for (long i = 0; i < *m; ++i) {
        for (long j = 0; i < *n; ++j) {
            for (long k = 0; k < *n; ++k) {
                p[k + *m * i] += mat[k + *n * j] * v[j + *n * i];
            }
        }
    }
    return CKRYLOV_OK;
}
```

Figure 3.5: Example of calling libkrylov from C. See Fig. 3.4 for variable definitions. Integer constant `CKRYLOV_OK` indicates successful exit.

product function is provided by the user and may be arbitrarily complex as long as it adheres to the interface. The convergence or any error conditions of the iterative algorithm are communicated to the user by the return value of the `(c)krylov_solve_real_equation` function call, with 0 indicating success. In that case, the solution vectors may be retrieved by the `(c)krylov_get_real_space_solutions`. The convergence may be verified by checking the residual norm of the last iteration (`(c)krylov_get_space_last_residual_norm`). The converged eigenvalues may be obtained by additional function calls, which are omitted for brevity. After the interaction with libkrylov is completed, the allocated memory is freed by the (`(c)krylov_finalize`) function call. The complete libkrylov API documentation is provided with the libkrylov code.[111]

The current implementation keeps all program data in memory, which allows for the fastest data access. However, this strategy becomes unfeasible for very large problems or in memory-constrained environments. For these cases, a flexible data storage backend is planned for future releases, which enables both in-memory and out-of-core algorithms with adaptive blocking.

For ease of installation, distribution, and testing, libkrylov uses the CMake cross-platform build system[124]. The library is released under the open-source 3-clause BSD license.

3.4 Numerical Tests

The computational cost and the speed of convergence of Krylov subspace algorithms may strongly depend on the numerical properties of the coefficient matrix $\mathbf{A}$ and the details of the iterative procedure, in particular, on the choice of the preconditioning and orthonormalization methods. The evaluation of matrix–vector products in each iteration is the most computationally expensive step, scaling as $\mathcal{O}(n^2)$ for general dense matrices, as long as the number of iterations $K \ll n$. A large literature is devoted to specialized matrix–vector multiplication "engines", which can achieve significant computational cost reductions per iteration by implementing domain-specific intermediate representations, prescreening of matrix elements of $\mathcal{A}$, and other structure-dependent techniques.[75,83–92,110] In this section, we give selected examples of usage of libkrylov in calculations of molecular property calculations by time-dependent Hartree–Fock (TDHF)[11,125] and time-dependent density functional theory (TDDFT) methods.[126–130] For illustration purposes, we limit ourselves to calculations of size $n = 10^3..10^4$, in which the properties of the coefficient matrix $\mathbf{A}$ can be computed in advance and correlated with the performance characteristics of Krylov subspace algorithms. We focus on the convergence behavior of the iterative algorithm as a function of the preconditioning and orthonormalization techniques. We consider the structural characteristics

contributing to the efficiency of the Davidson preconditioner and related methods. The efficiency of iterative algorithms for solving the linear and eigenvalue equations of TDHF and TDDFT is compared in Ref.[15].

Within the Tamm–Dancoff approximation (TDA)[131–133] to TDDFT and the configuration interaction singles (CIS) method,[134] which may be considered the equivalent approximation to TDHF, electronic excitation energies and intensities are obtained from the solutions of a real symmetric eigenvalue equation,

$$\mathbf{A}\mathbf{X}_i = \Omega_i \mathbf{X}_i \,, \tag{3.19}$$

where the coefficient matrix $\mathbf{A}$ is positive definite. In most cases, only the $i = 1, ..., p$ lowest eigenvalues are of interest. Static polarizabilities in TDHF and TDDFT are computed by solving the real symmetric linear equation

$$(\mathbf{A} + \mathbf{B})(\mathbf{X} + \mathbf{Y})_\alpha = (\mathbf{P} + \mathbf{Q})_\alpha \,, \tag{3.20}$$

for the Cartesian components $\alpha = x, y, z$. The RHS vectors $(\mathbf{P} + \mathbf{Q})_\alpha$ contains matrix elements of the dipole moment operator.

Table 3.2 summarizes the structural characteristics of the coefficient matrices and the speed of convergence of eigenvalue problems for selected medium-size inorganic and organic molecules with TDDFT using the PBE exchange–correlation functional[135] and the resolution-of-the-identity (RI-J) approximation for the Coulomb interaction.[87,136] The basis sets used in the calculations were of split-valence quality (def2-SVP) and triple-zeta valence quality (def2-TZVP),[137] including diffuse augmentation (def2-SVPD).[138] $p = 1, 2$, and 10 lowest eigenvalues were determined simultaneously. The residual norm convergence criterion was $\tau = 10^{-7}$. The coefficient matrices and RHS vectors were generated by a modified version of the TURBOMOLE[139] escf program.

For each coefficient matrix $\mathbf{A}$, selected numerical characteristics relevant to the convergence and numerical stability of Krylov subspace algorithms are shown. Specifically, Table 3.2 includes the inverse 2-norm condition number κ_2^{-1},

$$\kappa_2 = \|\mathbf{A}\|_2 \|\mathbf{A}^{-1}\|_2 \,, \tag{3.21}$$

which serves as a general measure of numerical stability in operations involving $\mathbf{A}$. The ratio ρ_p of the highest requested eigenvalue and the next-higher eigenvalue controls the convergence of iterative algorithms for the p lowest eigenvalues in the absence of preconditioning,[79,140]

$$\rho_p = \Omega_{p+1}/\Omega_p \,. \tag{3.22}$$

To illustrate the performance of preconditioners based on the diagonal approximation to $\mathbf{A}$, Table 3.2 shows the average eigenvalue shift of the p lowest eigenvalues $\overline{\delta\Omega}_p$ of $\mathbf{A}$ (in a.u.), which characterizes the the closeness of the spectra of $\mathbf{A}$ and its diagonal approximation $\mathbf{D}$,

$$\overline{\delta\Omega}_p = \frac{1}{p} \sum_{i=1}^{p} \min_{j=1}^{n} |\Omega_i - A_{jj}| \,. \tag{3.23}$$

The average of the p lowest eigenvalues $\overline{\Omega}_p$ (in a.u.) is included in for comparison. Lastly, Table 3.2 lists iteration counts K of the orthonormal Krylov subspace algorithm until convergence (within maximum residual norm threshold $\tau = 10^{-7}$) using $q_0 = p$ starting vectors. Results are given for the CG preconditioner, Davidson (D) preconditioner, and Jacobi–Davidson preconditioner, variants 1 (JD1) and 2 (JD2). Without preconditioning, the Krylov subspace algorithm does not convergence within 50 iterations and is thus of little use for the eigenvalue problems considered here.

As Table 3.2 shows, the coefficient matrices are well-conditioned with $\kappa_2^{-1} = 10^{-4}..10^{-2}$, even with relatively flexible and diffuse basis sets. The spectral radius of $\mathbf{A}$ increases slowly

with basis set size, as excitations at the high end of the spectrum are progressively added due to the greater flexibility of the basis set. At the same time, the low end of the spectrum is relatively dense, as shown by the eigenvalue ratios ρ_p close to 1. In some cases, for example, S_8 ($p = 10$) and C_{20} ($p = 2$), the p-th lowest eigenvalue is part of a generate set due to point group symmetry, which results in the eigenvalue ratio ρ_p being exactly 1. The small eigenvalue gap explains the slow convergence in the absence of preconditioning. However, the preconditioning techniques based on the matrix diagonal lead to fast convergence in all cases. Notably, the Davidson preconditioner and the JD1 and JD2 variants of the Jacobi–Davidson preconditioner show nearly identical iteration counts and are superior to the CG preconditioning. Fig. 3.6 illustrates the convergence of the maximum residual norms in trans-thioindigo for $p = 1, 2$, and 10 lowest excitations. As expected from the small eigenvalue gap, the unpreconditioned Krylov subspace algorithm, which is equivalent to Arnoldi iteration, converges very slowly, managing to only reduce the maximum residual norm by only about one order of magnitude after 50 iterations. Diagonal (CG) preconditioning achieves convergence in 34 iterations for the lowest eigenvalue, 27 iterations for the $p = 2$ lowest eigenvalues, and 23 iterations for $p = 10$ lowest eigenvalues. With the Davidson and Jacobi–Davidson preconditioners, the algorithm converges in 23 iterations for the lowest eigenvalue, 19 iterations for the $p = 2$ lowest eigenvalues, and 14 iterations for the $p = 10$ lowest eigenvalue. Alternatively, the convergence of the Krylov subspace algorithm can be assessed using the corresponding Lagrangian functional F, as shown in Fig. 3.7. Once a sufficiently large subspace is constructed, the Lagrangian converges very rapidly towards its stationary value F_0.

Fig. 3.8 compares the convergence of the orthonormal Krylov subspace algorithm with the nKs and the semiorthonormal methods for the lowest excitations in trans-thioindigo. As discussed in detail in Ref. 75, the nonorthonormal algorithms do not degrade the speed of convergence but are helpful in reducing the computational effort per iteration, if the matrix–vector product function can exploit the reduction in vector norms during the iterative

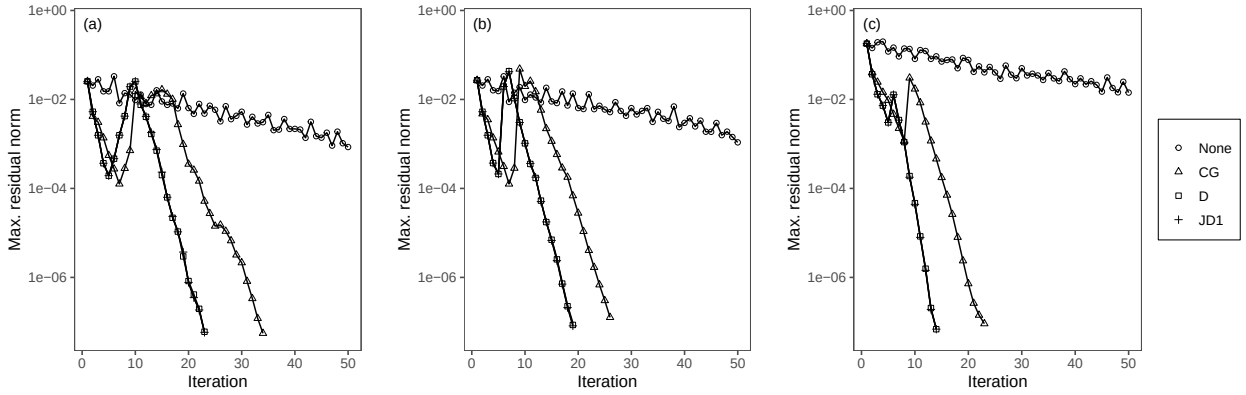

Figure 3.6: Convergence of the maximum residual norms of the orthonormal Krylov subspace algorithm for calculations of the p lowest electronic excitations of trans-thioindigo without preconditioning (None) and with diagonal (conjugate gradient, CG), Davidson (D), and Jacobi–Davidson preconditioner, variant 1 (JD1). (a) $p = 1$, (b) $p = 2$, (c) $p = 10$. $q_0 = p$ basis vectors are used as starting subspace basis.

process. In our tests, the semiorthonormal variant has not shown an improvement over the nKs method.

The dependence of the speed of convergence on the dimension of the starting subspace basis is illustrated in Fig. 3.9 for calculations in trans-thioindigo using the Davidson preconditioner. Larger starting subspace produce faster convergence due to greater overlap with the desired eigenvectors. With smaller starting subspaces, monotonic convergence is often achieved only after one or more "jumps", during which the maximum residual norm is increasing. However, the final subspace dimension q_K at convergence is not strongly dependent on q_0: In the example of Fig. 3.9, the lowest eigenvalue is converged for $q_K = 23, 24, 21, 25$, and 31 given starting subspace dimensions of $q_0 = 1, 2, 4, 8, 16$, respectively. Similarly, in the case of $p = 2$ lowest eigenvalues, the final subspace dimension is $q_K = 35, 32, 37, 39$ for $q_0 = 2, 4, 8, 16$, respectively. For $p = 10$ lowest eigenvalues, the final subspace dimension is $q_K = 107, 104, 109$ for $q_0 = 10, 12$, and 16, respectively. The optimal choice of the starting subspace dimension depends on the specifics of the matrix–vector product computation. If the computational effort for simultaneously computing q matrix–vector products is proportional to q, one should seek to minimize the final subspace dimension q_K. In that case, $q_0 = p + 2..4$ appears to

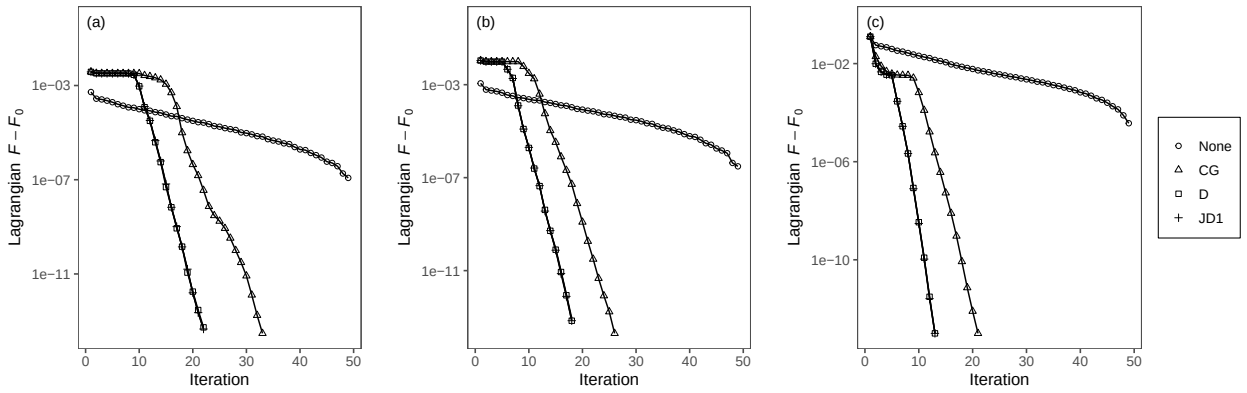

Figure 3.7: Convergence of the Lagrangian functional F for the p lowest electronic excitations of trans-thioindigo without preconditioning (None) and with diagonal (conjugate gradient, CG), Davidson (D), and Jacobi–Davidson preconditioner, variant 1 (JD1). (a) $p = 1$, (b) $p = 2$, (c) $p = 10$. F_0 is the stationary value. $q_0 = p$ basis vectors are used as starting subspace basis.

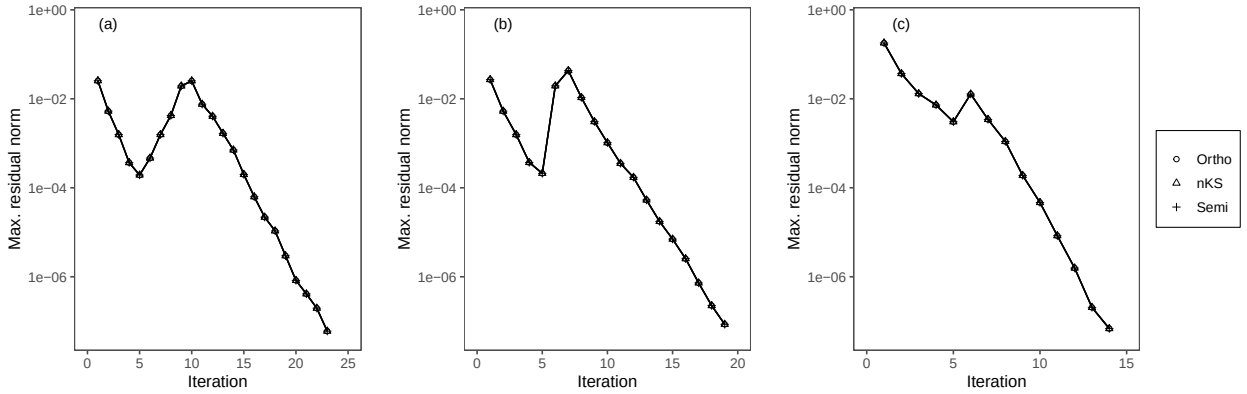

Figure 3.8: Convergence of the maximum residual norms of the Krylov subspace algorithm for the p lowest electronic excitations in trans-thioindigo with Davidson preconditioner using orthonormal algorithm (Ortho), nonorthonormal Krylov subspace method (nKs), and semiorthonormal method (Semi). (a) $p = 1$, (b) $p = 2$, (c) $p = 10$. $q_0 = p$ basis vectors are used as starting subspace basis.

