ABSTRACT

NANCE, JAMES DANIEL. Investigating Molecular Dynamics with Sparse Grid Surrogate Models. (Under the direction of Carl T. Kelley.)

Molecules in nature conform to a geometry that minimizes their potential energy, and some molecules have multiple potential energy minima. One can study how a molecule transitions from one stable geometry to another by studying dynamics on its potential energy surface. The potential energy of a molecule with N atoms is a function of $3N-6$ molecular coordinates and is computed via an expensive optimization process, thus modeling reaction pathways in all $3N-6$ coordinates can be cumbersome for large molecules. In this thesis we describe a cheaper surrogate model for the potential energy surfaces constructed using a sparse grid interpolation algorithm initially developed by Smolyak [198]. Evaluation of the surrogate is much less expensive than the evaluation of the actual energy function, so our technique offers a more computationally efficient way to study dynamics than traditional methods. Furthermore, molecular vibrations and thermal fluctuations can cause randomness in dynamics, so it is of interest to follow many reaction paths at once, necessitating a fast and efficient implementation of Smolyak's interpolation algorithm. We describe a new implementation that computes analytical gradients of Smolyak's interpolating polynomial and is designed to evaluate a large number of points simultaneously. We compare performance times of our implementation to MATLAB's Sparse Grid Interpolation Toolbox [121] and present dynamical simulations for various test molecules.

We also describe how one could extend our new reaction path following method to nonadiabatic dynamics, or dynamics where the Born-Oppenheimer approximation breaks down. In particular, we are interested in studying intersystem crossing dynamics of iron-based molecular complexes for an application to solar cells. We present three-dimensional potential energy surfaces for the $[\mathrm{Fe(terpy)_2}]^{2+}$ complex, the first time these surfaces have been studied in more than two dimensions.

Finally, we employ sparse grids for Bayesian inference for a groundwater model. Interpolation and integration on sparse grids offer an alternative to other expensive methods such as Markov Chain Monte Carlo (MCMC) algorithms to estimate summary statistics for quantities of interest. Here we interpolate the likelihood function and compute marginal densities using sparse grids for four parameters to verify results from MCMC.

Investigating Molecular Dynamics with Sparse Grid Surrogate Models

by
James Daniel Nance

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Applied Mathematics

Raleigh, North Carolina

2015

APPROVED BY:

Elena Jakubikova	Jordan E. Massad
Ralph Smith	Carl T. Kelley Chair of Advisory Committee

BIOGRAPHY

James Nance was born on June 20, 1989 in Knoxville, TN to parents Brad and Catherine Nance. He graduated from Farragut High School in Knoxville in 2007, and in 2011 he graduated with Highest Honors from Emory University with degrees in both Applied Mathematics and Music Performance. In August of 2011 James started graduate studies in Applied Mathematics at North Carolina State University and earned his M.S. in December of 2013.

ACKNOWLEDGEMENTS

I am extremely lucky to have had Tim Kelley for an advisor. Tim cares deeply about his students and works very hard to ensure our success. His sense of humor, passion for mathematics, and constant motivation made my graduate career enjoyable.

I would also like to thank each of my committee members: Jordan Massad, for being a wonderful mentor during my internship at Sandia National Laboratories; Ralph Smith, for being one of the most engaging lecturers and supportive professors; and Elena Jakubikova, who taught me everything I know about quantum chemistry and who has been a wonderful collaborator.

Finally, I would like to thank my parents for their constant encouragement, love, and support. I would not be where I am today without them.

This work has been partially supported by Army Research Office Grant W911NF-11-1-0367, and NSF Grants CDI-0941253, DMS-1406349, SI2-SSE-1339844, and DMS-1127914 to the Statistical and Applied Mathematical Sciences Institute. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Army Research Office or the National Science Foundation.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

Molecules in nature conform to a geometry that minimizes their potential energy, and some molecules have multiple potential energy minimizers. There is great interest in studying all the possible stable geometries of molecules, with the goal of yielding new compounds with the necessary characteristics for various needs of science and technology. For example, laser technology and molecular electronics require chemical compounds with special structures which can easily and rapidly be changed in a specified direction [131, 31]. Other applications include molecular pharmacology, conversion and storage of solar energy, photochemistry, biopolymers and polyelectrolytes [94], laser stimulation of chemical reaction [189], biological sensing [226] and many others [142]. Recently, selective isomerization of photochromes have allowed the design of simple Boolean logic devices [5]. Furthermore, one can study how a molecule transitions from one geometry to another by studying dynamics on its potential energy surface.

A molecule's potential energy depends on the interatomic forces between all of its electrons and nuclei, thus a molecule's potential energy is a function of its geometry. The geometry of a molecule can be uniquely determined by nuclear coordinates consisting of bond lengths, bond angles, and dihedral angles. A nonlinear molecule requires $3N-6$ nuclear coordinates $\mathbf{p} \in \mathbb{R}^{3N-6}$, whereas a linear molecule requires only $\mathbf{p} \in \mathbb{R}^{3N-5}$. For a given geometry, the molecule's potential energy $\mathcal{E}_i$ is found by approximating a solution the time-independent Schrödinger equation

$$\hat{\mathcal{H}}\Psi_i(\mathbf{p}) = \mathcal{E}_i\Psi_i(\mathbf{p})$$

where $\hat{\mathcal{H}}$ is the molecular Hamiltonian operator and Ψ_i is the molecular wavefunction for electronic state $i \in \mathbb{N}^0$. Potential energy is a quantized value, meaning that for a fixed geometry $\mathbf{p}$ the molecule's potential energy takes on discrete values that make up a a countable subset of the real numbers. That is, the energies $\mathcal{E}_i$ are such that $\mathcal{E}_0 < \mathcal{E}_1 \leq \mathcal{E}_2 \leq \dots$. As a function of geome-

try, $\mathcal{E}_i(\mathbf{p})$ is called the *potential energy surface* (PES) for electronic state $i \in \mathbb{N}^0$. Approximating a solution is no trivial task. Popular approaches such as the Hartree-Fock Self-Consistent-Field method [177, 22] or Density functional theory [107, 126, 125] involve writing the wavefunction as $\Psi_i = \sum_{j=1}^{M} c_j \phi_j$ for some set of basis functions $\{\phi_j\}$ and iteratively varying the coefficients c_i to minimize the energy $\mathcal{E}_i$. If n is the number of basis functions, the computational cost of these methods is $\mathcal{O}(n^3)$ for density functional theory and $\mathcal{O}(n^4)$ for Hartree-Fock [125]. More accurate and thus computationally intensive methods scale can scale as $\mathcal{O}(n^6)$ or even $\mathcal{O}(n^7)$.

One can cause a molecule to transition from one electronic state to another by exciting electrons with photons. When an electron is excited it will change states to occupy a higher energy orbital. Once excited to a new state, the molecule conforms to minimize its potential energy in the new electronic state. There is no guarantee that a molecular geometry that is a minimum for one energy state will be a minimum for the next excited state. By exciting and relaxing a molecule through one or several energy states back to its lowest energy state, or ground state, it is possible for a molecule to conform to a different ground state geometry from which it began. Other than simply having different molecular geometries, some molecules have different properties in different ground state potential energy minima. For example, the boiling point of cis-2-butene is different from that of trans-2-butene [61].

The conformational path that the molecule takes from one geometry to another is called the *reaction path* and is the solution to

$$\dot{\mathbf{p}} = -\nabla \mathcal{E}_i(\mathbf{p}). \tag{1.1}$$

Geometrically, the reaction path is the path of continuous steepest descent on the PES from the initial geometry to a local minimum.

Since many coordinates do not change significantly during state transitions, we can reduce the dimensionality of the dynamics by partitioning the molecular coordinates into design variables $\mathbf{x}$ and remainder variables ξ. Then, we can compute the reaction path as

$$\dot{\mathbf{x}} = -\nabla E_i(\mathbf{x}) \tag{1.2}$$

where now the energy is computed by freezing the design variables and optimizing the remainder variables via

$$E_i(\mathbf{x}) = \min_{\xi} \mathcal{E}_i(\mathbf{x}, \xi). \tag{1.3}$$

For example, the isomerization of 2-butene takes place via the first excited electronic state and can be modeled as a function of the rotation of the double bond connecting the two central carbon atoms. Figure 1.1 shows this one-dimensional reaction path from *cis*-2-butene to *trans*-

2-butene.

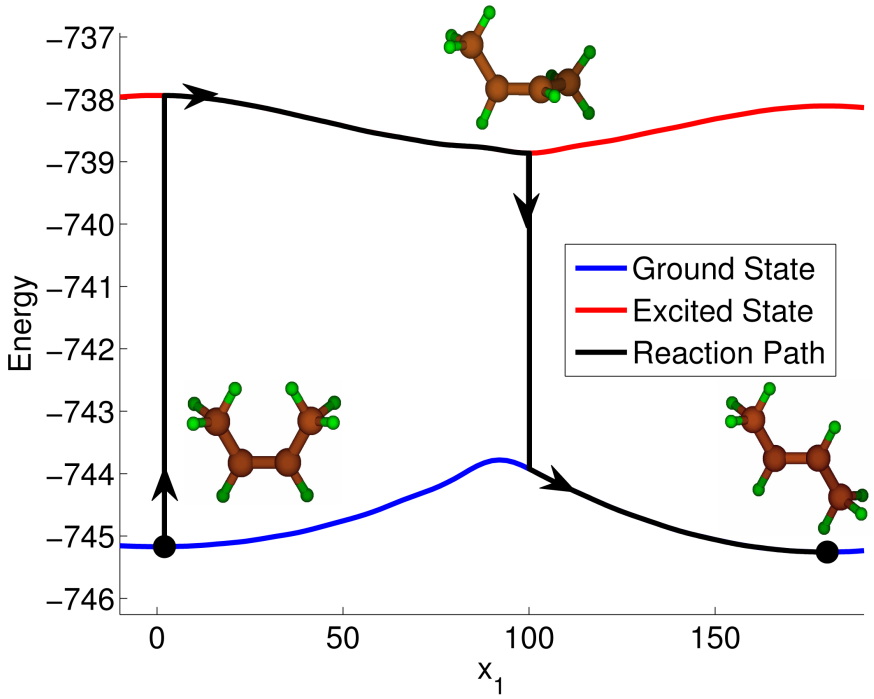


Figure 1.1: The isomerization of 2-butene. The blue line is the ground state PES, the red line is the first excited state PES, and the black line is the reaction path from *cis* to *trans*.

While this partitioning of geometric coordinates reduces the dimensionality of the reaction path dynamics, these simulations are still limited to relatively few design coordinates $\mathbf{x}$ because of the computational cost of the energy $E_i(\mathbf{x})$ in Equation 1.3 and its gradient in Equation 1.2. Each iteration of any standard optimization algorithm for Equation 1.3 involves approximating a solution to the time-independent Schrödinger equation, a task which is prohibitively burdensome for large molecules. The aim of this thesis is to replace the PES $E_i(\mathbf{x})$ with a computationally inexpensive surrogate model. The technique we use to do so constructs surrogate PESs by means of interpolation. Standard interpolation methods on full grids, however, are restricted to small molecules because of the computational cost of constructing the PESs to within a sufficient accuracy. The number of grid points required for interpolation on full grids grows exponentially with dimension, a problem known as the *curse of dimensionality*. As such, we build a surrogate model using a sparse grid interpolation algorithm initially developed by Smolyak [198]. Smolyak's sparse grid interpolation algorithm approximates a function with a

linear combination of tensor product interpolations on different levels of sparse grids. Sparse grid points grow only *polynomially* with the dimension and thus help alleviate the curse of dimensionality. Consequently, sparse grids enable us to increase the number of design variables for reaction path simulations.

Once we calculate our surrogate PESs, our molecular dynamics simulation follows the relaxation path of a molecule back to its ground state on these surrogate PESs after an initial light-induced excitation to a higher energy state. The goal is to identify the end configuration and report the entire path so that one can look for nearby paths to other interesting configurations. We also wish to investigate the effects of thermal fluctuations in molecular dynamics which can cause a molecule to jump over low energy barriers from one path to another. To do this we must simultaneously follow reaction paths of several different trajectories, necessitating a fast and efficient implementation of Smolyak's algorithm. In this thesis we present an implementation of Smolyak's algorithm that is specifically designed for this application.