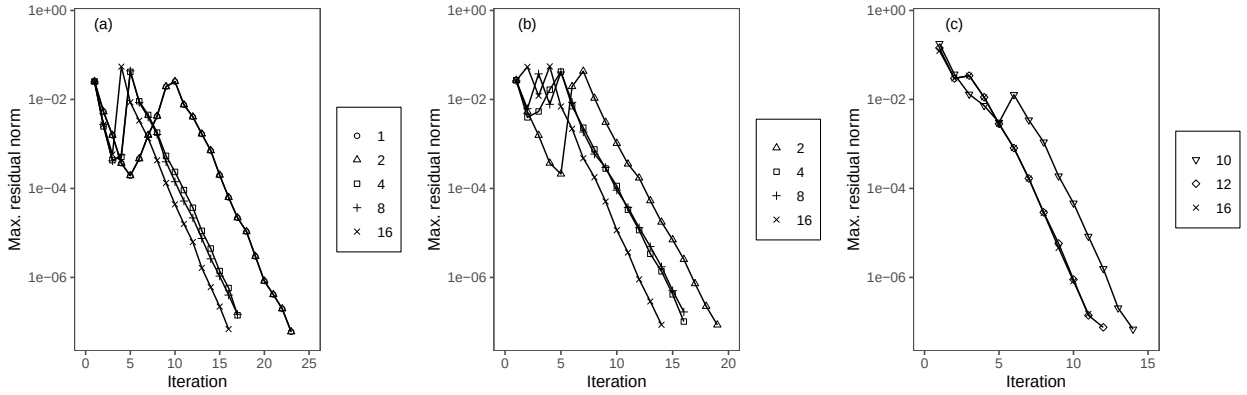

Figure 3.9: Convergence of the maximum residual norms of the orthonormal Krylov subspace algorithm for the p lowest electronic excitations in trans-thioindigo with different starting subspace basis dimensions q_0 using Davidson preconditioner. (a) $p = 1$, $q_0 = 1, 2, 4, 8, 16$, (b) $p = 2$, $q_0 = 2, 4, 8, 16$, (c) $p = 10$, $q_0 = 10, 12, 16$.

give the best results for the systems considered in this work. If the computational effort is approximately independent of q, as is the case in integral-direct TDDFT and TDHF implementations,[127,129,130] one prefers to minimize the number of iterations K and should choose a somewhat larger starting subspace. In this work, the choice of $q_0 = p + 6..8$ yields the smallest number of iterations. However, we should stress that the choice of the initial subspace may be strongly system-dependent, and a preliminary study is advisable for picking a suitable initial dimension. Testing different choices for the initial dimension can also help detect potential "missing" eigenvectors in the presence of symmetries.

The good performance of the Davidson method and related algorithm is frequently linked to the diagonal dominance of the coefficient matrix $\mathbf{A}$.[96,116] The matrix $\mathbf{A}$ is called diagonally dominant if the following condition is satisfied:[118,141]

$$|A_{ii}| \geq r'_i \quad \text{for all } i = 1..n, \quad \text{where} \quad r'_i = \sum_{j \neq i} |A_{ij}| \,. \tag{3.24}$$

The Gershgorin circle theorem relates the eigenvalues of $\mathbf{A}$ to the magnitude of the off-

diagonal elements,

$$|\Omega_i - A_{jj}| \leq r'_j \quad \text{for some} \quad j = 1..n\,, \tag{3.25}$$

that is each eigenvalue of $\mathbf{A}$ lies within a circle in the complex plane with the center at A_{jj} and the radius r'_j for some j. Crucially, the radius of the Gershgorin circle is determined by the off-diagonal elements. The eigenvalues of diagonally dominant matrices must thus be close to some diagonal elements. The converse statement, does not need to hold, however, as we will see shortly.

If we apply the definition of diagonal dominance to the smallest diagonal element of the matrix $\mathbf{A}$ for trans-thioindigo, we observe that this matrix is far from being diagonally dominant in the sense of Eq. (3.24). For example, the smallest diagonal value of the matrix $\mathbf{A}$, is approximately 0.086 a.u., while the corresponding Gershgorin circle sweeps the sizeable interval $[-0.880, 1.052]$, indicating large off-diagonal elements by absolute value. While the Gershgorin disks contain the eigenvalues of $\mathbf{A}$, the corresponding bounds are too loose to be useful. Similar observations can be made for the other coefficient matrices considered here. However, Table 3.2 shows that despite not being diagonally dominant, the coefficient matrices nevertheless show only relatively small deviations $\overline{\delta\Omega}_p$ between the eigenvalues of the coefficient matrix $\mathbf{A}$ and its diagonal values at the low end of the spectrum. With the exception of C_{20}, the eigenvalue shift $\overline{\delta\Omega}_p$ is 20% or less of the average eigenvalue $\overline{\Omega}_p$. The small eigenvalue shift explains the superior performance of the Davidson method since the preconditioning matrix $\mathbf{K}_{\mathrm{D}}^{(k)}$ is near-singular and enhances the relevant eigenvector in a way similar to inverse iteration.[118]

The performance of Krylov subspace algorithms depends significantly on the properties of the coefficient matrix $\mathbf{A}$. While the numerical tests discussed here are illustrative of the problems encountered in molecular property calculations, in other cases the above conclusions need

not hold. In this case, the user may well want to benchmark the preconditioners on their specific problem. The modular structure of libkrylov makes this benchmarking straightforward. We note that in our numerical tests, the Jacobi–Davidson method does not offer an improvement over the Davidson preconditioner. A similar observation has been previously reported for nearly diagonal matrices.[142] However, the Jacobi–Davidson preconditioner can give better performance for matrices that are not nearly diagonal, especially if a more accurate approximation to **A** than its diagonal **D** is chosen as the preconditioning matrix. For these matrices, the comparison with the LOBPCG method[105,106] is also instructive.

Compound	Basis	n	p	Coefficient matrix **A**				Iteration count K			
				κ_2^{-1}	ρ_p	$\overline{\delta\Omega}_p$	$\overline{\Omega}_p$	CG	D	JD1	JD2
Na_8	SVP	3344	10	$1.13 \cdot 10^{-3}$	1.060	0.012	0.053	13	9	9	9
	TZVP	9328	10	$9.85 \cdot 10^{-4}$	1.040	0.012	0.051	15	11	11	11
$(H_2O)_6$	SVP	3420	10	$1.01 \cdot 10^{-2}$	1.005	0.002	0.261	22	10	10	10
	SVPD	6120	10	$9.32 \cdot 10^{-3}$	1.009	0.002	0.243	20	8	8	8
	TZVP	6840	10	$3.52 \cdot 10^{-3}$	1.018	0.001	0.254	21	8	8	8
S_8	SVP	5120	10	$1.41 \cdot 10^{-3}$	1.000	0.006	0.133	38	19	19	19
	SVPD	9728	10	$1.37 \cdot 10^{-3}$	1.000	0.006	0.129	38	18	18	18
	TZVP	14848	10	$1.16 \cdot 10^{-3}$	1.000	0.006	0.131	41	21	21	21
$B_{10}C_2H_{12}$	SVP	7067	10	$1.97 \cdot 10^{-2}$	1.009	0.006	0.265	34	16	15	15
	SVPD	11063	10	$1.95 \cdot 10^{-2}$	1.005	0.004	0.263	29	14	14	14
	TZVP	15059	10	$7.68 \cdot 10^{-3}$	1.008	0.004	0.263	29	13	13	13
Ag_6	SVP	7353	10	$1.58 \cdot 10^{-3}$	1.024	0.012	0.116	34	14	14	14
	TZVP	10431	10	$7.89 \cdot 10^{-4}$	1.024	0.012	0.115	33	14	17	18
C_{20}	SVP	13200	1	$2.83 \cdot 10^{-3}$	1.027	0.009	0.037	15	12	12	12
			2		1.000	0.010	0.037	11	11	11	11
			10		1.052	0.021	0.076	36	23	24	24

Compound	Basis	n	p	Coefficient matrix $\mathbf{A}$				Iteration count K			
				κ_2^{-1}	ρ_p	$\overline{\delta\Omega}_p$	$\overline{\Omega}_p$	CG	D	JD1	JD2
Coumarin	SVP	5548	10	$5.86 \cdot 10^{-3}$	1.033	0.022	0.187	22	14	14	14
	SVPD	8968	10	$6.02 \cdot 10^{-3}$	1.005	0.019	0.185	23	16	16	16
	TZVP	12882	10	$2.12 \cdot 10^{-3}$	1.043	0.022	0.185	21	13	14	14
DMABN	SVP	6435	10	$8.65 \cdot 10^{-3}$	1.003	0.012	0.206	24	13	13	13
	SVPD	10179	10	$5.41 \cdot 10^{-3}$	1.042	0.011	0.162	25	14	14	14
	TZVP	14118	10	$3.14 \cdot 10^{-3}$	1.020	0.012	0.199	25	14	14	14
YP	SVP	7095	10	$5.67 \cdot 10^{-3}$	1.021	0.015	0.179	19	11	11	11
	SVPD	11610	10	$5.79 \cdot 10^{-3}$	1.010	0.013	0.175	21	12	12	12
DPA	SVP	8640	10	$8.15 \cdot 10^{-3}$	1.004	0.027	0.183	22	12	12	12
Anthracene	SVP	9353	10	$8.63 \cdot 10^{-3}$	1.009	0.023	0.172	32	16	16	16
	SVPD	14711	10	$8.46 \cdot 10^{-3}$	1.004	0.018	0.169	31	16	15	15
Luciferin	SVP	16416	10	$1.32 \cdot 10^{-3}$	1.031	0.013	0.148	22	12	12	12
cis-Thioindigo	SVP	19152	10	$7.27 \cdot 10^{-4}$	1.002	0.011	0.111	25	15	15	15
trans-Thioindigo	SVP	19152	1	$8.99 \cdot 10^{-4}$	1.040	0.004	0.082	34	23	23	23
			2		1.011	0.005	0.084	27	19	19	19
			10		1.024	0.013	0.112	23	14	14	14
Crystal violet	SVP	44200	10	$5.31 \cdot 10^{-3}$	1.045	0.023	0.123	22	15	14	14

Table 3.2: Characteristics of the coefficient matrix $\mathbf{A}$ (of size n, in a.u.) and iteration counts K of orthonormal Krylov subspace algorithms for computing the p lowest electronic excitations using TDDFT in the TDA approximation with diagonal (conjugate gradient, CG), Davidson (D), and Jacobi–Davidson preconditioners, variants 1 (JD1) and 2 (JD2). κ_2^{-1} is the inverse 2-norm condition number of $\mathbf{A}$, $\overline{\delta\Omega}_p$ is the average eigenvalue shift, and ρ_p is the $(p+1)/p$ eigenvalue ratio. The number of starting vectors was $q_0 = p$, convergence threshold was $\tau = 10^{-7}$. See text for details.

3.5 Conclusions

Libkrylov strives to unify both battle-tested and more recent algorithmic developments in on-the-fly matrix computations within an extensible framework that enables simple integration with user codes and easy experimentation. The scientific software developer does not need to reinvent the wheel and can focus on the domain-specific aspects of implementing an efficient matrix–vector product function. The parameters of the Krylov subspace algorithm, such as orthonormalization and preconditioning methods, can be then optimized for the specific application. As an open-source library, the continuing development of libkrylov is driven by the requirements of its users. Due to its modular structure and a flexible API, implementations of Krylov subspace algorithms for different equation types, intermediate representations, and preconditioners can be incorporated in an efficient and backwards-compatible manner. More algorithms and implementations from the rich landscape of Krylov subspace methods[76–81,100,101] may be contributed to libkrylov in the future.

Several directions for further developments are currently envisioned. Structured problems of the symplectic type[75] arise in the computation of response properties and excitation energies with TDHF[11,125] and TDDFT methods.[126–130] The extension to linear problems involving real unsymmetric and complex non-Hermitian coefficient matrices is necessary for implementations of coupled cluster (CC) response theory.[143,144] An unsymmetric variant of the Davidson preconditioner[145] can be used in these cases. Finally, additional equation types, for example, Sylvester equation and least-squares problems allow for expansions into other problem domains.

Chapter 4

Extending Libkrylov to Structured Problems for Molecular Response Calculations

This chapter is based on a manuscript under preparation. This work was supported by the Molecular Sciences Software Institute (MolSSI) under NSF grant CHE-2136142.

4.1 Introduction

Time-dependent density functional theory (TDDFT) plays a crucial role in the calculation of excitation energies, providing a powerful and versatile approach for studying electronic excited states. TDDFT has become an invaluable tool in various fields of research, including chemistry, physics, biology and materials science, due to its accuracy and computational efficiency.

One of the primary applications of TDDFT is in the prediction and interpretation of optical

absorption spectra. By solving the time-dependent Schrödinger equation within the framework of DFT, TDDFT allows for the calculation of excitation energies and corresponding transition probabilities. This enables researchers to understand and interpret experimental absorption spectra, providing insights into the electronic structure and properties of materials.

TDDFT also allows for the investigation of excited-state properties such as oscillator strengths, transition dipole moments, and excited-state geometries. These properties are crucial for understanding phenomena such as fluorescence, phosphorescence, and photochemistry. By accurately calculating these quantities, TDDFT provides a theoretical basis for interpreting and predicting experimental observations, facilitating the design and optimization of materials with desired electronic and optical properties.

The solutions to the TDDFT equations involve solving large eigenvalue and linear problems. For larger problems, direct solutions are computationally prohibitive, and the coefficient matrices may even exceed available storage capacity. To address this challenge, matrix-free iterative subspace algorithms have become a standard practice. These methods only require computation of matrix-vector products, eliminating the need for explicit matrix calculations and storage. The origins of these methods can be traced back to Roos'[83] integral direct configuration interaction methods in 1983 and have gained popularity following Davidson's work in the 1970s[94].

The efficiency of these matrix-free methods relies on exploiting the Hermiticity of the problem. However, the TDDFT eigenvalue problem is non-Hermitian, which poses challenges. The Davidson method, which has been extended to non-Hermitian problems[146], has not been as successful in the context of TDDFT. Non-Hermitian problems can yield complex eigenvalues, which can impede convergence in iterative subspace algorithms due to the difficulty of generating new trial vectors. The work by Olsen and coworkers[14] has shown considerable success by addressing the convergence challenges.

Over the years, substantial advancements have been made both in the development of the formal theory and algorithms for TDDFT.[11,126,129,147] These algorithmic improvements have led to a level of accuracy that is often satisfactory for practical applications in larger systems that are otherwise prohibitive for traditional correlated ab initio methods.

Despite this progress, TDDFT is plagued by reference state instabilities within the adiabatic approximation (AA).[147–149] Even in more advance theories such as multi-configurational self-consistent field (MCSCF) or coupled cluster theories[150–153], instabilities remain ever present. In multireference wavefunction methods such as complete active space self-consistent field (CASSCF) instabilities can be avoided by including the corresponding excitations in the active space[154–156], but remain intractable for many applications due to their factorial scaling.

The accurate computation of response properties in time-dependent approximate many-electron theories has been limited by widespread instabilities. In this study, we present the application and effectiveness of structure-preserving Krylov subspace algorithms for calculating molecular response properties within the framework of TDDFT.

Our study represents the first known attempt at implementing an algorithm that preserves the full SO(1, 1) symmetry of the problem. In the absence of magnetic fields, this symmetry reduces to the use of split-complex numbers. By adopting a structure-preserving approach, we aim to address the limitations and instabilities observed in existing methods for calculating response properties.

Preserving the structure of the linear response function is crucial for achieving efficient and numerically stable solutions in iterative algorithms for large-scale response problems.[14] In the context of Krylov subspace methods, this preservation is achieved by selecting symmetry-adapted basis vectors $\mathbf{V}^{(k)}$ that maintain the symplectic orthonormality condition.

4.2 Time-Dependent Density Functional Theory

We provide a brief overview of the fundamental equations used in this context, focusing on the necessary information to comprehend our approach to the problem, recognizing that more comprehensive treatments can be found in the existing literature[157–160]. We adopt the usual notation; indices $i, j, \ldots$ label occupied, $a, b, \ldots$ virtual, and $p, q, \ldots$ general molecular orbitals (MOs). Molecular response properties within the TDDFT formalism are solutions to the linear response function

$$(\mathbf{\Lambda} - \omega\mathbf{\Delta})|\mathbf{X}, \mathbf{Y}\rangle = -|\mathbf{P}, \mathbf{Q}\rangle, \tag{4.1}$$

where the 2×2 superoperators $\mathbf{\Lambda}$ and $\mathbf{\Lambda}$ within the adiabatic approximation (AA) are

$$\mathbf{\Lambda} = \begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B} & \mathbf{A} \end{pmatrix}, \quad \mathbf{\Delta} = \begin{pmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & -\mathbf{1} \end{pmatrix}. \tag{4.2}$$

$\mathbf{X}$ and $\mathbf{Y}$ are conveniently collected as a vector on a Hilbert space

$$|\mathbf{X}, \mathbf{Y}\rangle = \begin{pmatrix} \mathbf{X} \\ \\ \mathbf{Y} \end{pmatrix} \tag{4.3}$$

of occupied and virtual MOs. The MOs are solutions of the static Kohn-Sham (KS) equations with eigenvalues $\epsilon_{p\sigma}$. $\mathbf{P}$ and $\mathbf{Q}$ contain the coupling between the perturbing field and the molecular system.