Our implementation utilizes Judd et al. 's reformulation of Smolyak's algorithm that rearranges terms to eliminate redundant calculations [118]. Their implementation is used for an application to derivative-free dynamic economic models, where Smolyak's interpolating polynomial must be evaluated at a large number of points *sequentially*. On the other hand, our application requires evaluating Smolyak's interpolating polynomial *and* its gradient at a large number of points both *sequentially* and *simultaneously*. As such, we extended the work of Judd et al. in these two ways: first, our implementation evaluates the analytical gradient of Smolyak's interpolating polynomial. Second, our implementation is designed to quickly evaluate the interpolating polynomial and its gradient at a large number of points simultaneously by recursively computing the unidimensional basis polynomials. Our implementation of Smolyak's algorithm has been accepted for publication [152].

With our new implementation we are able to improve upon the reaction path following method of Mokrauer et al. and efficiently follow multiple reaction paths simultaneously. Our method allows one to visualize the entire PES landscape before simulating the reaction process, a task that is often unfeasible with conventional reaction path following methods. Also, our method allows one to increase the number of degrees of freedom in simulations and track entire reaction paths on all involved electronic state PESs. In [151] and this thesis, for example, we study the photoisomerization of 2-butene with 6 degrees of freedom for the first time. Prior to this work, simulating this electronic state reaction has been limited to one [179], two [6, 174], or three [146] molecular coordinates.

Finally, we use Smolyak's sparse grid interpolation algorithm to study the PESs of Fe(II)-polypyridines. These complexes are of great interest to the chemical community because of their potential application to solar cells [93, 51, 159, 27, 239]. Sunlight can be converted to electricity in solar cells via interfacial electron transfer (IET) between a molecular complex and

a semiconductor. The most successful class of complexes is based on Ru(II)-polypyridines. While efficient, Ru is a relatively rare and expensive metal. On the other hand, Fe is common and inexpensive, but Fe-based complexes are not as efficient as their Ru counterparts. The presence of many electronic states of various spin multiplicities impedes successful IET. The study of their PESs could lead to a better understanding of the photochemical processes that take place during IET, and ultimately to the design of more efficient Fe-based complexes for solar energy applications. With our sparse grid PESs, we are also able to compute entire intersection seams between PES and locate minimum energy crossing points (MECPs) on these seams. The changes in molecular spin state during IET necessitate modifications to our reaction path following method that account for more quantum effects. As such, we provide a theoretical framework that extends our reaction path following method to Tully's surface hopping algorithm [211], a mixed quantum mechanical/classical mechanical method for studying such dynamics.

This thesis is outlined as follows: Chapter 2 reviews sparse grids and presents our implementation Smolyak's interpolation algorithm. Chapter 3 gives a brief introduction to quantum chemistry and outlines the computational chemistry algorithms used in energy optimizations and Chapter 4 details our reaction path following method and presents simulation results for several different molecules. In we Chapter 5 motivate our application of sparse grids to study PESs of Fe(II)-polypyridines, present results for the Fe(II)-polypyridine $[Fe(tpy)_2]^{2+}$, and provide a theoretical framework for a sparse grid surrogate-based implementation of Tully's surface hopping algorithm.

Chapter 2

Sparse Grids

2.1 Background

In 1963 Smolyak studied tensor product problems and introduced a general approach that uses optimal approximations from the unidimensional case to yield an almost optimal approximation for $d > 1$ dimensions [198]. The method is a discretization technique that constructs a multidimensional, multilevel basis by the tensor product expansion of a one-dimensional multilevel basis, and, compared to full grids, improves the ratio of invested storage and computing time to approximation accuracy [35]. The method is closely related to the blending methods for interpolation and approximation of Gordon [80], the Boolean interpolation method of Delvos [53] and an iterative interpolation process developed by Deslauriers and Dubuc [54]. The grid points used in Smolyak's algorithm form what is called a "sparse grid."

Since their inception, sparse grids have gained a significant amount of traction in the mathematical community, especially since the advent of supercomputers and the need for efficient methods for high dimensional problems. In this section we will present a brief, chronological literature review of sparse grids and their applications. Bungartz and Griebel present a comprehensive review of sparse grids in [37] that is much easier to read and digest than the English translation of Smolyak's original paper [198], and the interested reader is encouraged to start there for a more thorough introduction to sparse grids. Later in the chapter we will describe the interpolation method we use in detail and prove some its nice properties.

In the late 1980's, Bank, Dupont and Yserentant [232, 10] built off of Smolyak's multigrid foundation [198] and introduced the hierarchical basis multigrid method to solve elliptic boundary value problems. In 1991 Zenger [237] formally introduced the idea of "sparse grids" and applied them as a numerical scheme to solve partial differential equations. Like some finite-element methods, Zenger's technique used piecewise linear elements but also incorporated the aforementioned hierarchical basis method of [232, 10]. The sparse grid finite element technique

can be interpreted as a multigrid combination extrapolation technique to solve elliptic boundary value problems[89, 34]. In 1994, Stortkfuhl took the first step towards higher order sparse grid techniques by using a piecewise bicubic hierarchical Hermite basis [203], quickly followed by Bungartz in 1996, who used quadratic hierarchical splines for Lagrangian interpolation on sparse grids [32]. Bungartz then worked with Dirnstorfer to extend this technique for arbitrary dimension d and polynomial basis degree p in [55, 33, 36].

In 1995 Wasilkowski and Wozniakowski derived explicit cost bounds for algorithms which compute an approximation to a solution to within a given error tolerance, as well as upper bounds for the number of points required for such an approximation [222]. As interest in sparse grids and the dimensions of problems grew, it became necessary to begin investigating efficient algorithms and data structures for implementing sparse grids. In the same year, Balder and Zenger developed an efficient approach where all of the algorithmic work is done in a single dimension [9]. In 1996 Novak and Ritter [155] developed an algorithm for numerical integration on a d-dimensional cube using Smolayk's algorithm with the one-dimensional Clenshaw-Curtis rule [45]. Later, in 1999, they extended this work for cubature formulas with high polynomial exactness in [156]. Also working within the realm of numerical quadrature, in Gerstner and Griebel reviewed several multivariate quadrature formulas on sparse grids based on several different one-dimensional quadrature rules in [77]. Griebel also independently made an important stride with the successful implementation of adaptive sparse grids based on finite differences in [86]. The hierarchical basis construction allows one to estimate the error of an approximation, and Griebel used this fact to develop an adaptive algorithm for elliptic partial differential equations based on finite differences.

In 2000, Sprengel derived error estimates for interpolation on Gauss-Chebyshev grids for functions from special kinds of Besov-type spaces[199], Garcke and Griebel extended the combination technique [89] for d-dimensional eigenproblems on sparse grids and used it to solve the Schrodinger equations for hydrogen and helium [74]. This combination technique was studied further and generalized by Hegland, Garcke and Challis in [102]. Barthelmann worked with Novak and Ritter to develop a polynomial interpolation method using sparse grids with Chebyshev nodes [13]. In 2003 Achatz used a sparse grid method that employed higher order finite elements for discretizing and solving partial differential equations with variable coefficients [1] and Petras investigated how to obtain a Smolyak cubature formula with a given degree of polynomial exactness and studied the asymptotically minimal number of nodes required to do so [164]. In 2005 Gajda studied Smolyak's algorithm for weighted L_1-approximation of multivariate functions and derived the asymptotic behavior of the error [73].

In 2005 Klimke and Wohlmuth developed a robust sparse gird interpolation toolbox for MATLAB (for documentation, see [121]) and published their algorithm for piecewise multilinear hierarchical sparse grid interpolation [122]. In 2006 Yserentant developed an adaptive sparse

grid refinement scheme that takes advantage of symmetry properties of the underlying function, and augmented sparse grid spaces to tackle high-dimensional, antisymmetric functions [233]. A year later, Sickel and Ullrich broadened the class of functions studied for convergence rates of Smolyak's algorithm by studying periodic functions in Sobolev and Besov spaces [192].

In 2008 the stochastic and statistical modeling communities began using sparse grids. Nobile, Timpana and Webster developed a Smolyak sparse grid stochastic collocation method for partial differential equations with random input data [154]. Building off of this work, in 2010 Ma and Zabaras developed a technique using adaptive sparse grid collocation [138] and in 2011 Agarwal and Aluru proposed a weighted Smolyak algorithm [2], both to solve stochastic partial differential equations.

In 2011 Murarasu et al. optimize sparse grid discretization for their use on GPUs for a data imaging application [150], and later present $fastsg$, a library of C routines for the sparse grid technique that optimizes cache use and vectorization to improve algorithm performance on single processors [149]. Recently, Griebel and Harbrecht studied the optimal construction of sparse tensor products for certain spaces, deriving cost complexities to approximate functions with anisotropic and isotropic smoothness on a tensor product domain [87].

While a great deal of work with sparse grids is based on more theoretical aspects, sparse grids have also been used in a wide range of applications. Sparse grids are used most often as a finite element discretization for partial differential equations as in [237], [115] and [1], but other applications are vast. Sparse grids have been used to study: fluid flow [90, 238], quantum mechanics [81], stochastic differential equations and optimization [138, 185, 180], chemistry [103], discrete differential forms [82], boundary integral equations [88], economics and finance [127, 38, 139], time-dependent advection problems [129], data mining [75], colocation methods for uncertainty quantification [228, 227, 39], and Bayesian inverse problems [236, 238, 133] among others.

Before we discuss Smolyak's interpolation algorithm, we first present the concept of sparse grids and their construction as first proposed originally by Zenger in [237]. From this we will gain insights as to why sparse grids are increasingly superior to full grids for increasing dimension d. We begin by outlining the hierarchical multilevel subspace splitting using notation from [37].

2.1.1 Subspace splitting and the curse of dimensionality

In this section we introduce the hierarchical subspace splitting as originally proposed in [237] and draw attention to the curse of dimensionality and how sparse grids offer a more tenable solution. Consider multivariate functions $u(\mathbf{x}) \in \mathbb{R}$ on the d-dimensional unit interval $\bar{\Omega} = [0,1]^d$. We

will denote mixed derivatives as

$$D^{\boldsymbol{\alpha}} u = \frac{\partial^{|\boldsymbol{\alpha}|_1} u}{\partial x_1^{\alpha_1} u \dots \partial x_d^{\alpha_d}}$$

where $\boldsymbol{\alpha} \in \mathbb{N}^d$ is a d-dimensional multi-index with norms $|\boldsymbol{\alpha}|_1 = \sum \alpha_i$ and $|\boldsymbol{\alpha}|_\infty = \max \alpha_i$. For multi-indices $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ and $x \in \mathbb{R}$ we have the following relations and operators:

$$\begin{aligned} x^{\boldsymbol{\alpha}} &= (x^{\alpha_1}, \dots, x^{\alpha_d}), \\ \boldsymbol{\alpha} \leq \boldsymbol{\beta} &\Leftrightarrow \alpha_i \leq \beta_i \; \forall \; 1 \leq i \leq d, \\ \boldsymbol{\alpha} < \boldsymbol{\beta} &\Leftrightarrow \boldsymbol{\alpha} \leq \boldsymbol{\beta} \text{ and } \alpha_i \neq \beta_i \; \forall \; 1 \leq i \leq d. \end{aligned}$$

$X^{p,r}(\bar{\Omega})$ denotes the space of all functions of bounded (with respect to the L_p-norm) mixed derivatives up to order r and $X_0^{p,r}(\bar{\Omega})$ is the subspace of $X^{p,r}(\bar{\Omega})$ that is zero on the boundary of $\bar{\Omega}$. For purposes of simplicity we will only consider $X_0^{p,r}(\bar{\Omega})$.

The family of d-dimensional standard rectangular grids is

$$\{\Omega_{\mathbf{l}} : \mathbf{l} \in \mathbb{N}^d\}$$

where $\mathbf{l}$ is a multi-index that denotes the level of a grid in a multivariate sense. The mesh size is

$$\mathbf{h_l} = (h_{l_1}, \dots, h_{l_d}) = 2^{-\mathbf{l}}, \tag{2.1}$$

where h is the width of the interval. Taking negative powers of 2 cuts each dimension d_i's mesh width in half as we increase the corresponding level l_i by one. Grid points are denoted

$$\mathbf{x_{l,i}} = (x_{l_1,i_1}, \dots, x_{l_d,i_d}) = \mathbf{i} \cdot \mathbf{h_l}$$

for $\mathbf{0} \leq \mathbf{i} \leq \mathbf{2^l}$. Since we are interested in approximating functions in $X_0^{p,r}(\bar{\Omega})$, we must first define discrete approximation spaces. We will use the standard 1-dimensional hat function

$$\phi(x) = \begin{cases} 1 - |x|, & \text{if } x \in [-1, 1] \\ 0, & \text{otherwise} \end{cases} \tag{2.2}$$

and note that $\phi(x)$ can be used to generate an arbitrary hat function with support $[x_{l_j,i_j} - h_{l_j}, x_{l_j,i_j} + h_{l_j}]$ by dilation and translation via

$$\phi_{l_j,i_j}(x_j) = \phi\left(\frac{x_j - i_j h_{l_j}}{h_{l_j}}\right). \tag{2.3}$$

Figure 2.1 shows hat functions for $l = 1$ (solid) and $l = 2$ (dashed) on the interval $[-1,\ 1]$.