$\mathbf{A}$ and $\mathbf{B}$, often called orbital rotation Hessians are gathered as

$$\begin{aligned}(\mathbf{A} + \mathbf{B})_{ia\sigma jb\sigma'} = (\epsilon_{a\sigma} - \epsilon_{i\sigma})\delta_{ij}\delta_{ab}\delta_{\sigma\sigma'} + 2(ia\sigma|jb\sigma') + 2f^{\mathrm{xc}}_{ia\sigma jb\sigma'} \\ - c_{\mathrm{x}}\delta_{\sigma\sigma'}[(ja\sigma|ib\sigma) + (ab\sigma|ij\sigma)],\end{aligned} \tag{4.4a}$$

$$(\mathbf{A}-\mathbf{B})_{ia\sigma jb\sigma'} = (\epsilon_{a\sigma}-\epsilon_{i\sigma})\delta_{ij}\delta_{ab}\delta_{\sigma\sigma'} + c_{\mathrm{x}}\delta_{\sigma\sigma'}[(ja\sigma|ib\sigma)+(ab\sigma|ij\sigma)] \tag{4.4b}$$

$f^{\mathrm{xc}}_{ia\sigma jb\sigma'}$ is the matrix element of the exchange-correlation kernel and $(pq\sigma|rs\sigma')$ is the two electron repulsion term. The right-hand side (RHS) $|\mathbf{P},\mathbf{Q}\rangle$ couples the molecular system to the external perturbing field. The structure of eq 4.1 also emerges in time-dependent Hatree-Fock (TDHF) and advanced theories.

4.2.1 Conventional Algorithms for Solving the Response Equations

Excitation energies are determined as the poles of the linear response function and are the solutions of the symplectic eigenvalue problem

$$(\mathbf{\Lambda}-\omega_n\mathbf{\Delta})|\mathbf{X}_n,\mathbf{Y}_n\rangle = 0, \tag{4.5}$$

The eigenvalue ω_n is the transition frequency between the ground state $|0\rangle$ and the $n-th$ excited state $|n\rangle$ with corresponding eigenvector $|\mathbf{X}_n,\mathbf{Y}_n\rangle$ subject to the orthonormality constraints

$$\langle\mathbf{X}_n,\mathbf{Y}_n|\mathbf{\Delta}|\mathbf{X}_n,\mathbf{Y}_n\rangle = 1, \tag{4.6}$$

Rearranging eq 4.5 gives

$$(\mathbf{\Lambda}+\omega_n\mathbf{\Delta})|\mathbf{Y}_n,\mathbf{X}_n\rangle = 0, \tag{4.7}$$

suggesting a paired structure where $-\omega_n$ is also an eigenvalue with corresponding eigenvector $|\mathbf{Y}_n, \mathbf{X}_n\rangle$ normalized as

$$\langle \mathbf{Y}_n, \mathbf{X}_n | \boldsymbol{\Delta} | \mathbf{Y}_n, \mathbf{X}_n \rangle = -1. \tag{4.8}$$

Equations 4.5 and 4.7 are written in a more compact blocked matrix form as

$$\begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B} & \mathbf{A} \end{pmatrix} \begin{pmatrix} \mathbf{X} & \mathbf{Y} \\ \mathbf{Y} & \mathbf{X} \end{pmatrix} - \begin{pmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{-1} \end{pmatrix} \begin{pmatrix} \mathbf{X} & \mathbf{Y} \\ \mathbf{Y} & \mathbf{X} \end{pmatrix} \begin{pmatrix} \boldsymbol{\Omega} & \mathbf{0} \\ \mathbf{0} & -\boldsymbol{\Omega} \end{pmatrix} = \mathbf{0}, \tag{4.9}$$

and the orthonormality conditions eq 4.6 and 4.8 as

$$\begin{pmatrix} \mathbf{X} & \mathbf{Y} \\ \mathbf{Y} & \mathbf{X} \end{pmatrix}^T \begin{pmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{-1} \end{pmatrix} \begin{pmatrix} \mathbf{X} & \mathbf{Y} \\ \mathbf{Y} & \mathbf{X} \end{pmatrix} = \begin{pmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{-1} \end{pmatrix}. \tag{4.10}$$

Olsen and coworkers introduced an extension of Davidson's method to address symplectic problem.[14] In their approach, they preserve the symplectic structure throughout the iterative process by including paired trial vectors in the subspace. Similarly, Kauczor *et al.* also employed a method that involves using symmetric and antisymmetric trial vectors to preserve the problem's structure.[15] This approach yields results similar to the approach proposed by Olsen *et al.*. The two methods have not garnered much attention because of their computational cost.

Following a suggestion by Casida in 1990, the response eigenproblem can be reformulated into a block-diagonal form using a unitary transformation.[86,126,129,161],

$$\left[\begin{pmatrix} \mathbf{A}+\mathbf{B} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}-\mathbf{B} \end{pmatrix} - \omega \begin{pmatrix} \mathbf{0} & \mathbf{1} \\ \mathbf{1} & \mathbf{0} \end{pmatrix} \right] \begin{pmatrix} \mathbf{X}+\mathbf{Y} \\ \mathbf{X}-\mathbf{Y} \end{pmatrix} = \mathbf{0}. \tag{4.11}$$

This transformation allows decoupling of the response equations, simplifying the problem and facilitating the solution to a non-Hermitian problem

$$(\mathbf{A}-\mathbf{B})(\mathbf{A}+\mathbf{B})(\mathbf{X}+\mathbf{Y}) = \Omega^2(\mathbf{X}+\mathbf{Y}). \tag{4.12}$$

If (**A** + **B**) and (**A** - **B**) are positive definite, a further transformation of eq 4.12 results in a Hermitian problem,

$$(\mathbf{M}-\lambda)\mathbf{T} = 0 \tag{4.13}$$

with the eigenpair λ, **T**, where

$$\mathbf{M} = (\mathbf{A}-\mathbf{B})^{1/2}(\mathbf{A}+\mathbf{B})(\mathbf{A}-\mathbf{B})^{1/2} \tag{4.14}$$

and the solutions to the original problem are recovered as

$$\begin{aligned} \Omega &= \sqrt{\lambda}, \\ (\mathbf{X}+\mathbf{Y}) &= \frac{1}{\sqrt{\Omega}}(\mathbf{A}-\mathbf{B})^{1/2}\mathbf{T}, \\ (\mathbf{X}-\mathbf{Y}) &= \sqrt{\Omega}(\mathbf{A}-\mathbf{B})^{-1/2}\mathbf{T}. \end{aligned} \tag{4.15}$$

The above approach requires both (**A** + **B**) and (**A** - **B**) to be positive definite, restricting the excitation spectrum to only positive reals. In the spin-restricted closed-shell formalism, non-positive spectrum of (**A** + **B**) correspond to singlet and triplet instabilities, whereas that of (**A** - **B**) indicate non-real instabilities.[147,148]

4.3 Solving the Response Equations Using Split-complex Numbers

4.3.1 The Split-Complex Numbers

The split-complex or hyperbolic number[162], $z \in \mathbb{S}$ is an ordered pair of two real numbers x and y in the standard basis $\{1, j\}$

$$z = x + jy \tag{4.16}$$

where $j^2 = 1$ and the split-complex conjugate of z is $z^* = x - jy$. In this form, the split-complex numbers $\mathbb{S} \equiv \mathbb{R}[j]$ is an extension of the real numbers to include the unipotent/split-imaginary j in the same way the complex numbers $\mathbb{C} \equiv \mathbb{R}[i]$ extend real numbers by including the imaginary i. The arithmetic operations in split-complex numbers are similar to those in complex numbers, with the addition and multiplication defined as follows:

$$(x_1 + jy_1) + (x_2 + jy_2) = (x_1 + x_2) + j(y_1 + y_2),$$
$$(x_1 + jy_1)(x_2 + jy_2) = (x_1x_2 + y_1y_2) + j(x_1y_2 + y_1x_2).$$

The conjugate product of the split-complex number

$$\eta(z) = z^*z = x^2 - y^2 \tag{4.17}$$

does not induce a norm like in $\mathbb{C}$.

More generally the conjugate product of two split-complex numbers z_1 and z_2 is given by

$$z_1^*z_2 = (x_1 - jy_1)(x_2 + jy_2) = (x_1x_2 - y_1y_2) + j(x_1y_2 - y_1x_2). \tag{4.18}$$

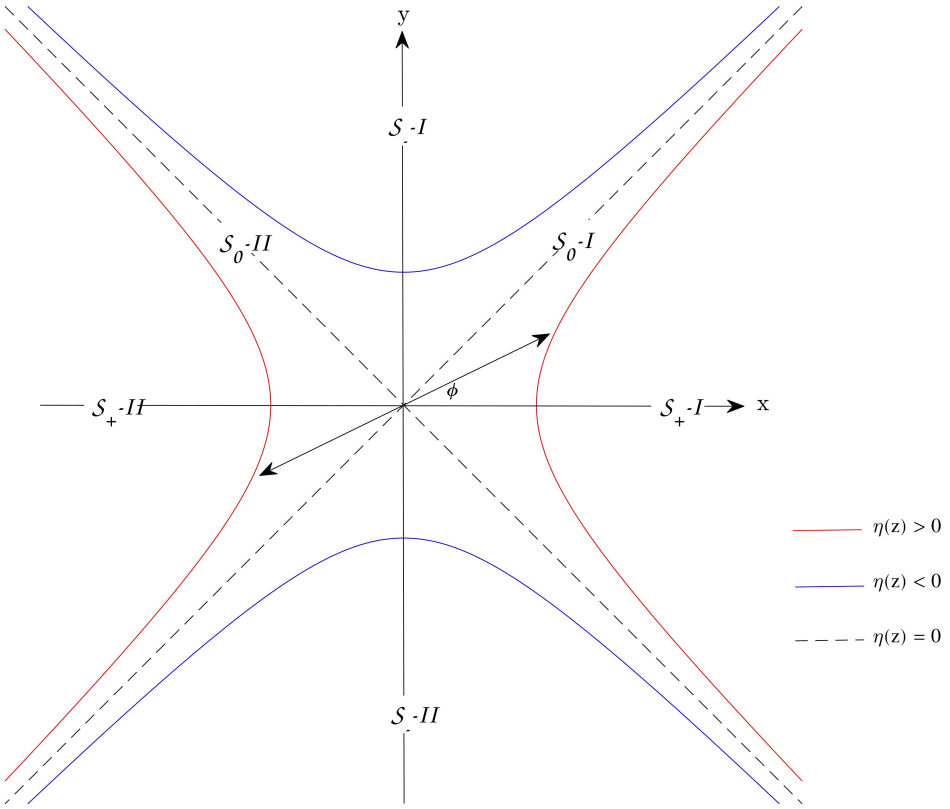

Figure 4.1: The split-complex hyperbolic plane.

The set of all split-complex numbers is a four-branched hyperbola of radius r, Fig 4.1, satisfying

$$z = \pm r(cosh\phi + jsinh\phi) = \pm re^{j\phi} \tag{4.19}$$

when z lies in one of the $\mathcal{S}_+$ quadrants, or

$$z = \pm r(sinh\phi + jcosh\phi) = \pm rje^{j\phi} \tag{4.20}$$

when z lies in one of the $\mathcal{S}_-$ quadrants. $\phi \in \mathbf{R}$ is the hyperbolic angle of rotation. Split-complex numbers that lie on the asymptotic lines $\mathcal{S}_0$ are said to be neutral or isotropic.

The split-complex numbers algebra is isomorphic to the algebra of 2×2 real symmetric

matrices just as complex numbers are isomorphic to 2×2 real skew-symmetric matrices.

$$z = x + jy \longmapsto \begin{pmatrix} x & y \\ y & x \end{pmatrix} \tag{4.21}$$

Alternatively, the split-complex numbers can be represented in null or idempotent basis $\{j_+, j_-\}$ as

$$z = z_+ j_+ + z_- j_- \tag{4.22}$$

where the basis $\{j_+, j_-\}$ is defined as

$$j_+ = \frac{1}{2}(1 + j), \quad j_- = \frac{1}{2}(1 - j)$$

and the components z_+ and z_- are

$$z_+ = x + y, \quad z_- = x - y$$

for $x, y \in \mathbb{R}$. From the above definitions; $j_+^2 = j_+$ and $j_-^2 = j_-$, hence the nomenclature idempotent basis. Likewise the basis elements are said to be null or mutually annihilating, i.e., $j_+ j_- = 0$. The conjugate of z is $z^* = z_- j_+ + z_+ j_-$.

Special properties of this basis makes it suitable for calculation. Multiplication of two complex numbers is component-wise in this basis

$$wz = (w_+ j_+ + w_- j_-)(z_+ j_+ + z_- j_-) = w_+ z_+ j_+ + w_- z_- j_-.$$

and

$$\eta(w, z) = \eta(w)\eta(z).$$

A more interesting result is that z is isomorphic with real diagonal matrices in the null basis

$$z = z_+ j_+ + z_- j_- \longmapsto \begin{pmatrix} x+y & 0 \\ 0 & x-y \end{pmatrix}. \tag{4.23}$$

From the isomorphism of eqs 4.23 it is apparent that the decomposition $z = z_+ j_+ + z_- j_-$ is the spectral decomposition of $z = x + jy$, eqs 4.21.

4.3.2 The 2D Problem

We consider a 2×2 real symmetric eigenvalue problem ($A, B, x, y, \omega \in \mathbb{R}$),

$$\left[\begin{pmatrix} A & B \\ B & A \end{pmatrix} - \omega \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \right] \begin{pmatrix} x \\ y \end{pmatrix} = 0 \tag{4.24}$$

subject to the normalization

$$\begin{pmatrix} x \\ y \end{pmatrix}^T \begin{pmatrix} x \\ -y \end{pmatrix} = x^2 - y^2 = 1. \tag{4.25}$$

This $2d$ real problem corresponds to a $1d$ split-complex problem

$$Cz = \xi z^* \tag{4.26}$$

by mapping $B \to jB$ and $y \to jy$ in the split-complex plane, where $C = A + jB$ and $z = x + jy$ are split-complex by definition. z satisfies the correct normalization constraints, $z^*z = 1$.

C has the polar decomposition according to eq 4.19 and 4.20 where $r = \sqrt{|A^2 - B^2|}$ and $\phi = \tanh^{-1}(B/A)$ in the quadrants defined by the $\mathcal{S}_+$ or $\phi = \tanh^{-1}(A/B)$ in $\mathcal{S}_-$, respectively.

The solutions to eq 4.26 are uniquely determined by which quadrant C lies in the split-complex plane; $\xi = \pm r$ when C lies in one of the $\mathcal{S}_+$ hyperbolic surfaces or $C^*C > 0$, whereas $\xi = \pm jr$ for $C^*C < 0$. That is to say jz is also a solution to eq 4.26 with $j\xi$,

$$C(jz) = (j\xi)(jz)^*. \tag{4.27}$$

In this sense, it is always possible to scale the z by j such that corresponding the ξ is either purely real or purely split-imaginary.

4.3.3 Extending to Higher Dimension

We extend the 2×2 problem 4.24 to higher dimensions. For $n \times n$ matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{X}$, and $\mathbf{Y}$. We assume for the purposes of our derivations that these matrices are real-valued and $\mathbf{A}$ and $\mathbf{B}$ be real symmetric matrices. Defining $\mathbf{C} = \mathbf{A} + j\mathbf{B}$ and $\mathbf{Z} = \mathbf{X} + j\mathbf{Y}$, the $n \times n$ split-complex problem

$$\mathbf{CZ} = \mathbf{Z}^*\mathbf{\Xi}, \tag{4.28}$$

corresponds to the symplectic eigenvalue problem 4.9 where $\mathbf{\Xi}$ is $n \times n$ diagonal matrix containing the split-complex eigenvalues of $\mathbf{C}$. The solution to eq 4.28 is the eigenpair $\{\mathbf{\Xi},\ \mathbf{Z}\}$ which corresponds to the solution of 4.5, $\{\mathbf{\Omega},\ |\mathbf{X}, \mathbf{Y}\rangle\}$. Scaling $\{\mathbf{\Xi},\ \mathbf{Z}\}$ by j results in the solution $\{j\mathbf{\Xi},\ j\mathbf{Z}\}$ which map to the solution of 4.7, $\{-\mathbf{\Omega},\ |\mathbf{Y}, \mathbf{X}\rangle\}$.

Orthogonality Condition

For pseudo-eigenvectors $\mathbf{z}_1$ and $\mathbf{z}_2$ of $\mathbf{C}$ with corresponding pseudo-eigenvalues ξ_1 and ξ_2. For $\eta(\xi_1) \neq \eta(\xi_2)$ and $\eta(\xi_1), \eta(\xi_2) \neq 0$ it can be shown that

$$\xi_1 \mathbf{z}_2^T \mathbf{z_1}^* = \xi_2 \mathbf{z}_1^T \mathbf{z}_2^* \tag{4.29}$$

and hence

$$(\eta(\xi_1) - \eta(\xi_2))(\mathbf{z}_1^T \mathbf{z}_2^*) = 0. \tag{4.30}$$

Since ξ_1 and ξ_2 are distinct, $\mathbf{z}_1$ and $\mathbf{z}_2$ must be orthogonal. Thus for all cases where ξ_1 and ξ_2 are (a) both purely real, (b) both pure split-imaginary and (c) one purely real and the other purely split-imaginary it is a necessary requirement that $\mathbf{z}_1^T \mathbf{z}_2^* = 0$.

$\mathbf{C}$ has the spectral decomposition

$$\mathbf{C} = \mathbf{Z}^* \Xi \mathbf{Z}^{*T}, \tag{4.31}$$

subject to the $\mathbb{K}$-orthonormality constraints

$$\mathbf{Z}^T \mathbf{Z}^* = \mathbf{1}. \tag{4.32}$$

We refer to eq 4.31 as the split-complex Takagi factorization because of its equivalence to the symplectic Takagi factorization[163] of the response eigenvalue equation 4.6 and 4.8,

$$\begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B} & \mathbf{A} \end{pmatrix} = \begin{pmatrix} \mathbf{X} & -\mathbf{Y} \\ -\mathbf{Y} & \mathbf{X} \end{pmatrix} \begin{pmatrix} \mathbf{\Omega} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Omega} \end{pmatrix} \begin{pmatrix} \mathbf{X} & -\mathbf{Y} \\ -\mathbf{Y} & \mathbf{X} \end{pmatrix}^T. \tag{4.33}$$

The $\mathbb{K}$-orthonormality constraints 4.32 is so called because it obeys the orthonormality con-

straints 4.10. Multiplication by $\mathbf{\Delta}$ takes the place of split-complex conjugation.

4.4 Implementation

4.4.1 Iterative Subspace Method in Split-Complex Type

In this section, we present our implementation of the proposed method. Our implementation aims to address the challenges and improve the efficiency of the computational procedures involved in the calculations of molecular response properties. Building upon the concepts and techniques discussed in the preceding section, we have developed a robust and efficient algorithm based on the Krylov subspace method that allows for accurate determination of the desired properties.

In this presentation, we adopt the blocked form of Krylov subspace algorithms and using matrix notation for clarity and conciseness. This notation allows us to express the algorithms in a unified manner and facilitates a better understanding of their underlying principles and computational procedures. For consistency * denotes split-complex conjugation and † denotes split-complex conjugate transposition. All matrices are of split-complex elements unless stated otherwise.

4.4.2 New Algorithm for Response Eigenvalue Problem

Using the generic type support of libkrylov in Ch. 3, we have extended the library to the split-complex type for the calculation of molecular response properties. Our implementation is presented as follows:

Step 1. To compute p lowest eigenpairs of eq 4.28, we select $\mathbb{K}$-orthonormal subspace basis

$\mathbf{V}^{(k)}$ of dimensions $n \times q_k$ ($q_k \geq p$) where $k = 1, 2, \ldots$ is the iteration number. For convenience the vectors are orthonormalized as

$$\mathbf{V}^{(k)T}\mathbf{V}^{(k)*} = \mathbf{1}. \tag{4.34}$$

by modified Gram-Schmidt (MGS) orthogonalization. The iteration becomes unstable if the $\mathbb{K}$-orthonormality of 4.34 is not enforced explicitly, see Section 4.4.5 below.

Step 2. Compute and save the matrix-vector products, $\mathbf{W}^{(k)}$

$$\mathbf{W}^{(k)} = \mathbf{A}\mathbf{V}^{(k)}, \tag{4.35}$$

Step 3. Compute the Rayleigh matrix

$$\mathbf{c}^{(k)} = \mathbf{V}^{(k)T}\mathbf{C}\mathbf{V}^{(k)}, \tag{4.36}$$

which is the projection of $\mathbf{C}$ onto $\mathcal{K}^{(k)}$.

Step 4. Solve for the eigenpair $\{\boldsymbol{\Xi}^{(k)}, \mathbf{z}^{(k)}\}$ of the projected problem

$$\mathbf{c}^{(k)}\mathbf{z}^{(k)} = \mathbf{z}^{(k)*}\boldsymbol{\Xi}^{(k)}, \tag{4.37}$$

according to Section 4.4.3 below. Because $q_k \ll n$, the computational cost involved in the Ritz-step (eq 4.37) can be kept as low as possible. The best approximation to the k-th eigenvectors is given by

$$\mathbf{Z}^{(k)} = \mathbf{V}^{(k)}\mathbf{z}^{(k)}, \tag{4.38}$$

Step 5. Convergence of the i-th eigenpair, $i = 1, \ldots, p$, is assessed by computing the residual

matrix $\mathbf{R}$

$$\mathbf{R}^{(k)} = \mathbf{C}\mathbf{Z}^{(k)} - \mathbf{Z}^{(k)*}\mathbf{\Xi}^{(k)}. \tag{4.39}$$

The residual vectors of i-th approximate eigenvectors $\mathbf{Z}^{(k)}$ form the columns of $\mathbf{R}^{(k)}$ and will vanish when convergence is reached. The i-th eigenpairs are converged for the residual vectors whose norms are below a pre-defined threshold.

Step 6. Expand the subspace with the columns of $\mathbf{R}^{(k)}$ whose pseudo-norm are above the convergence threshold, However, for faster converging Krylov subspace[76,81,120–123] a suitable chosen preconditioning matrix $\mathbf{\Pi}$ is applied to the residual

$$\tilde{\mathbf{R}}^{(k)} = \mathbf{\Pi}^{(k)}\mathbf{R}^{(k)}, \tag{4.40}$$

before adding to the subspace for the $(k+1)$th iteration.

$$\mathbf{V}^{(k+1)} = [\mathbf{V}^{(k)} \quad \tilde{\mathbf{R}}^{(k)}]. \tag{4.41}$$

4.4.3 Solution to the Split-complex Eigenvalue Problem

In the absence of readily available linear algebra libraries for the split-complex numbers we can resort to solving a real symmetric problem for 4.37 on the subspace.[86,161]

In the null basis, $\mathbf{c}$ has real components $(\mathbf{a}+\mathbf{b})$ and $(\mathbf{a}-\mathbf{b})$. To simplify the problem, we can transform 4.37 to an eigendecomposition form

$$\mathbf{g}\mathbf{z} = \mathbf{z}\mathbf{\Upsilon}, \tag{4.42}$$

where $\mathbf{g} = \mathbf{c}^{\dagger}\mathbf{c}$ and $\mathbf{\Upsilon} = \mathbf{\Xi}^{\dagger}\mathbf{\Xi}$. This eigendecomposition has the structure of a non-Hermitian

eigenvalue problem:

$$(\mathbf{a}-\mathbf{b})(\mathbf{a}+\mathbf{b})(\mathbf{x}+\mathbf{y}) = \boldsymbol{\Upsilon}(\mathbf{x}+\mathbf{y}). \tag{4.43}$$

We can further transform 4.43 to a Hermitian problem following the approach discussed in Sec. 4.2.1 where it assumed that $(\mathbf{a} \pm \mathbf{b})$ is positive definite. This allows us to utilize optimized LAPACK libraries, which are readily available for solving real symmetric eigenvalue problems. We can leverage the efficiency and reliability of these libraries to compute the eigenvalues and eigenvectors of the transformed problem, thereby obtaining the solutions to the split-complex pseudo-eigenvalue problem.

It is worth noting that it is possible to relax the constraints on $(\mathbf{a} \pm \mathbf{b})$, which is beyond the scope of this present work. Relaxing this constraint would require a proper definition for the square root operator of split-complex numbers. Relaxation of this constraints provides greater flexibility in solving the split-complex pseudo-eigenvalue problem and expands the range of problems that can be effectively addressed using our method.

4.4.4 Preconditioning in the Split-Complex Type

A preconditioner is a mathematical technique used in iterative methods for solving linear systems of equations or eigenvalue problems. Its purpose is to improve the convergence properties and efficiency of the iterative algorithms by transforming the original problem into an equivalent one that is easier to solve.[76]

The choice of a preconditioner depends on the specific problem at hand and the characteristics of the system matrix. A good preconditioner should approximate the behavior of the original matrix in a way that reduces its condition number and makes it easier to solve. Common preconditioning techniques include diagonal scaling, incomplete factorization, and

algebraic multigrid methods, among others.

In Krylov subspace methods the preconditioner is applied to the residual, resulting in a preconditioned residual that represents the preconditioned system. A simple and sometimes efficient way to improve convergence is the static diagonal preconditioner approach often employed in the conjugate-gradient (CG) algorithm.[76,77,81] Here we approximate the preconditioner as

$$\mathbf{\Pi}_{CG}^{(k)} = \mathbf{D}. \tag{4.44}$$

where $\mathbf{D} = \mathrm{diag}(C_{11}, \ldots, C_{nn})$.

In the context of eigenvalue problems, one of the most effective preconditioning techniques is the shift-and-invert technique.[79] This technique involves selecting a suitable shift value σ (usually chosen as the k-th eigenvalue) and using it to shift and invert the original matrix $\mathbf{M}$. This results in a new matrix $(\mathbf{M} - \sigma\mathbf{I})^{-1}$, with improved spectral separation leading to faster convergence.

In our approach to the Davidson (D) algorithm in split-complex numbers, we select the preconditioner, $\mathbf{\Pi}$ as

$$\mathbf{\Pi}_{D}^{(k)} = \mathbf{D} - \mathbf{\Xi}^{(k)}, \tag{4.45}$$

$\mathbf{\Xi}^{(k)}$ is the kth eigenvalue. For purely real $\mathbf{\Xi}^{(k)}$, $\mathbf{\Pi}^{(k)}$ decomposes to $2n \times 2n$ real symmetric matrix

$$\mathbf{\Pi}_{D}^{(k)} = \begin{pmatrix} \mathrm{diag}(\mathbf{A}) - \mathbf{\Xi}^{(k)} & \mathrm{diag}(\mathbf{B}) \\ \mathrm{diag}(\mathbf{B}) & \mathrm{diag}(\mathbf{A}) - \mathbf{\Xi}^{(k)} \end{pmatrix}, \tag{4.46}$$

recovering the preconditioner in the Olsen algorithm when $\mathrm{diag}(B) = 0$ and for purely split-

imaginary $\mathbf{\Xi}^{(k)}$

$$\mathbf{\Pi}_D^{(k)} = \begin{pmatrix} \mathrm{diag}(\mathbf{A}) & \mathrm{diag}(\mathbf{B}) - \mathbf{\Xi}^{(k)} \\ \mathrm{diag}(\mathbf{B}) - \mathbf{\Xi}^{(k)} & \mathrm{diag}(\mathbf{A}) \end{pmatrix}. \tag{4.47}$$

In the Olsen algorithm[14], the preconditioner takes the form $\begin{pmatrix} \mathrm{diag}(\mathbf{A}) - \Omega^{(k)} & \mathbf{0} \\ \mathbf{0} & \mathrm{diag}(\mathbf{A}) + \Omega^{(k)} \end{pmatrix}$.

To enforces additional orthogonality constraints, the Jacobi-Davidson (JD) preconditioner is used

$$\mathbf{\Pi}_{JD}^{(k)} = (\mathbf{1} - \mathbf{P}^{(i)})\mathbf{\Pi}_D^{(k)}(\mathbf{1} - \mathbf{P}^{(i)}), \tag{4.48}$$

ensuring that the preconditioned residuals are orthogonal to the solution.[116,117] $\mathbf{P}^{(i)}$ is the projector onto the set of the i-th approximate solutions.

4.4.5 Iterated Orthogonalization

In a Euclidean inner product space, when starting with a set of linearly independent vectors $\{v_1, \ldots, v_k\}$, the process of orthogonalization leads to a set of orthogonal vectors $\{u_1, \ldots, u_k\}$ that span the same subspace as $\{v_1, \ldots, v_k\}$. Each vector u_i is constructed to be orthogonal to all previously computed vectors $\{u_1, u_2, ..., u_{i-1}\}$. Moreover, these vectors can be normalized to unit vectors.

However, in the split-complex plane, this is not the case due to the presence of an indefinite inner product. When working with a set of linearly independent vectors $\{v_1, \ldots, v_k\}$ consisting of split-complex elements, orthogonalization results in an orthogonal set of vectors $\{u_1, \ldots, u_k\}$ that can span either the positive subspace ($\mathcal{S}_+$) or the negative subspace ($\mathcal{S}_-$), or both. Vectors spanning both $\mathcal{S}_+$ and $\mathcal{S}_-$ can introduce serious numerical instabilities

during an iterative process.

In our approach, we address this issue by employing modified Gram-Schmidt (MGS) orthogonalization for the split-complex numbers in an iterative manner. We achieve this by performing a 45° rotation (equivalent to scaling by the split-imaginary unit, j) of vectors located outside the desired subspace after an initial MGS orthogonalization. These vectors, while orthogonal to those within the desired subspace, are adjusted to ensure the generation of vectors that exclusively span either $\mathcal{S}_+$ or $\mathcal{S}_-$.

While it is theoretically possible to generate neutral vectors (i.e., $\eta(u_i) = 0$ where $u_i \neq 0$), we have implemented robust techniques to avoid such cases. However, if such a situation does arise, it could indicate nontrivial solution that requires further analysis and consideration. Alternatively, scaling neutral vectors by $\frac{1}{\sqrt{2}}(1 \pm j)$ in the null basis rotates them by 45° in the split-complex plane. This scaling operation allows us to handle these vectors appropriately and continue the iterative process within the Krylov subspace algorithm.

4.4.6 Error Bounds and Convergence of the Algorithm

The norm of the residual matrix $\mathbf{R}^{(k)}$ can be interpreted as an estimation of the error in the solution and therefore provide an upper bound to the approximate eigenvalues in the k-th iteration[118],

$$|\mathbf{\Xi}_i - \mathbf{\Xi}_i^{(k)}| \leq \sqrt{2}\|\mathbf{R}^{(k)}\|_{\mathbf{\Lambda}} \quad \text{for } 1 \leq i \leq q_k\,, \tag{4.49}$$

where the $\|.\|_{\mathbf{\Lambda}}$ refers to the norm, typically the Euclidean norm, induced by the inner-product over $\mathbf{\Lambda}$. It has also been argued that the Frobenius norm is a more stringent convergence criterion since $\|\mathbf{R}^{(k)}\|_F \geq \|\mathbf{R}^{(k)}\|_2$.[75] However both the Euclidean and Frobenius norms are somewhat artificial choices in regard to the TDDFT response equations. The

correct error bounds to the symplectic nature of the problem should then be

$$|\Xi_i - \Xi_i^{(k)}| \leq \sqrt{2}\|\mathbf{R}^{(k)}\|_s \quad \text{for } 1 \leq i \leq q_k \,, \tag{4.50}$$

where $\|\mathbf{R}^{(k)}\|_s$ is defined as

$$\|\mathbf{R}^{(k)}\|_s = \sqrt{|\eta(\mathbf{R}^{(k)})|}. \tag{4.51}$$

This approach is more consistent according to the Bauer-Fike theorem [79,164], since the associated approximate eigenvector is $\mathbb{K}$-orthonormal. This requirement is only valid under the strong assumption that the basis remains in the $\mathcal{S}_+$ subspace.

By considering the functional,

$$F_s[\mathbf{Z}, \mathbf{\Xi}] = \langle \mathbf{Z}^T\mathbf{C}\mathbf{Z} - \mathbf{\Xi}(\mathbf{Z}^T\mathbf{Z}^* - \mathbf{1})\rangle \tag{4.52}$$

where the brackets $\langle . \rangle$ denote trace operation, we gain an alternative perspective on measuring convergence. [75] The stationary points of F_s correspond to the solutions of the linear problem.

4.5 Numerical Tests

The efficiency of iterative algorithms are usually judged based on the number of iterations and the convergence behaviour is significantly affected by the initial guess or starting point of the iterative process. To ensure a fair comparison between our method and others, we use the same initial conditions or starting points as in the other methods. This allows for a more accurate assessment of their relative performance and convergence behavior. We compare our work to methods aimed at preserving the structure of the TDDFT response equation in

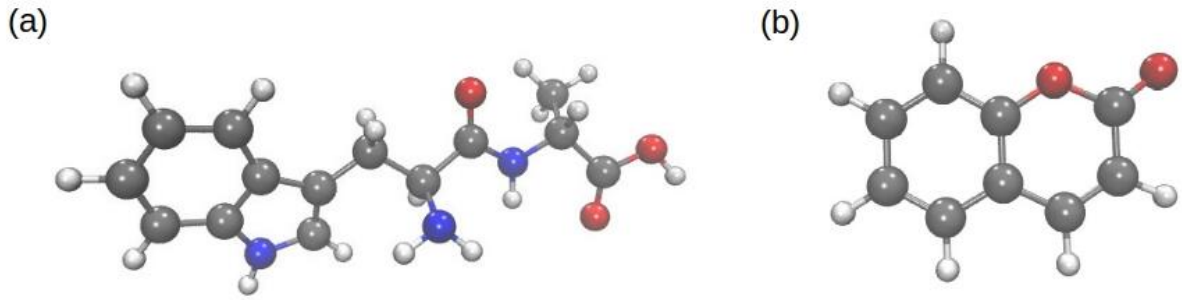

Figure 4.2: Molecular structures of: (a) Ala-Trp and (b) coumarin

4.5 as well as methods that implement the Hermitian problem of eq 4.12.