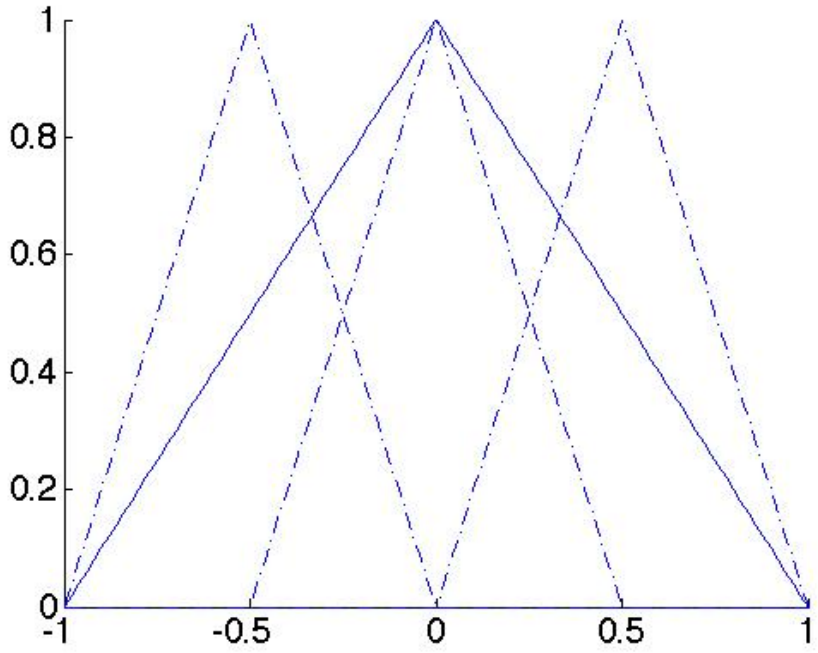


Figure 2.1: Hat functions for $l = 1$ (solid) and $l = 2$ (dashed) on the interval $[-1,\ 1]$.

We build piecewise d-linear basis functions in each grid point $\mathbf{x}_{\mathbf{l},\mathbf{i}}$ by the tensor product construction

$$\phi_{\mathbf{l},\mathbf{i}}(\mathbf{x}) = \prod_{j=1}^{d} \phi_{l_j,i_j}(x_j). \tag{2.4}$$

Since we are only concerned with basis functions that correspond to inner grid points, we define

$$V_{\mathbf{l}} = \text{span}\{\phi_{\mathbf{l},\mathbf{i}} : \mathbf{1} \leq \mathbf{i} \leq \mathbf{2^l} - \mathbf{1}\} \tag{2.5}$$

as the space of piecewise d-linear functions with respect to the interior of $\Omega_{\mathbf{l}}$. Furthermore, we define the hierarchical increments

$$W_{\mathbf{l}} = \text{span}\{\phi_{\mathbf{l},\mathbf{i}} : \mathbf{1} \leq \mathbf{i} \leq \mathbf{2^l} - \mathbf{1},\ i_j \text{ odd for all } 1 \leq j \leq d\} \tag{2.6}$$

and note that

$$V_{\mathbf{l}} = \bigotimes_{\mathbf{k} \le \mathbf{l}} W_{\mathbf{k}}. \tag{2.7}$$

Thus, by defining the index set

$$\mathbf{I}_{\mathbf{l}} = \{\mathbf{i} \in \mathbb{N}^d : \mathbf{1} \le \mathbf{i} \le \mathbf{2}^{\mathbf{l}} - \mathbf{1},\ i_j \text{ odd for all } 1 \le j \le d\} \tag{2.8}$$

we get a second basis of $V_{\mathbf{l}}$, the hierarchical basis

$$\{\phi_{\mathbf{k},\mathbf{i}} : \mathbf{i} \in \mathbf{I}_{\mathbf{k}}, \mathbf{k} \le \mathbf{l}\}. \tag{2.9}$$

Finally, with hierarchical difference spaces $W_{\mathbf{l}}$ we define

$$V = \sum_{l_1=1}^{\infty} \cdots \sum_{l_d=1}^{\infty} W_{(l_1,\ldots,l_d)} = \bigotimes_{\mathbf{l} \in \mathbb{N}^d} W_{\mathbf{l}} \tag{2.10}$$

with its natural hierarchical basis $\{\phi_{\mathbf{l},\mathbf{i}} : \mathbf{i} \in \mathbf{I}_{\mathbf{l}}, \mathbf{l} \in \mathbb{N}^d\}$. An advantage of hierarchical bases is its multilevel structure enables one to distinguish between high-level basis functions with a large support that usually already contain a significant part of the information, and low-level basis functions whose contribution to an interpolant or finite element approximation is comparatively small [35].

Note that in finite-dimensional subspaces of V, e.g.

$$V_n^{(\infty)} = \bigotimes_{|\mathbf{l}|_\infty \le n} W_{\mathbf{l}}, \tag{2.11}$$

any function $u \in X_0^{p,r}(\bar{\Omega})$ can be uniquely split by

$$u(\mathbf{x}) = \sum_{\mathbf{l}} u_{\mathbf{l}}(\mathbf{x}), \quad u_{\mathbf{l}}(\mathbf{x}) = \sum_{\mathbf{i} \in \mathbf{I}_{\mathbf{l}}} v_{\mathbf{l},\mathbf{i}} \cdot \phi_{\mathbf{l},\mathbf{i}}(\mathbf{x}) \in W_{\mathbf{l}} \tag{2.12}$$

where $v_{\mathbf{l},\mathbf{i}}$ are the coefficient values of the hierarchical product basis representation of u. These $v_{\mathbf{l},\mathbf{i}}$ are called the hierarchical surpluses. Note that $V_n^{(\infty)}$ is the space of piecewise d-linear functions on the rectangular grid with equidistant mesh size in each coordinate direction. It is well known that the number of inner grid points for $V_n^{(\infty)}$ is

$$|V_n^{(\infty)}| = (2^n - 1)^d = O(h_n^{-d}) \tag{2.13}$$

and interpolation error for an approximating function $u_n^{(\infty)} \in V_n^{(\infty)}$ is

$$\|u - u_n^{(\infty)}\|_2 \leq O(h_n^2). \tag{2.14}$$

Equation 2.13 reveals the curse of dimensionality that often impedes solving problems in higher dimensions, as the number of nodes required to achieve an accuracy of $O(h_n^2)$ grows exponentially as d increases. In the next section, we discuss the derivation of sparse grids and how they help alleviate this problem.