4.5.1 Convergence Properties of the New Algorithm

Table 4.1 presents the convergence behavior of our algorithm. It provides information about the convergence of our method used to solve eq 4.5, indicating the number of iterations and residual norms. This table allows for an assessment of the efficiency and effectiveness of the algorithm in finding the eigenvalues of interest. The calculation were carried out for the vertical excitation energies of Ala-Trp molecule at the CAM-B3LYP level of theory using the def2-SV(P) basis set (the equivalence of 6-31G basis set used in Olsen and Kauczor algorithms). Ground state structural optimization without symmetry constraints was carried out with geometry convergence thresholds of 10^{-4} a.u and energy convergence of 10^{-8} a.u. The coefficient matrix was generated by a modified version of the TURBOMOLE[139] escf program and $p = 1, 2$, and 10 lowest eigenvalues were determined simultaneously.

According to Table 4.1, the Olsen[14] and Kauczor[15] algorithms are essentially identical. This similarity arises from the fact that both methods utilize paired and symmetrized trial vectors that span the same subspace to preserve the structure of $\mathbf{\Lambda}$ and $\mathbf{\Delta}$. By preserving the structure of the problem throughout the iterative process, these algorithms find solutions to

	This work	TURBOMOLE	Kauczor alg.		Olsen alg.
	structured	unpaired	paired	symmetrized	symmetrized
Iteration	$\|\mathbf{R}\|_s$	$\|\mathbf{R}\|_2$	$\|\mathbf{R}\|_2$	$\|\mathbf{R}_g + \mathbf{R}_u\|_2$	$\|\mathbf{R}_g + \mathbf{R}_u\|_2$
1	0.06020(0.22474)	0.22375	0.11663	0.11663	0.11663
2	0.01274(0.06541)	0.05322	0.03819	0.03819	0.03819
3	0.01128(0.03428)	0.02718	0.02775	0.02775	0.02782
4	0.01352(0.00300)	0.01588	0.01723	0.01723	0.01725
5	0.00775(0.00124)	0.01066	0.02148	0.02148	0.02149
⋮	⋮	⋮	⋮	⋮	⋮
13	0.00069(0.00131)	0.00083	⋮	⋮	⋮
14	0.00033(0.00069)	0.00038	⋮	⋮	⋮
15	0.00017(0.00038)	0.00019	⋮	⋮	⋮
16	0.00009(0.00022)	0.00012	⋮	⋮	⋮
17	(0.00010)	0.00011	⋮	⋮	⋮
18	(0.00006)	0.00008	0.00146	0.00146	0.00144
19			0.00134	0.00134	0.00132
20			0.00068	0.00068	0.00068
21			0.00045	0.00045	0.00045
22			0.00030	0.00030	0.00030
23			0.00018	0.00018	0.00018
24			0.00008	0.00008	0.00008

Table 4.1: Residual Norms (to a threshold of 10^{-4}) for calculating the $p = 1$ lowest root of the TDDFT eigenvalue equation (See eq 4.5) using different iterative methods. $\|\mathbf{R}\|_s$ is the pseudo-norm (see eq 4.51), $\|\mathbf{R}\|_2$ refers to the Euclidean norm. $\mathbf{R}_g$ and $\mathbf{R}_u$ refer to the symmetric and antisymmetric residuals, respectively. The numbers in parenthesis indicate Euclidean residual norm.

a projected problem of size $2q_k \times 2q_k$.

In contrast, our method achieves convergence in nearly half the iterations required by the Olsen and Kauczor algorithms while still preserving the problem's structure. As explained in Section 4.3.3, our approach only needs to solve for half of the spectrum $\mathbf{\Lambda}$ in the split-complex type. The other half of the spectrum can be obtained by scaling with j, eliminating the need for solving a second equation or introducing additional paired vectors.

This efficiency in convergence is a significant advantage of our method. By reducing the number of iterations required, we streamline the computational process while maintaining the structure of the problem. This advancement allows for more efficient computations in the split-complex type while maintaining similar accuracy.

In the Olsen and Kauczor methods, the orthonormalization of vectors is performed using the Euclidean norm instead of the $\mathbf{\Delta}$ "metric" (we use metric here loosely since $\mathbf{\Delta}$ is strictly by definition not a metric). However, this choice does not yield a suitable update to the subspace in both algorithms. While it is often presented as an optimal choice for efficiency, we argue that this perspective stems from a misunderstanding of the correct algebraic approach to the problem.

While the use of the Euclidean norm may be seen as computationally efficient, it neglects the essential algebraic properties of the problem. By considering the correct algebraic approach, vis-à-vis the split-complex numbers which takes into account the $\mathbf{\Delta}$ "metric", for orthonormalization, we can find suitable update to the subspace, leading to faster convergence.

We agree that the computational cost for the matrix-vector product in our method is equivalent to that of the Olsen and Kauczor methods. All three algorithms require four matrix times vector multiplications for each new expansion vector, involving the computation of $\mathbf{AX}$, $\mathbf{AY}$, $\mathbf{BX}$, and $\mathbf{BY}$. However, the overall performance and efficiency of our algorithm is better compared to the Olsen and Kauczor algorithms.

In specialized methods that consider linear combinations of the $\mathbf{A}$ and $\mathbf{B}$ matrices, we only need to compute $(\mathbf{A}-\mathbf{B})(\mathbf{X}-\mathbf{Y})$ in the null basis. This advantage leads to a significant improvement, resulting in nearly twice the efficiency compared to the algorithm proposed by Olsen *et al.* This reduced computational cost is what makes the Hermitian approach more attractive than structure preserving approaches.[86,129]

Based on the results of Table 4.1, it is apparent that our work does not provide a significant improvement over the existing TURBOMOLE implementation. TURBOMOLE tackles the $n \times n$ Hermitian problem described in eq 4.12, enabling it to solve for the positive spectrum of $\mathbf{\Lambda}$ only. To obtain the paired spectrum, an expansion into a second subspace is necessary, effectively doubling the computational cost of the TURBOMOLE implementation. In contrast, our implementation is capable of solving for the paired spectrum without incurring this additional computational overhead. As a result, we achieve the desired outcome at a lower cost compared to Hermitian approach.

4.5.2 Effect of Preconditioning

Preconditioning plays a crucial role in our algorithm, significantly enhancing its convergence rate. In Section 4.4.4, we highlighted a key difference between our approach and the Olsen algorithm regarding the treatment of the $\mathbf{B}$ matrix in the diagonal approximation of $\mathbf{\Lambda}$.

The Olsen algorithm disregards the $\mathbf{B}$ matrix entirely in their diagonal approximation of $\mathbf{\Lambda}$, possibly due to convenience. The diagonal elements of the $\mathbf{A}$ matrix, contain orbital energy differences which are readily accessible, making it a straightforward choice for approximation. On the other hand, constructing $\mathrm{diag}(B)$ would be computationally expensive since it involves the two-electron integrals. However, by neglecting the $\mathbf{B}$ matrix in the diagonal approximation, the Olsen algorithm overlooks important information related to the split-complex nature of the problem. This choice represents a deviation from the correct

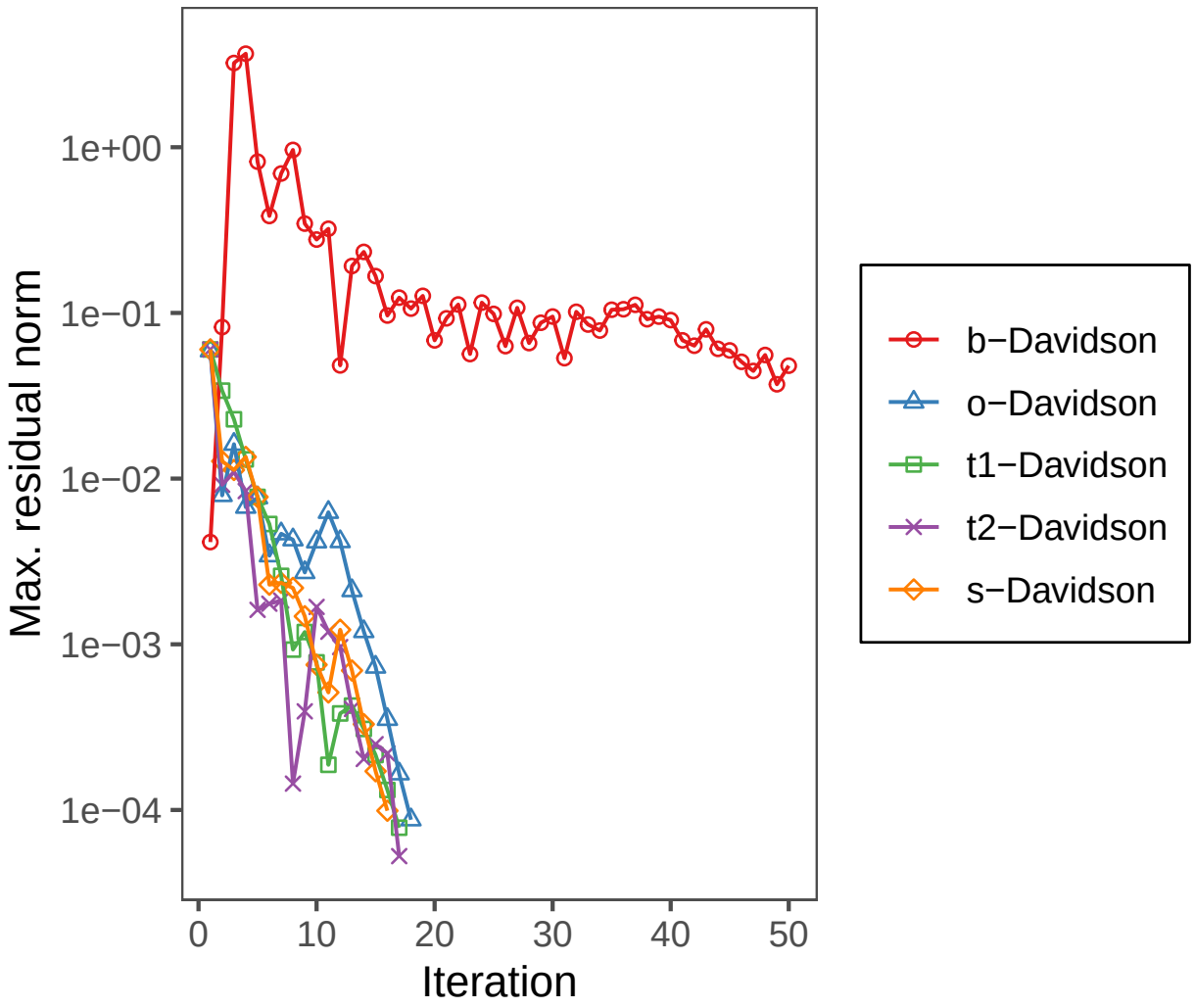

Figure 4.3: Convergence of the residual norm for the $p = 1$ lowest electronic excitation in Ala-Trp using different diagonal approximation for the Davidson preconditioner. b-Davidson uses diag(B), o-Davidson uses diag(A), t1-Davidson uses diag($A - B$), t2-Davidson uses diag($B - A$), b-Davidson uses diag(A) and diag(B) as in eq 4.45.

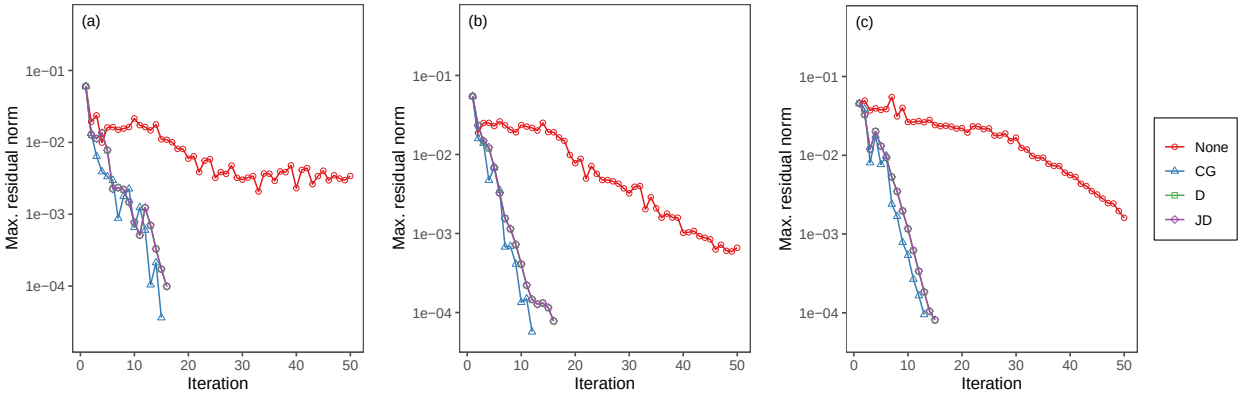

Figure 4.4: Convergence of the maximum residual norms for calculations of the p lowest electronic excitations of Ala-Trp without preconditioning (None) and with diagonal (conjugate gradient, CG), Davidson (D), and Jacobi–Davidson preconditioner, variant (JD). (a) $p = 1$, (b) $p = 2$, (c) $p = 10$. $q_0 = 2p$ basis vectors are used as starting subspace basis

algebraic approach to the problem.

In contrast, our algorithm incorporates the **B** matrix into the diagonal approximation. By doing so, we capture the complete information contained in both **A** and **B** matrices, leading to improved convergence properties. This choice aligns with the correct algebraic treatment of the split nature of the problem. It will be worthwhile to make use of the full diag(C), Fig. 4.3, if diag(B) can be computed at a minimal cost in large-scale applications.

After establishing that incorporating both the **A** and **B** matrices for preconditioning leads to faster convergence in the Davidson case, we proceeded to test the convergence of other widely used diagonal preconditioners as shown in Fig. 4.4. These preconditioners are commonly employed in iterative algorithms to improve convergence rates.

Figures 4.4 and 4.5 reveal an interesting finding: the diagonal (conjugate-gradient, CG) preconditioner achieves similar or even better convergence compared to the conventional Davidson and Jacobi-Davidson preconditioners. In the standard eigenvalue problem, it is typically observed that the Davidson and Jacobi-Davidson preconditioners outperform diagonal preconditioning.[165] This unexpected result, demonstrating the effectiveness of the conjugate-gradient (CG) preconditioner in the context of the split-complex pseudo-eigenvalue

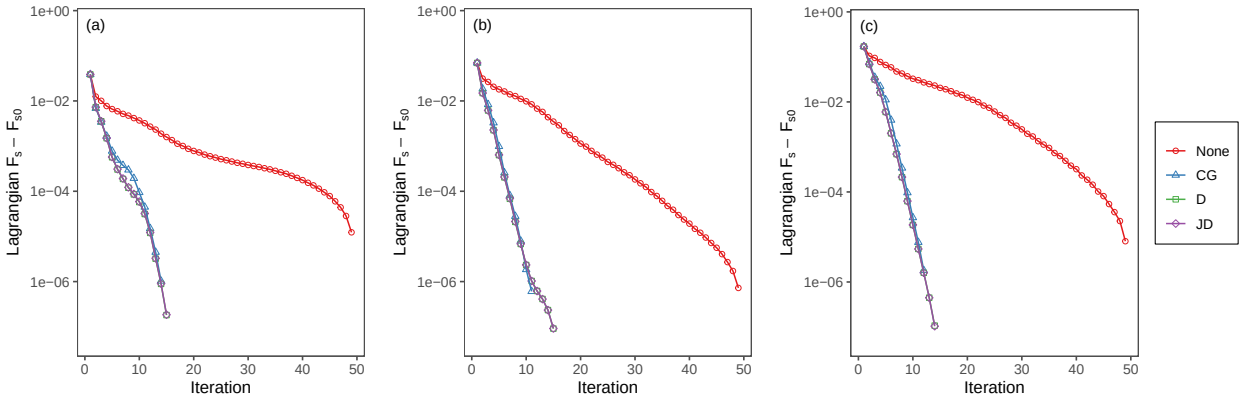

Figure 4.5: Convergence of the Lagrangian functional for calculations of the p lowest electronic excitations of Ala-Trp without preconditioning (None) and with diagonal (conjugate gradient, CG), Davidson (D), and Jacobi–Davidson preconditioning, variant (JD). (a) $p = 1$, (b) $p = 2$, (c) $p = 10$. $q_0 = 2p$ basis vectors are used as starting subspace basis

problem, raises the possibility that our understanding of the other preconditioners may be incorrect or incomplete.

The observed success of the CG preconditioner suggests that it is particularly well-suited for addressing the challenges and characteristics specific to the split-complex pseudo-eigenvalue problem. It may exploit certain properties or structures inherent to the problem that make it more effective compared to other preconditioning techniques.

It is possible that different preconditioners have varying levels of effectiveness depending on the specific problem at hand. The success of the CG preconditioner in this particular context highlights the importance of exploring and tailoring preconditioning techniques to suit the characteristics and requirements of the problem being addressed.