For sparse grids we are more interested in decompositions of these finite-dimensional approximations spaces of V rather than the splitting given in Equation 2.12. The classical sparse grid construction arises from a cost to benefit analysis that is detailed in [37]. In short, by balancing the number of degrees of freedom involved in a grid (cost) with the square of the upper bounds for interpolation error (benefit), one can solve a constrained optimization problem to solve for underlying index sets $\mathbf{I} \in \mathbb{N}^d$ to form the approximation space $U = \bigotimes_{\mathbf{l} \in \mathbf{I}} W_{\mathbf{l}}$. The nodes corresponding to these spaces U are called sparse grids, and the choice of norm for interpolation error leads to different sparse grids. For example, for L_2 based sparse grids we get the relation

$$|\mathbf{l}|_1 \leq n + d - 1$$

as a qualification for a subspace $W_{\mathbf{l}}$ to be taken into account. Thus, the new discrete approximation space for our sparse grid is

$$V_n^{(1)} = \bigotimes_{|\mathbf{l}|_1 \leq n+d-1} W_{\mathbf{l}}. \tag{2.15}$$

The dimension, or number of degrees of freedom or inner grid points, of $V_n^{(1)}$ is

$$|V_n^{(1)}| = O(h_n^{-1} \cdot |\log_2 h_n|^{d-1}) \tag{2.16}$$

and the L_2 upper bound for the interpolation error of a function u in the sparse grid space $V_n^{(1)}$ is

$$\|u - u_n^{(1)}\|_2 \leq O(h_n^2 \cdot n^{d-1}). \tag{2.17}$$

Proofs of these bounds and other properties of sparse grids can be found in [37]. Observe that the number of degrees of freedom is reduced greatly while the accuracy is only slightly decreased in comparison to $V_n^{(\infty)}$ (see Equations 2.13 and 2.14). Sparse grids only lessen the curse of dimensionality, however, and do not overcome it completely.

Our method uses Lagrange interpolation with Chebyshev nodes in conjunction with an approximation algorithm developed by Smolyak in [198], as first proposed by Barthelmann,

Novak and Ritter [13]. This method is outlined in the following section.

2.2 Smolyak's Algorithm

Smolayk's algorithm uses a linear combination of approximations on sparse grids. To understand the multi-dimensional interpolation algorithm, we first consider the one dimensional interpolation problem where we would like to approximate the value of a function $f : [-1, 1] \to \mathbb{R}$ at some point in the domain. Given a set of $m_i = 2^{i-1} + 1$ nodes $\{x_j\} \in [-1, 1]$ and the corresponding set of function values $\{f(x_j)\}$ we can construct the unique interpolating polynomial of degree $m_i - 1$ denoted

$$U^i[f](x) = \sum_{i=1}^{m_i} f(x_j^i)\ell_j^i(x) \tag{2.18}$$

where $\ell_j^i(x)$ are the Lagrange basis polynomials

$$\ell_j^i(x) = \prod_{k \neq j} \frac{x - x_k^i}{x_j^i - x_k^i}. \tag{2.19}$$

We will refer to i as the *level* of the interpolation. As suggested in [13], we utilize Chebyshev nodes that are of the form

$$x_j^i = -\cos\frac{\pi(j-1)}{m_i}, \quad 1 \leq j \leq m_i \tag{2.20}$$

with $x_1^i = 0$ if $m_i = 1$;. Note that our choice of m_i leads to sets of nodes that are nested as we increase in level, i.e. if χ^i is the set of nodes for level i we have $\chi^i \subset \chi^{i+1}$. For example, the set of nodes for the $i = 2$ is

$$\chi^2 = \{-1, 0, 1\},$$

and the set of nodes for $i = 3$ is

$$\chi^3 = \left\{-1, -\frac{\sqrt{2}}{2}, 0, \frac{\sqrt{2}}{2}, 1\right\} = \chi^2 \cup \left\{-\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}\right\}.$$

The Lagrange basis polynomials for level $i = 3$ are shown in Figure 2.2.

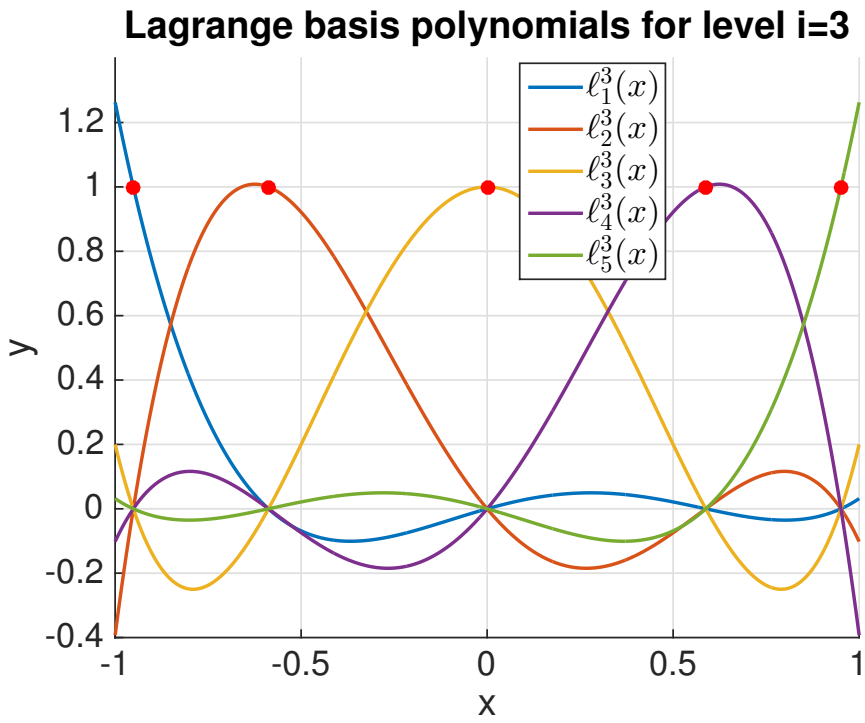


Figure 2.2: The Lagrange basis polynomials on for interpolation level $i = 3$ on the domain $[-1, 1]$.