Further investigation and analysis may be necessary to better understand the reasons behind the unexpected results and to refine our understanding of the various preconditioners. This knowledge can then be applied to optimize and select appropriate preconditioning strategies for different problem domains, ensuring more accurate and efficient computations.

Symmetry	def2-SV(P)	def2-SVP	def2-TZVP	def2-SVPD	Expt.[166]
1 A′	4.34	4.33	4.27	4.30	4.00
1 A″	4.54	4.53	4.58	4.60	4.35
2 A′	4.87	4.86	4.77	4.78	4.52
3 A′	5.73	5.72	5.59	5.60	5.64
4 A′	6.12	6.11	6.00	5.99	

Table 4.2: Basis set comparison for the lowest lying singlet states (in eV) of Coumarin at the PBE0 level of theory.

4.5.3 Convergence at Different Levels of Theory

To assess the performance of our algorithm at various levels of theory, we selected coumarin as a test molecule. We conducted geometry optimizations of coumarin without symmetry constraints using different levels of theory and basis sets. Subsequently, we inferred the C_s point group symmetry by performing a re-optimization. Functional and basis set benchmarks have been extensively studied and reported in the literature. Therefore, in our analysis, we focused specifically on evaluating the robustness of our implementation rather than reiterating the existing benchmarks.

The basis sets used in this study include the def2-SV(P), SVP, TZVP, SVPD, and TZVPD. We compare the basis set convergence of our implementation at the PBE0 level of theory in Table 4.2. We have also repeated the calculation using the def2-SVP basis set for different theoretical methods in Table 4.3.

In our implementation, the final converged excitation energies of coumarin in Tables 4.2 and 4.3 typically agree to within 10^{-6} eV with the results computed using the TURBOMOLE program. The small discrepancies in the excitation energies are primarily attributed to numerical errors that arise during the generation of coefficient matrices and the use of slightly different physical constants in the two codes. The convergence achieved within the split-complex Krylov subspace algorithm and the preservation of the underlying structure of the

Symmetry	PBE0	B3LYP	CAM-B3LYP	TPSSh	TPSS	M06-2X	TDHF	Expt.
1 A′	4.33	4.23	4.53	4.15	3.76	4.58	5.01	4.00
1 A″	4.53	4.38	4.90	4.15	3.97	4.93	5.62	4.35
2 A′	4.86	4.74	5.11	4.66	4.47	5.20	6.08	4.52
3 A′	5.72	5.57	6.10	5.46	5.08	6.14	6.71	5.64
4 A′	6.11	5.94	6.39	5.69	5.20	6.47	7.18	
2 A″	6.24	6.05	6.70	5.78	5.46	6.68	7.89	

Table 4.3: Lowest lying singlet states (in eV) of Coumarin at various density functional levels of theory/TDHF using the def2-SVP basis set.

TDDFT response equations contribute to the accuracy of the calculated excitation energies.

4.6 Conclusion

We successfully implemented an efficient algorithm that computes excitation energies using Time-Dependent Density Functional Theory (TDDFT) within the adiabatic approximation. Our algorithm achieves a significant accuracy compared to similar approaches, at a fraction of the computational cost. These results highlight the efficiency and effectiveness of our approach in handling large-scale problems encountered in computational chemistry, providing valuable insights into electronic excited states and properties.

This algorithm exhibits better convergence properties compared to the approach presented by Olsen *et al.* and the work by Kauczor *et al.*, but it requires only half the computational cost. Moreover, we have demonstrated that our algorithm has similar convergence properties with the Hermitian approach of TURBOMOLE in terms of computational cost for the cases studied. These findings highlight the potential of our algorithm in efficiently and accurately solving the relevant equations in TDDFT/TDHF calculations.

Implementing an algorithm that can rigorously handle cases where either $(\mathbf{A}+\mathbf{B})$ or $(\mathbf{A}-\mathbf{B})$

is indefinite, as discussed in Sec. 4.4.3, would be a logical next step. Such a method would offer significant benefits in addressing instabilities in a physically meaningful way. Additionally, a more rigorous development of the Davidson preconditioner specifically tailored for the split-complex pseudo-eigenvalue problem would offer significant benefits. The implementation presented in this Chapter will be made available in Version 2.0.0 of the open-source libkrylov library.[111]

Chapter 5

5.1 Conclusion and Outlook

The thesis presents an approach to solving the TDDFT response equations using the split-complex algorithm. The applicability of the split-complex approach for studying molecular response properties within the TDDFT framework is evaluated through testing with different systems. The successful application of the method to the tested systems instills confidence in its ability to accurately compute TDDFT response properties. It demonstrates that the split-complex algorithm can be a viable tool for studying the electronic structure and properties of molecular systems within the TDDFT framework.

Due to the nature of split-complex numbers, not being a field, concepts such multiplicative inverse or inner product space are not straightforward. One of the significant shortcomings of the method discussed in the thesis is the lack of such concept for the square root operation of split-complex numbers. This hampers the potential of the method to address the instabilities that arise in response calculations.

Addressing this limitation could involve exploring alternative numerical techniques, specialized algorithms, or approximations to handle the square root operation in a way that is compatible with the properties of split-complex numbers.

The limitations arising from the lack of readily available standard linear algebra libraries tailored for the split-complex type are less crucial. Standard linear algebra libraries, such as BLAS (Basic Linear Algebra Subprograms) and LAPACK (Linear Algebra Package), are widely used and optimized for efficient matrix operations. However, these libraries do not have built-in support for the split-complex type, which hinders the seamless integration of the method with existing linear algebra routines.

By leveraging standard linear algebra libraries specifically designed for the split-complex type, researchers would be able to benefit from optimized implementations and take advantage of parallel computing techniques, ultimately improving the computational efficiency of the method and enabling its wider application to larger systems.

Overall, the thesis contributes a new approach to solving the TDDFT response equations, showcasing its strengths and limitations through a few case studies. It provides valuable insights into the applicability of the split-complex algorithm for studying molecular response properties, while also highlighting the challenges that may arise with our approach.

Bibliography

[1] Boreen, M. A.; Arnold, J. The synthesis and versatile reducing power of low-valent uranium complexes. *Dalton Trans.* **2020**, *49*, 15124–15138.

[2] Hill, M. S.; Liptrot, D. J.; Weetman, C. Alkaline earths as main group reagents in molecular catalysis. *Chem. Soc. Rev.* **2016**, *45*, 972–988.

[3] Ortu, F.; Formanuik, A.; Innes, J. R.; Mills, D. P. New vistas in the molecular chemistry of thorium: low oxidation state complexes. *Dalton Trans.* **2016**, *45*, 7537–7549.

[4] Gould, C. A.; McClain, K. R.; Yu, J. M.; Groshens, T. J.; Furche, F.; Harvey, B. G.; Long, J. R. Synthesis and Magnetism of Neutral, Linear Metallocene Complexes of Terbium(II) and Dysprosium(II). *J. Am. Chem. Soc.* **2019**, *141*, 12967–12973.

[5] Volker, Z. *A New Approach to the Nexus of Supply, Demand and Use: Exemplified along the Use of Neodymium in Permanent Magnets*; Rare Earth Elements; Springer, Berlin, Heidelberg, 2013.

[6] Evans, W. J.; Davis, B. L. Chemistry of Tris(pentamethylcyclopentadienyl) f-Element Complexes, $(C_5Me_5)_3M$. *Chemical Reviews* **2002**, *102*, 2119–2136.

[7] Wedal, J. C.; Bekoe, S.; Ziller, J. W.; Furche, F.; Evans, W. J. In search of tris(trimethylsilylcyclopentadienyl) thorium. *Dalton Trans.* **2019**, *48*, 16633–16640.

[8] Huh, D. N.; Ciccone, S. R.; Bekoe, S.; Roy, S.; Ziller, J. W.; Furche, F.; Evans, W. J. Synthesis of Ln(II)-in-Cryptand Complexes by Chemical Reduction of Ln(III)-in-Cryptand Precursors: Isolation of a Nd(II)-in-Cryptand Complex. *Angew. Chem., Int. Ed.* **2020**, *59*, 16141–16146.

[9] Shavitt, I.; Bartlett, R. *Many-Body Methods in Chemistry and Physics: MBPT and Coupled-Cluster Theory*; Cambridge University Press, 2009.

[10] Trygve, H.; Poul, J.; Jeppe, O. *Molecular Electronic-Structure Theory*; John Wiley & Sons, 2000.

[11] McLachlan, A.; Ball, M. Time-Dependent Hartree–Fock Theory for Molecules. *Rev. Mod. Phys.* **1964**, *36*, 844–855.

[12] Saad, Y. *Iterative Methods for Sparse Linear Systems*, 2nd ed.; Society for Industrial and Applied Mathematics, 2003.

[13] Furche, F.; Krull, B. T.; Nguyen, B. D.; Kwon, J. Accelerating molecular property calculations with nonorthonormal Krylov space methods. *J. Chem. Phys.* **2016**, *144*, 174105.

[14] Olsen, J.; Jensen, H. J. A.; Jørgensen, P. Solution of the large matrix equations which occur in response theory. *J. Comput. Phys.* **1988**, *74*, 265–282.

[15] Kauczor, J.; Jørgensen, P.; Norman, P. On the Efficiency of Algorithms for Solving Hartree–Fock and Kohn–Sham Response Equations. *J. Chem. Theory Comput.* **2011**, *7*, 1610–1630.

[16] Dietrich, B.; Lehn, J.; Sauvage, J.; Blanzat, J. Cryptates–X: Syntheses et proprietes physiques de systemes diaza-polyoxa-macrobicycliques. *Tetrahedron* **1973**, *29*, 1629–1645.

[17] Dietrich, B.; Lehn, J. M.; Sauvage, J. P. Cryptates–XI: Complexes macrobicycliques, formation, structure, proprietes. *Tetrahedron* **1973**, *29*, 1647–1658.

[18] Burns, J. H. Crystal and molecular structure of a cryptate complex of samarium: $C_{18}H_{36}O_6N_2Sm_2(NO_3)_6 \cdot H_2O$. *Inorg. Chem.* **1979**, *18*, 3044–3047.

[19] Benetollo, F.; Bombieri, G.; Cassol, A.; De Paoli, G.; Legendziewicz, J. Coordination chemistry of lanthanides with cryptands. An X-ray and spectroscopic study of the complex $Nd_2(NO_3)_6[C_{18}H_{36}O_6N_2]$. *Inorganica Chim. Acta* **1985**, *110*, 7–13.

[20] Yang, G.; Liu, S.; Jin, Z. Coordination chemistry and structure characterization of $C_{18}H_{36}O_6N_2Eu_2(NO_3)_6$. *Inorganica Chim. Acta* **1987**, *131*, 125–128.

[21] Mao, J.; Jin, Z. Synthesis and structure characterization of lanthanum [2,2,2]cryptates, $[LaCl[2,2,2](H_2O)]Cl_2 \cdot H_2O$ and $[La(CF_3SO_3)[2,2,2](DMF)](CF_3SO_3)_2$. *Polyhedron* **1994**, *13*, 319–323.

[22] Gamage, N.-D. H.; Mei, Y.; Garcia, J.; Allen, M. J. Oxidatively Stable, Aqueous Europium(II) Complexes through Steric and Electronic Manipulation of Cryptand Coordination Chemistry. *Angew. Chem., Int. Ed.* **2010**, *49*, 8923–8925.

[23] Lenora, C. U.; Carniato, F.; Shen, Y.; Latif, Z.; Haacke, E. M.; Martin, P. D.; Botta, M.; Allen, M. J. Structural Features of Europium(II)-Containing Cryptates That Influence Relaxivity. *Chem. Eur. J.* **2017**, *23*, 15404–15414.

[24] Ekanger, L. A.; Polin, L. A.; Shen, Y.; Haacke, E. M.; Martin, P. D.; Allen, M. J. A Eu^{II}-Containing Cryptate as a Redox Sensor in Magnetic Resonance Imaging of Living Tissue. *Angew. Chem., Int. Ed.* **2015**, *54*, 14398–14401.

[25] Huh, D. N.; Ziller, J. W.; Evans, W. J. Facile Encapsulation of Ln(II) Ions into Cryptate Complexes from $LnI_2(THF)_2$ Precursors (Ln = Sm, Eu, Yb). *Inorg. Chem.* **2019**, *58*, 9613–9617.

[26] Huh, D. N.; Kotyk, C. M.; Gembicky, M.; Rheingold, A. L.; Ziller, J. W.; Evans, W. J. Synthesis of rare-earth-metal-in-cryptand dications, $[Ln(2.2.2\text{-cryptand})]^{2+}$, from Sm^{2+}, Eu^{2+}, and Yb^{2+} silyl metallocenes $(C_5H_4SiMe_3)_2Ln(THF)_2$. *Chem. Commun.* **2017**, *53*, 8664–8666.

[27] Jenks, T. C.; Kuda-Wedagedara, A. N. W.; Bailey, M. D.; Ward, C. L.; Allen, M. J. Spectroscopic and Electrochemical Trends in Divalent Lanthanides through Modulation of Coordination Environment. *Inorg. Chem.* **2020**, *59*, 2613–2620.

[28] Huh, D. N.; Windorff, C. J.; Ziller, J. W.; Evans, W. J. Synthesis of uranium-in-cryptand complexes. *Chem. Commun.* **2018**, *54*, 10272–10275.

[29] Jenks, T. C.; Bailey, M. D.; Hovey, J. L.; Fernando, S.; Basnayake, G.; Cross, M. E.; Li, W.; Allen, M. J. First use of a divalent lanthanide for visible-light-promoted photoredox catalysis. *Chem. Sci.* **2018**, *9*, 1273–1278.

[30] Bochkarev, M. N.; Fagin, A. A. A New Route to Neodymium(II) and Dysprosium(II) Iodides. *Chem. Eur. J.* **1999**, *5*, 2990–2992.

[31] Bochkarev, M. N.; Fedushkin, I. L.; Fagin, A. A.; Petrovskaya, T. V.; Ziller, J. W.; Broomhall-Dillard, R. N. R.; Evans, W. J. Synthesis and Structure of the First Molecular Thulium(II) Complex: $[TmI_2(MeOCH_2CH_2OMe)_3]$. *Angew. Chem., Int. Ed. in English* **1997**, *36*, 133–135.

[32] Evans, W. J.; Allen, N. T.; Ziller, J. W. The Availability of Dysprosium Diiodide as a Powerful Reducing Agent in Organic Synthesis: Reactivity Studies and Structural Analysis of $DyI_2((DME)_3$ and Its Naphthalene Reduction Product1. *J. Am. Chem. Soc.* **2000**, *122*, 11749–11750.

[33] Bochkarev, M. N.; Fedushkin, I. L.; Dechert, S.; Fagin, A. A.; Schumann, H. $[NdI_2(thf)_5]$, the First Crystallographically Authenticated Neodymium(II) Complex. *Angew. Chem., Int. Ed.* **2001**, *40*, 3176–3178.

[34] MacDonald, M. R.; Bates, J. E.; Ziller, J. W.; Furche, F.; Evans, W. J. Completing the Series of +2 Ions for the Lanthanide Elements: Synthesis of Molecular Complexes of Pr^{2+}, Gd^{2+}, Tb^{2+}, and Lu^{2+}. *J. Am. Chem. Soc.* **2013**, *135*, 9857–9868.

[35] MacDonald, M. R.; Bates, J. E.; Fieser, M. E.; Ziller, J. W.; Furche, F.; Evans, W. J. Expanding Rare-Earth Oxidation State Chemistry to Molecular Complexes of Holmium(II) and Erbium(II). *J. Am. Chem. Soc.* **2012**, *134*, 8420–8423.

[36] Fieser, M. E.; MacDonald, M. R.; Krull, B. T.; Bates, J. E.; Ziller, J. W.; Furche, F.; Evans, W. J. Structural, Spectroscopic, and Theoretical Comparison of Traditional vs Recently Discovered Ln^{2+} Ions in the $[K(2.2.2-\text{cryptand})][(C_5H_4SiMe_3)_3Ln]$ Complexes: The Variable Nature of Dy^{2+} and Nd^{2+}. *J. Am. Chem. Soc.* **2015**, *137*, 369–382.

[37] MacDonald, M. R.; Ziller, J. W.; Evans, W. J. Synthesis of a Crystalline Molecular Complex of Y^{2+}, $[(18-\text{crown}-6)\,K][(C_5H_4SiMe_3)_3Y]$. *J. Am. Chem. Soc.* **2011**, *133*, 15914–15917.

[38] Evans, W. J. Tutorial on the Role of Cyclopentadienyl Ligands in the Discovery of Molecular Complexes of the Rare-Earth and Actinide Metals in New Oxidation States. *Organometallics* **2016**, *35*, 3088–3100.

[39] Yee, E. L.; Gansow, O. A.; Weaver, M. J. Electrochemical studies of europium and ytterbium cryptate formation in aqueous solution. Effects of varying the metal oxidation state upon cryptate thermodynamics and kinetics. *J. Am. Chem. Soc.* **1980**, *102*, 2278–2285.