Expanding this idea to $d > 1$ dimensions, we now let $f : \mathbb{R}^d \to \mathbb{R}$. We will use the standard multi-index notation

$$\mathbf{i} = (i_1, \ldots, i_d) \text{ and } |\mathbf{i}| = \sum_{j=1}^{d} i_j.$$

For $x \in \mathbb{R}^d$ and multi-index $\mathbf{i}$ we define the d-dimensional Lagrange polynomial by a tensor product of one-dimensional Lagrange polynomials:

$$U^{\mathbf{i}}[f](x) = \bigotimes_{r=1}^{d} U^{i_r}[f](x) \tag{2.21}$$

$$= \sum_{j_1=1}^{m_{i_1}} \cdots \sum_{j_d=1}^{m_{i_d}} f(x_{j_1}^{i_1}, \ldots, x_{j_d}^{i_d}) \prod_{r=1}^{d} \ell_{j_r}^{i_r}(x_r). \tag{2.22}$$

This tensor product has a poor order of convergence, but serves as the foundation for the more complicated algorithm of Smolyak [155].

Example 2.2.0.1. *Let $d = 3$ and $\mathbf{i} = (2, 3, 2)$ and consider a function $f : [-1, 1]^3 \to \mathbb{R}$. Our choice of $\mathbf{i}$ says we want to take the tensor product of three one-dimensional Lagrange polynomials. The Lagrange polynomials in dimensions 1 and 3 are of interpolation level 2 and the Lagrange polynomial in dimension 2 is of interpolation level 3. Evaluating this tensor product at $x = (x_1, x_2, x_3)$, Equation 2.22 becomes*

$$U^{\mathbf{i}}[f](x) = \sum_{j_1=1}^{3} \sum_{j_d=1}^{5} \sum_{j_d=1}^{3} f(x_{j_1}^2, x_{j_d}^3, x_{j_d}^2) \prod_{r=1}^{d} \ell_{j_r}^{i_r}(x_r).$$

Given a degree of exactness k, we define $q = d + k$. For Smolyak's algorithm we define the set of allowable multi-indices $\mathbf{i}$ by

$$Q(q, d) = \{\, \mathbf{i} \in \mathbb{N}^d |\, k + 1 \leq |\mathbf{i}| \leq q\}. \tag{2.23}$$

Each multi-index $\mathbf{i} \in Q(q, d)$ contains the levels of each dimension's interpolation and can be thought of as representing a different sparse grid on which we must approximate the function. Smolyak's algorithm [198] uses linear combinations of Equation 2.22 on different sparse grids to approximate the multivariate function f.

Smolyak's formula is originally given in [198] by the operator

$$\mathcal{A}(q, d) = \sum_{|\mathbf{i}| \leq q} (\Delta^{i_1} \otimes \ldots \otimes \Delta^{i_d}) \tag{2.24}$$

with

$$\Delta^i = U^i - U^{i-1}$$

for $i \in \mathbb{N}$ and $U^0 = 0$. In [222] Wasilkowski and Wozniakowski showed that Equation 2.24 is equivalent to

$$\mathcal{A}(q, d) = \sum_{\mathbf{i} \in Q(q,d)} (-1)^{q-|\mathbf{i}|} \begin{pmatrix} d-1 \\ q-|\mathbf{i}| \end{pmatrix} U^i, \tag{2.25}$$

which we will prove in the following theorem.

Theorem 2.2.0.1. *Equations 2.24 and 2.25 are equivalent.*

Proof. First define the set

$$P(q, d) = \{\mathbf{i} = [i_1, \ldots, i_d] : \mathbf{1} \leq \mathbf{i}, |\mathbf{i}| \leq q\}.$$

The cardinality of $P(q, d)$ is $\begin{pmatrix} q \\ d \end{pmatrix}$ [222] and contains all the indices used in Equation 2.24.

The restriction of $\mathbf{1} \leq \mathbf{i}$ is included since we have $\Delta^0 = 0$, so any $\mathbf{i}$ containing an element that is 0 will not be included in the sum in Equation 2.24.

Now we have

$$\mathcal{A}(q,d) = \sum_{\mathbf{i} \in P(q,d)} \bigotimes_{r=1}^{d} \Delta^{i_r} = \sum_{\mathbf{i} \in P(q-1,d-1)} \left(\bigotimes_{r=1}^{d-1} \Delta^{i_r} \right) \otimes \sum_{i_d=1}^{q-|\mathbf{i}|} \Delta^{i_d} \tag{2.26}$$

$$= \sum_{\mathbf{i} \in P(q-1,d-1)} \left(\bigotimes_{r=1}^{d-1} \Delta^{i_r} \right) \otimes U^{q-|\mathbf{i}|} \tag{2.27}$$

since we obtain a telescoping series

$$\sum_{j=1}^{m} \Delta^j = U^m. \tag{2.28}$$

Now observe that

$$\bigotimes_{r=1}^{d} \Delta^{i_r} = \sum_{\alpha \in \{0,1\}^d} (-1)^{|\alpha|} \bigotimes_{r=1}^{d} U^{i_r - \alpha_r}$$

and $\bigotimes_{r=1}^{d} U^{j_r}$ appears in $\mathcal{A}(q,d)$ for all indices $\mathbf{i}$ such that $i_r = j_r + \alpha_r$ with $\alpha \in \{0,1\}^d$ and $|\alpha| \leq q - |\mathbf{j}|$. With these observations we can explicitly derive a form of $\mathcal{A}(q,d)$ for all indices $\mathbf{i}$ such that $i_r = j_r + \alpha_r$ where $\alpha \in \{0,1\}^d$ and $|\alpha| \leq q - |\mathbf{j}|$. First observe that the sign of $\bigotimes_{r=1}^{d} U^{j_r}$ is $(-1)^{|\alpha|}$. Defining

$$b(z,d) = \sum_{\alpha \in \{0,1\}^d, |\alpha| \leq z} (-1)^{|\alpha|}, \tag{2.29}$$

Equation 2.27 now reads

$$\mathcal{A}(q,d) = \sum_{\mathbf{j} \in P(q,d)} b(q - |\mathbf{j}|, d) \bigotimes_{r=1}^{d} U^{j_r}. \tag{2.30}$$