[40] Huh, D. N.; Ziller, J. W.; Evans, W. J. Chelate-Free Synthesis of the U(II) Complex, $[(C_5H_3(SiMe_3)_2)_3U]^{1-}$, Using Li and Cs Reductants and Comparative Studies of La(II) and Ce(II) Analogs. *Inorg. Chem.* **2018**, *57*, 11809–11814.

[41] Fieser, M. E.; Palumbo, C. T.; La Pierre, H. S.; Halter, D. P.; Voora, V. K.; Ziller, J. W.; Furche, F.; Meyer, K.; Evans, W. J. Comparisons of lanthanide/actinide +2 ions in a tris(aryloxide)arene coordination environment. *Chem. Sci.* **2017**, *8*, 7424–7433.

[42] Huh, D. N.; Roy, S.; Ziller, J. W.; Furche, F.; Evans, W. J. Isolation of a Square-Planar Th(III) Complex: Synthesis and Structure of $[Th(OC_6H_2{}^tBu_2{-}2,6{-}Me{-}4)4]^{1-}$. *J. Am. Chem. Soc.* **2019**, *141*, 12458–12463.

[43] Huh, D. N.; Ziller, J. W.; Evans, W. J. Isolation of reactive Ln(ii) complexes with C_5H_4Me ligands (CpMe) using inverse sandwich countercations: synthesis and structure of [(18-crown-6)K(μ-CpMe)K(18-crown-6)][$Cp^{Me}{}_3Ln^{II}$] (Ln = Tb, Ho). *Dalton Trans.* **2018**, *47*, 17285–17290.

[44] Ryan, A. J.; Angadol, M. A.; Ziller, J. W.; Evans, W. J. Isolation of U(II) compounds using strong donor ligands, C_5Me_4H and $N(SiMe_3)_2$, including a three-coordinate U(II) complex. *Chem. Commun.* **2019**, *55*, 2325–2327.

[45] Ryan, A. J.; Darago, L. E.; Balasubramani, S. G.; Chen, G. P.; Ziller, J. W.; Furche, F.; Long, J. R.; Evans, W. J. Synthesis, Structure, and Magnetism of Tris(amide) [LnN(SiMe3)23]1 Complexes of the Non-traditional +2 Lanthanide Ions. *Chem. Eur. J.* **2018**, *24*, 7702–7709.

[46] Langeslay, R. R.; Fieser, M. E.; Ziller, J. W.; Furche, F.; Evans, W. J. Synthesis, structure, and reactivity of crystalline molecular complexes of the $[C_5H_3(SiMe_3)_2]_3Th^{1-}$ anion containing thorium in the formal +2 oxidation state. *Chem. Sci.* **2015**, *6*, 517–521.

[47] Palumbo, C. T.; Darago, L. E.; Windorff, C. J.; Ziller, J. W.; Evans, W. J. Trimethylsilyl versus Bis(trimethylsilyl) Substitution in Tris(cyclopentadienyl) Complexes of La, Ce, and Pr: Comparison of Structure, Magnetic Properties, and Reactivity. *Organometallics* **2018**, *37*, 900–905.

[48] Huh, D. N.; Darago, L. E.; Ziller, J. W.; Evans, W. J. Utility of Lithium in Rare-Earth Metal Reduction Reactions to Form Nontraditional Ln^{2+} Complexes and Unusual $[Li(2.2.2\text{-cryptand})]^{1+}$ Cations. *Inorg. Chem.* **2018**, *57*, 2096–2102.

[49] Coughlin, E. J.; Zeller, M.; Bart, S. C. Neodymium(III) Complexes Capable of Multi-Electron Redox Chemistry. *Angew. Chem., Int. Ed.* **2017**, *56*, 12142–12145.

[50] Galley, S. S.; Pattenaude, S. A.; Higgins, R. F.; Tatebe, C. J.; Stanley, D. A.; Fanwick, P. E.; Zeller, M.; Schelter, E. J.; Bart, S. C. A reduction series of neodymium supported by pyridine(diimine) ligands. *Dalton Trans.* **2019**, *48*, 8021–8025.

[51] Anderson, N. H.; Yin, H.; Kiernicki, J. J.; Fanwick, P. E.; Schelter, E. J.; Bart, S. C. Investigation of Uranium Tris(imido) Complexes: Synthesis, Characterization, and Reduction Chemistry of $[U(NDIPP)_3(thf)_3]$. *Angew. Chem., Int. Ed.* **2015**, *54*, 9386–9389.

[52] Aguilar-Calderón, J. R.; Metta-Magaña, A. J.; Noll, B.; Fortier, S. $C(sp^3)H$ Oxidative Addition and Transfer Hydrogenation Chemistry of a Titanium(II) Synthon: Mimicry of Late-Metal Type Reactivity. *Angew. Chem., Int. Ed.* **2016**, *55*, 14101–14105.

[53] Du, J.; King, D. M.; Chatelain, L.; Lu, E.; Tuna, F.; McInnes, E. J. L.; Wooles, A. J.; Maron, L.; Liddle, S. T. Thorium- and uranium-azide reductions: a transient dithorium-nitride versus isolable diuranium-nitrides. *Chem. Sci.* **2019**, *10*, 3738–3745.

[54] Angadol, M. A.; Woen, D. H.; Windorff, C. J.; Ziller, J. W.; Evans, W. J. tert-Butyl(cyclopentadienyl) Ligands Will Stabilize Nontraditional +2 Rare-Earth Metal Ions. *Organometallics* **2019**, *38*, 1151–1158.

[55] Jenkins, T. F.; Woen, D. H.; Mohanam, L. N.; Ziller, J. W.; Furche, F.; Evans, W. J. Tetramethylcyclopentadienyl Ligands Allow Isolation of Ln(II) Ions across the Lanthanide Series in $[K(2.2.2\text{-cryptand})][(C_5Me_4H)_3Ln]$ Complexes. *Organometallics* **2018**, *37*, 3863–3873.

[56] Paul, R. C.; Anand, B. N.; Kapoor, R. Preparation and characterization of alkali-metal dimethylformamidyls. *Indian J. Chem.* **1975**, *13*, 1338–1340.

[57] Staroverov, V. N.; Scuseria, G. E.; Tao, J.; Perdew, J. P. Comparative assessment of a new nonempirical density functional: Molecules and hydrogen-bonded complexes. *J. Chem. Phys.* **2003**, *119*, 12129–12137.

[58] Grimme, S.; Antony, J.; Ehrlich, S.; Krieg, H. A consistent and accurate ab initio parametrization of density functional dispersion correction (DFT-D) for the 94 elements H-Pu. *J. Chem. Phys.* **2010**, *132*.

[59] Küchle, W.; Dolg, M.; Stoll, H.; Preuss, H. Energy-adjusted pseudopotentials for the actinides. Parameter sets and test calculations for thorium and thorium monoxide. *J. Chem. Phys.* **1994**, *100*, 7535–7542.

[60] Weigend, F.; Ahlrichs, R. Balanced basis sets of split valence, triple zeta valence and quadruple zeta valence quality for H to Rn: Design and assessment of accuracy. *Phys. Chem. Chem. Phys.* **2005**, *7*, 3297–3305.

[61] Schäfer, A.; Horn, H.; Ahlrichs, R. Fully optimized contracted Gaussian basis sets for atoms Li to Kr. *J. Chem. Phys.* **1992**, *97*, 2571–2577.

[62] Treutler, O.; Ahlrichs, R. Efficient molecular numerical integration schemes. *J. Chem. Phys.* **1995**, *102*, 346–354.

[63] Klamt, A.; Schüürmann, G. COSMO: a new approach to dielectric screening in solvents with explicit expressions for the screening energy and its gradient. *J. Chem. Soc., Perkin Trans. 2* **1993**, 799–805.

[64] Lide, D. R. *CRC Handbook of Chemistry and Physics, 85th edition*; CRC Press, 2008.

[65] Rappoport, D.; Furche, F. Property-optimized Gaussian basis sets for molecular response calculations. *J. Chem. Phys.* **2010**, *133*.

[66] VMD 1.9; available from `http://www.ks.uiuc.edu/Research/vmd/`.

[67] TURBOMOLE V7.4.1 2019, a development of University of Karlsruhe and Forschungszentrum Karlsruhe GmbH, 1989-2007, TURBOMOLE GmbH, since 2007; available from `http://www.turbomole.com`.

[68] Shannon, R. D. Revised effective ionic radii and systematic studies of interatomic distances in halides and chalcogenides. *Acta Crystallographica Section A* **1976**, *32*, 751–767.

[69] Dreuw, A.; Head-Gordon, M. Failure of Time-Dependent Density Functional Theory for Long-Range Charge-Transfer Excited States: The ZincbacteriochlorinBacteriochlorin and BacteriochlorophyllSpheroidene Complexes. *J. Am. Chem. Soc.* **2004**, *126*, 4007–4016.

[70] Autschbach, J. Charge-Transfer Excitations and Time-Dependent Density Functional Theory: Problems and Some Proposed Solutions. *ChemPhysChem* **2009**, *10*, 1757–1760.

[71] Alexander, F. et al. Exascale applications: skin in the game. *Philos. Trans. R. Soc., A* **2020**, *378*, 20190056.

[72] Kolev, T. et al. Efficient exascale discretizations: High-order finite element methods. *Int. J. High Perform. Comput. Appl.* **2021**, *35*, 527–552.

[73] Dongarra, J.; Grigori, L.; Higham, N. J. Numerical algorithms for high-performance computational science. *Philos. Trans. R. Soc., A* **2020**, *378*, 20190066.

[74] Kilina, S.; Kilin, D.; Tretiak, S. Light-Driven and Phonon-Assisted Dynamics in Organic and Semiconductor Nanostructures. *Chem. Rev.* **2015**, *115*, 5929–5978.

[75] Furche, F.; Krull, B. T.; Nguyen, B. D.; Kwon, J. Accelerating molecular property calculations with nonorthonormal Krylov space methods. *J. Chem. Phys.* **2016**, *144*, 174105.

[76] Saad, Y. *Iterative Methods for Sparse Linear Systems*, 2nd ed.; SIAM: Philadelphia, 2003.

[77] van der Vorst, H. A. *Iterative Krylov Methods for Large Linear Systems*; Cambridge University Press: Cambridge, 2003.

[78] Watkins, D. S. *The Matrix Eigenvalue Problem. GR and Krylov Subspace Methods*; SIAM: Philadelphia, 2007.

[79] Saad, Y. *Numerical Methods for Large Eigenvalue Problems*; SIAM: Philadelphia, 2011.

[80] Liesen, J.; Strakoš, Z. *Krylov Subspace Methods*; Oxford University Press: Oxford, 2012.

[81] Olshanskii, M. A.; Tyrtyshnikov, E. E. *Iterative Methods for Linear Systems*; SIAM: Philadelphia, 2014.

[82] Huang, C.; Liu, W.; Xiao, Y.; Hoffmann, M. R. iVI: An iterative vector interaction method for large eigenvalue problems. *J. Comput. Chem.* **2017**, *38*, 2481–2499.

[83] Roos, B. A new method for large-scale CI calculations. *Chem. Phys. Lett.* **1972**, *15*, 153–159.

[84] Almlöf, J.; Faegri, K.; Korsell, K. Principles for a direct SCF approach to LCAO-MO ab-initio calculations. *J. Comput. Chem.* **1982**, *3*, 385–399.

[85] Häser, M.; Ahlrichs, R. Improvements on the direct SCF method. *J. Comput. Chem.* **1989**, *10*, 104–111.

[86] Weiss, H.; Ahlrichs, R.; Häser, M. A direct algorithm for self-consistent-field linear response theory and application to C_{60}: Excitation energies, oscillator strengths, and frequency-dependent polarizabilities. *J. Chem. Phys.* **1993**, *99*, 1262.

[87] Bauernschmitt, R.; Häser, M.; Treutler, O.; Ahlrichs, R. Calculation of excitation energies within time-dependent density functional theory using auxiliary basis set expansions. *Chem. Phys. Lett.* **1997**, *264*, 573–578.

[88] Rappoport, D.; Furche, F. Analytical time-dependent density functional derivative methods within the RI-J approximation, an approach to excited states of large molecules. *J. Chem. Phys.* **2005**, *122*, 064105.

[89] Neese, F.; Olbrich, G. Efficient use of the resolution of the identity approximation in time-dependent density functional calculations with hybrid density functionals. *Chem. Phys. Lett.* **2002**, *362*, 170–178.

[90] Petrenko, T.; Kossmann, S.; Neese, F. Efficient time-dependent density functional theory approximations for hybrid density functionals: analytical gradients and parallelization. *J. Chem. Phys.* **2011**, *134*, 054116.

[91] Ko, C.; Malick, D. K.; Braden, D. A.; Friesner, R. A.; Martínez, T. J. Pseudospectral time-dependent density functional theory. *J. Chem. Phys.* **2008**, *128*, 104103.

[92] Maier, T. M.; Bahmann, H.; Kaupp, M. Efficient Semi-numerical Implementation of Global and Local Hybrid Functionals for Time-Dependent Density Functional Theory. *J. Chem. Theory Comput.* **2015**, *11*, 4226–4237.

[93] Sherrill, C. D.; Manolopoulos, D. E.; Martínez, T. J.; Michaelides, A. Electronic structure software. *J. Chem. Phys.* **2020**, *153*, 070401.

[94] Davidson, E. R. The iterative calculation of a few of the lowest eigenvalues and corresponding eigenvectors of large real-symmetric matrices. *J. Comput. Phys.* **1975**, *17*, 87–94.

[95] Liu, B. In *Numerical Algorithms in Chemistry: Algebraic Methods*; Moler, C., Shavitt, I., Eds.; Lawrence Berkeley Laboratory: Berkeley, 1978; pp 49–53.

[96] Crouzeix, M.; Philippe, B.; Sadkane, M. The Davidson Method. *SIAM J. Sci. Comput.* **1994**, *15*, 62–76.

[97] Tretiak, S.; Isborn, C. M.; Niklasson, A. M. N.; Challacombe, M. Representation independent algorithms for molecular response calculations in time-dependent self-consistent field theories. *J. Chem. Phys.* **2009**, *130*, 054111.

[98] Zuev, D.; Vecharynski, E.; Yang, C.; Orms, N.; Krylov, A. I. New algorithms for iterative matrix-free eigensolvers in quantum chemistry. *J. Comput. Chem.* **2015**, *36*, 273–284.

[99] Vogiatzis, K. D.; Ma, D.; Olsen, J.; Gagliardi, L.; de Jong, W. A. Pushing configuration-interaction to the limit: Towards massively parallel MCSCF calculations. *J. Chem. Phys.* **2017**, *147*, 184111.

[100] Barrett, R.; Berry, M.; Chan, T. F.; Demmel, J.; Donato, J.; Dongarra, J.; Eijkhout, V.; Pozo, R.; Romine, C.; van der Vorst, H. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*; SIAM: Philadelphia, 1994.

[101] Bai, Z., Demmel, J., Dongarra, J., Ruhe, A., van der Vorst, H., Eds. *Templates for the Solution of Algebraic Eigenvalue Problems. A Practical Guide*; SIAM: Philadelphia, 2000.

[102] MATLAB version 9.12.0.1884302 (R2022a). The Mathworks, Inc.: Natick, Massachusetts, 2022.

[103] Eaton, J. W.; Bateman, D.; Hauberg, S.; Wehbring, R. GNU Octave version 7.3.0 manual: a high-level interactive language for numerical computations. 2022.

[104] Lehoucq, R. B.; Sorensen, D. C.; Yang, C. *ARPACK Users' Guide*; SIAM: Philadelphia, 1998.

[105] Knyazev, A. V. Toward the Optimal Preconditioned Eigensolver: Locally Optimal Block Preconditioned Conjugate Gradient Method. *SIAM J. Sci. Comput.* **2001**, *23*, 517–541.

[106] Knyazev, A. V.; Argentati, M. E.; Lashuk, I.; Ovtchinnikov, E. E. Block Locally Optimal Preconditioned Eigenvalue Xolvers (BLOPEX) in Hypre and PETSc. *SIAM J. Sci. Comput.* **2007**, *29*, 2224–2239.

[107] Hernandez, V.; Roman, J. E.; Vidal, V. SLEPc: A scalable and flexible toolkit for the solution of eigenvalue problems. *ACM Trans. Math. Softw.* **2005**, *31*, 351–362.

[108] Romero, E.; Roman, J. E. A parallel implementation of Davidson methods for large-scale eigenvalue problems in SLEPc. *ACM Trans. Math. Softw.* **2014**, *40*, 13.

[109] Bueler, E. *PETSc for Partial Differential Equations: Numerical Solutions in C and Python*; SIAM, 2020.

[110] Parrish, R. M.; Hohenstein, E. G.; Martínez, T. J. "Balancing" the Block Davidson–Liu Algorithm. *J. Chem. Theory Comput.* **2016**, *12*, 3003–3007.

[111] Furche, F.; co workers, libkrylov, a Modular Open-Source Software Library for Extremely Large Eigenvalue and Linear Problems. `https://gitlab.com/libkrylov/libkrylov-stable`, 2022.

[112] Lawson, C. L.; Hanson, R. J.; Kincaid, D. R.; Krogh, F. T. Basic Linear Algebra Subprograms for Fortran Usage. *ACM Trans. Math. Softw.* **1979**, *5*, 308–323.

[113] Dongarra, J. J.; Du Croz, J.; Hammarling, S.; Hanson, R. J. An extended set of FORTRAN basic linear algebra subprograms. *ACM Trans. Math. Softw.* **1988**, *14*, 1–17.

[114] Dongarra, J. J.; Du Croz, J.; Hammarling, S.; Duff, I. S. A set of level 3 basic linear algebra subprograms. *ACM Trans. Math. Softw.* **1990**, *16*, 1–17.

[115] Anderson, E.; Bai, Z.; Bischof, C.; Blackford, L. S.; Demmel, J.; Dongarra, J.; Du Croz, J.; Greenbaum, A.; Hammarling, S.; McKenney, A.; Sorensen, D. *LAPACK Users' Guide*; SIAM: Philadelphia, 1999.

[116] Sleijpen, G. L. G.; van der Vorst, H. A. A Jacobi–Davidson Iteration Method for Linear Eigenvalue Problems. *SIAM Rev.* **2000**, *42*, 267–293.

[117] Hochstenbach, M. E.; Notay, Y. The Jacobi–Davidson method. *GAMM-Mitteilungen* **2006**, *29*, 368–382.

[118] Golub, G. H.; Van Loan, C. F. *Matrix Computations*, 4th ed.; Johns Hopkins University Press: Baltimore, 2013.

[119] Arnoldi, W. E. The principle of minimized iterations in the solution of the matrix eigenvalue problem. *Quart. Appl. Math.* **1951**, *9*, 17–29.

[120] Axelsson, O. A survey of preconditioned iterative methods for linear systems of algebraic equations. *BIT Numer. Math.* **1985**, *25*, 165–187.

[121] Bruaset, A. M. *A Survey of Preconditioned Iterative Methods*; CRC Press: Boca Raton, 1995.

[122] Benzi, M. Preconditioning Techniques for Large Linear Systems: A Survey. *J. Comput. Phys.* **2002**, *182*, 418–477.

[123] Wathen, A. J. Preconditioning. *Acta Numer.* **2015**, *24*, 329–376.

[124] Kitware, Inc., CMake. https://cmake.org, 2021.

[125] Thouless, D. J. *The Quantum Mechanics of Many-Body Systems*, 2nd ed.; Academic Press: New York, 1972.

[126] Casida, M. E. In *Recent Advances in Density Functional Methods*; Chong, D. E., Ed.; Recent Advances in Computational Chemistry; World Scientific: Singapore, 1995; Vol. 1; Chapter 5, pp 155–192.

[127] Bauernschmitt, R.; Ahlrichs, R. Treatment of electronic excitations within the adiabatic approximation of time dependent density functional theory. *Chem. Phys. Lett.* **1996**, *256*, 454–464.

[128] Furche, F. On the density matrix based approach to time-dependent density functional response theory. *J. Chem. Phys.* **2001**, *114*, 5982–5992.

[129] Stratmann, R. E.; Scuseria, G. E.; Frisch, M. J. An efficient implementation of time-dependent density-functional theory for the calculation of excitation energies of large molecules. *J. Chem. Phys.* **1998**, *109*, 8218–8224.

[130] Furche, F.; Rappoport, D. In *Computational Photochemistry*; Olivucci, M., Ed.; Elsevier: Amsterdam, 2005; Chapter III, pp 93–128.

[131] Tamm, I. In *Selected Papers*, 1st ed.; Bolotovskii, B. M., Frenkel, V. Y., Peierls, R., Eds.; Springer, 1991; Chapter N.4, pp 157–174.

[132] Dancoff, S. M. Non-Adiabatic Meson Theory of Nuclear Forces. *Phys. Rev.* **1950**, *78*, 382–385.

[133] Hirata, S.; Head-Gordon, M. Time-dependent density functional theory within the Tamm—Dancoff approximation. *Chem. Phys. Lett.* **1999**, *314*, 291–299.

[134] Foresman, J. B.; Head-Gordon, M.; Pople, J. A.; Frisch, M. J. Toward a Systematic Molecular Orbital Theory for Excited States. *J. Phys. Chem.* **1992**, *96*, 135–149.

[135] Perdew, J. P.; Burke, K.; Ernzerhof, M. Generalized Gradient Approximation Made Simple. *Phys. Rev. Lett.* **1996**, *77*, 3865–3868.

[136] Eichkorn, K.; Treutler, O.; Öhm, H.; Häser, M.; Ahlrichs, R. Auxiliary basis sets to approximate Coulomb Potentials (erratum, 1995, **242**, 283). *Chem. Phys. Lett.* **1995**, *242*, 652–660.

[137] Weigend, F.; Ahlrichs, R. Balanced basis sets of split valence, triple zeta valence and quadruple zeta valence quality for H to Rn: Design and assessment of accuracy. *Phys. Chem. Chem. Phys.* **2005**, *7*, 3297–3305.

[138] Rappoport, D.; Furche, F. Property-optimized Gaussian basis sets for molecular response calculations. *J. Chem. Phys.* **2010**, *133*, 134105.

[139] Balasubramani, S. G. et al. TURBOMOLE: Modular program suite for ab initio quantum-chemical and condensed-matter simulations. *J. Chem. Phys.* **2020**, *152*, 184107.

[140] Parlett, B. N. *The Symmetric Eigenvalue Problem*; SIAM: Philadelphia, 1998.

[141] Horn, R. A.; Johnson, C. R. *Matrix Analysis*, 2nd ed.; Cambridge University Press: Cambridge, 2012.

[142] Notay, Y. Is Jacobi–Davidson Faster than Davidson? *SIAM J. Matrix Anal. Appl.* **2004**, *26*, 522–543.

[143] Sekino, H.; Bartlett, R. J. A linear response, coupled-cluster theory for excitation energy. *Int. J. Quantum Chem.* **1984**, *26*, 255–265.

[144] Koch, H.; Jørgensen, P. Coupled cluster response functions. *J. Chem. Phys.* **1990**, *93*, 3333–3344.

[145] Hirao, K.; Nakatsuji, H. A generalization of the Davidson's method to large nonsymmetric eigenvalue problems. *J. Comput. Phys.* **1982**, *45*, 246–254.

[146] Flament, J. P.; Gervais, H. P. Equations-of-motion method: Calculation of the k lowest or highest solutions. *International Journal of Quantum Chemistry* **1979**, *16*, 1347–1356.

[147] Bauernschmitt, R.; Ahlrichs, R. Stability analysis for solutions of the closed shell Kohn–Sham equation. *J. Chem. Phys.* **1996**, *104*, 9047–9052.

[148] Čížek, J.; Paldus, J. Stability Conditions for the Solutions of the Hartree—Fock Equations for Atomic and Molecular Systems. Application to the Pi-Electron Model of Cyclic Polyenes. *J. Chem. Phys.* **2004**, *47*, 3976–3985.

[149] Tóth, Z.; Pulay, P. Finding symmetry breaking Hartree-Fock solutions: The case of triplet instability. **2016**, *145*, 164102.

[150] Kowalski, K.; Jankowski, K. Towards Complete Solutions to Systems of Nonlinear Equations of Many-Electron Theories. **1998**, *81*, 1195–1198.

[151] Bulik, I. W.; Henderson, T. M.; Scuseria, G. E. Can Single-Reference Coupled Cluster Theory Describe Static Correlation? **2015**, *11*, 3171–3179.

[152] Szalay, P. G.; Vázquez, J.; Simmons, C.; Stanton, J. F. Triplet instability in doublet systems. **2004**, *121*, 7624–7631.

[153] Crawford, T. D.; Stanton, J. F.; Allen, W. D.; Schaefer, H. F. Hartree–Fock orbital instability envelopes in highly correlated single-reference wave functions. **1997**, *107*, 10626–10632.

[154] Roos, B. O.; Taylor, P. R.; Siegbahn, P. E. M. A complete active space SCF method (CASSCF) using a density matrix formulated super-CI approach. **1980**, *48*, 157–173.

[155] Roos, B. O. In *Ab Initio Methods in Quantum Chemistry*; Lawley, K. P., Ed.; Advances in Chemical Physics; John Wiley & Sons: Hoboken, NJ, USA, 1987; Vol. 69; pp 399–445.

[156] Roos, B. O. In *Lecture Notes in Quantum Chemistry: European Summer School in Quantum Chemistry*; Roos, B. O., Ed.; Springer: Berlin, Heidelberg, 1992; pp 177–254.

[157] Furche, F. On the density matrix based approach to time-dependent density functional response theory. *J. Chem. Phys.* **2001**, *114*, 5982–5992.

[158] Furche, F.; Ahlrichs, R. Adiabatic time-dependent density functional methods for excited state properties. **2002**, *117*, 7433–7447.

[159] Furche, F.; Burke, K. *Chapter 2 Time-Dependent Density Functional Theoryin Quantum Chemistry*; Annual Reports in Computational Chemistry; Elsevier, 2005; Vol. 1; pp 19–30.

[160] Parker, S. M.; Rappoport, D.; Furche, F. Quadratic Response Properties from TDDFT: Trials and Tribulations. **2018**, *14*, 807–819.

[161] Jørgensen, P.; Linderberg, J. Time-dependent Hartree–Fock calculations in the Pariser–Parr–Pople model. Applications to aniline, azulene and pyridine. *International Journal of Quantum Chemistry* **1970**, *4*, 587–602.

[162] Sobczyk, G. The Hyperbolic Number Plane. *Coll. Math. J.* **1995**, *26*, 268–280.

[163] de la Cruz, R. J.; Faßbender, H. On the diagonalizability of a matrix by a symplectic equivalence, similarity or congruence transformation. *Linear Algebra and its Applications* **2016**, *496*, 288–306.

[164] Bauer, F. L.; Fike, C. T. Norms and exclusion theorems. *Numer. Math.* **1960**, *2*, 137–141.

[165] Rappoport, D.; Bekoe, S.; Mohanam, L. N.; Le, S.; George, N.; Shen, Z.; Furche, F. Libkrylov: A modular open-source software library for extremely large on-the-fly matrix computations. *Journal of Computational Chemistry* **2023**, *44*, 1105–1118.

[166] Abu-Eittah, R. H.; El-Tawil, B. A. H. The electronic absorption spectra of some coumarins. A molecular orbital treatment. *Canadian Journal of Chemistry* **1985**, *63*, 1173–1179.

Appendix A

Split-Complex Type

A.1 Introduction

Due to the lack of a native type support for split-complex numbers in compiled languages, we had to develop our own implementation. Split-complex numbers consist of two real numbers, typically denoted as (x, y) like the traditional complex numbers, where the imaginary part is defined as the product of the split-imaginary unit j and the second real number.

To handle split-complex numbers, we created a custom data structure or class that encapsulates the two real components and provides operations for arithmetic and algebraic manipulations. This allows us to perform computations involving split-complex numbers in our code, even though the language itself does not natively support this data type.

By implementing our own split-complex number type, we can effectively work with these numbers and apply them in the context of our algorithms and calculations.

```
module split_complex

    type :: scomplex
        real(kind) :: re, im
    end type scomplex

    interface assignment( )
        module procedure :: scomplex_assign
    end interface assignment( )

    interface operator( )
        module procedure :: scomplex_arithmetic
    end interface operator( )

end module split_complex
```

Figure A.1: Split-complex derived type in FORTRAN

A.2 Efficient Operations

Addition and subtraction operations in the split-complex type are straight forward. Consider multiplication of a split-complex number (x, y) by another split-complex number or real number r.

A.2.1 Standard basis operations

Multiplication of a split-complex number by a real number

$$z = (x, y) * r$$

$$z\%re = x * r$$

$$z\%im = y * r$$

Multiplication of two split-complex numbers

$$z = (x, y) * (u, v)$$

$$z\%re = (x * u) + (y * v)$$

$$z\%im = (x * v) + (y * u)$$

Division of a split-complex number by a real number

$$z = (x, y)/r$$

$$z\%re = x/r$$

$$z\%im = y/r$$

Division of a real number by a split-complex number

$$z = \frac{r}{(x, y)}$$
$$z = \frac{r * (x, -y)}{(x, y)(x, -y)}$$
$$z\%re = \frac{r * x}{(x + y)(x - y)} = 0.5 * (\frac{r}{x + j} + \frac{r}{x - y})$$
$$z\%im = -\frac{r * y}{(x + y)(x - y)} = 0.5 * (\frac{r}{x + y} - \frac{r}{x - y})$$

Division of a split-complex number by a split-complex number

$$z = \frac{x, y}{u, vj}$$
$$z = \frac{(x, y)(u, -v)}{(u, v)(u, -v)}$$
$$z\%re = \frac{x * u - y * v}{(u + v)(u - v)} = 0.5 * (\frac{x + y}{u + v} + \frac{x - y}{u - v})$$
$$z\%im = \frac{x * v - y * u}{(u + v)(u - v)} = 0.5 * (\frac{x + y}{u + v} - \frac{x - y}{u - v})$$

A.2.2 Null basis Operation

Multiplication of a split-complex number by a real number

$$z = (p, q) * r$$

$$z\%re = p * r$$

$$z\%im = q * r.$$

Multiplication of two split-complex numbers

$$z = (p, q) * (r, s)$$

$$z\%re = p * r$$

$$z\%im = q * s.$$

Division of a split-complex number by a real number

$$z = (p, q)/r$$

$$z\%re = p/r$$

$$z\%im = q/r.$$

Division of a real number by a split-complex number

$$z = \frac{r}{(p, q)}$$

$$z = \frac{r(q, p)}{(p, q)(q, p)}$$

$$z\%re = \frac{r * q}{pq}$$

$$z\%im = \frac{r * p}{pq}.$$

Division of a split-complex number by a split-complex number

$$z = \frac{(p,q)}{(s,t)}$$
$$z = \frac{(p+q)(t+s)}{(s+t)(t+s)}$$
$$z\%re = \frac{p*t}{s*t}$$
$$z\%im = \frac{q*s}{s*t}.$$

A.3 Matrix Computation of Split-Complex Elements

In the absence of BLAS (Basic Linear Algebra Subprograms) routines specifically designed for split-complex numbers, we have adapted the existing BLAS interface to work with this data type. BLAS provides a standard interface for performing common linear algebra operations, such as matrix-vector multiplication, dot products, and matrix factorizations.

To adapt the BLAS routines for split-complex numbers, we modified the implementations to handle the split-complex data structure and perform the required arithmetic operations accordingly. This allows us to leverage the efficiency and optimized implementations of the BLAS routines while working with split-complex numbers.

By adapting the BLAS routines, we can benefit from their high performance and scalability, making our computations involving split-complex numbers more efficient and convenient. This approach enables us to seamlessly integrate split-complex computations into our codebase and leverage the power of BLAS for numerical linear algebra operations.

In our implementation, we have adapted the following BLAS routines for our work with split-complex numbers:

- `scomplex_nrm2` calculates the pseudo-norm of a split-complex vector. It takes a split-complex vector `x` and returns the difference of the squares of the real and split-imaginary parts of the vector elements. The result represents the magnitude or length of the vector.

- `scomplex_dot`: This routine calculates the dot product of two split-complex vectors. It takes two split-complex vectors as input and returns their dot product.

- `scomplex_axpy`: This routine performs the operation `y = alpha * x + y`, where `alpha` is a scalar and `x` and `y` are split-complex vectors. It computes the scaled sum of two split-complex vectors and stores the result in the `y` vector.

- `scomplex_scal` is used to scale a split-complex vector by a scalar factor. It takes a scalar value `alpha`, a split-complex vector `x`, and modifies `x` by multiplying each element of `x` by `alpha`. This routine is particularly useful for scaling split-complex vectors efficiently.

- `scomplex_gemv`: This routine performs the matrix-vector multiplication `y = alpha * A * x + beta * y`, where `alpha` and `beta` are scalars, `A` is a split-complex matrix, `x` is a split-complex vector, and `y` is the resulting split-complex vector. It computes the product of a split-complex matrix and a split-complex vector and adds it to another split-complex vector.

- `scomplex_gemm`: This routine performs the matrix-matrix multiplication `C = alpha * A * B + beta * C`, where `alpha` and `beta` are scalars, `A` and `B` are split-complex matrices, and `C` is the resulting split-complex matrix. It computes the product of two split-complex matrices and adds it to another split-complex matrix.

These adapted BLAS routines allow us to perform essential operations involving split-complex numbers efficiently. They enable us to handle matrix-vector and matrix-matrix

multiplications, dot products, and vector additions with split-complex numbers in a straightforward and optimized manner, enhancing the performance of our algorithms and computations.