To compute $b(z,d)$, note that we can sum with respect to $|\alpha| = 0, 1, \ldots, d$. Since $|\alpha| = j$ corresponds to $\begin{pmatrix} d \\ j \end{pmatrix}$ terms, we have

$$b(z,d) = \sum_{j=0}^{\min\{z,d\}} \begin{pmatrix} d \\ j \end{pmatrix} (-1)^j = (-1)^z \begin{pmatrix} d-1 \\ z \end{pmatrix}. \tag{2.31}$$

In particular, $b(z,d) = 0$ for $z \geq d$. Thus we have

$$\mathcal{A}(q,d) = \sum_{\mathbf{i} \in Q(q,d)} (-1)^{q-|\mathbf{i}|} \begin{pmatrix} d-1 \\ q-|\mathbf{i}| \end{pmatrix} U^i \tag{2.32}$$

and the equivalence is shown. □

Example 2.2.0.1. *Let $d = 2$ and $k = 2$, so $q = 4$. The set of allowable multi-indices is*

$$Q(4,2) = \{(1,1),(1,2),(1,3),(2,2),(3,1),(2,1)\}.$$

Since 3 is the largest element of any $\mathbf{i} \in Q(4,2)$, our nested sets of nodes are

$$\begin{aligned} \chi^1 &= \{0\}, \\ \chi^2 &= \{-1,0,1\}, \\ \text{and } \chi^3 &= \left\{-1, -\frac{\sqrt{2}}{2}, 0, \frac{\sqrt{2}}{2}, 1\right\}. \end{aligned}$$

Each $\mathbf{i} \in Q(4,2)$ defines a set of points via the cartesian product of the two corresponding sets of nodes. For example, for $\mathbf{i} = (2,2)$ we have

$$\chi^2 \times \chi^2 = \{(-1,-1),(-1,0),(-1,1),(0,-1),(0,0),(0,1),(1,-1),(1,0),(1,1)\},$$

which are shown in Figure 2.3 along with the entire set of points used by Smolyak's algorithm.

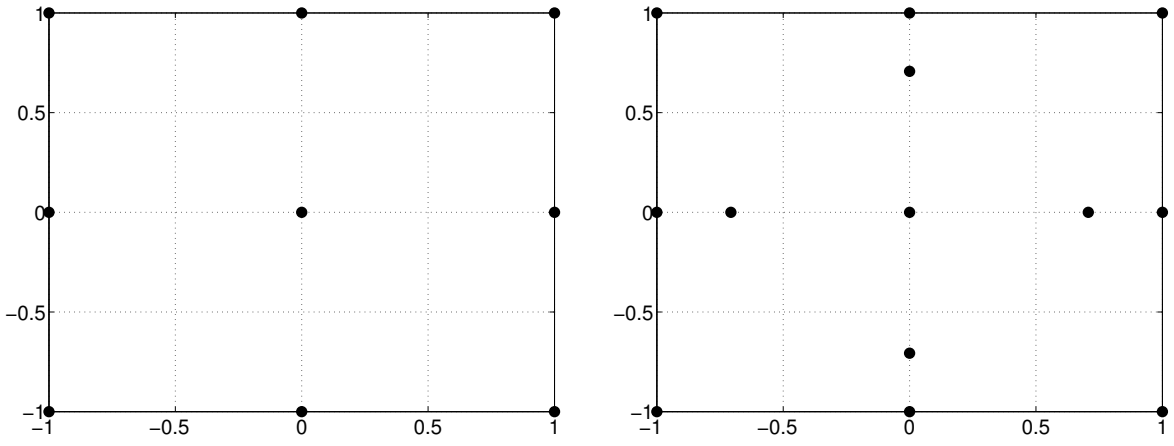


Figure 2.3: Left: sparse grid points for $\mathbf{i} = (2,2)$. Right: all sparse grid points for $d = k = 2$.

2.2.1 Complexity

To evaluate the Smolyak interpolating polynomial, one only needs to know function values at the sparse grid nodes

$$\mathcal{H}(q,d) = \bigcup_{k+1 \le |\mathbf{i}| \le q} (\chi^{i_1} \times \ldots \times \chi^{i_d}), \tag{2.33}$$

where $\chi^i = \{x_1^i, \ldots, x_{m_i}^i\}$ is the set of points used by the interpolant U^i. Since our sets of nodes are nested we also have $\mathcal{H}(q,d) \subset \mathcal{H}(q+1,d)$. Figure 2.4 shows the sparse grids $\mathcal{H}(7,2)$ and $\mathcal{H}(8,3)$.

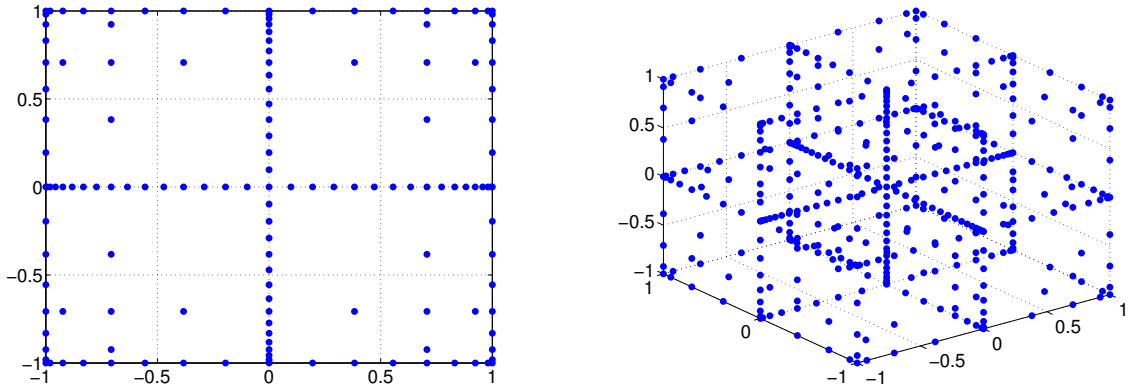


Figure 2.4: Examples of sparse grids for $\mathcal{H}(7,2)$ (left) and $\mathcal{H}(8,3)$ (right) on the domain $[-1,1]^d$. Notice that the nodes are well-dispersed throughout the domain.

Given a specified error tolerance, an upper bound for the number of nodes required for an approximation for a solution was first developed in [222] by Wasilkowski and Wozniakowski. In [164], Petras studied the asymptotically minimal number of nodes to obtain a cubature formula with a given degree of exactness. Here, however, we derive an estimate for the number of grid points, $n(k,d)$, required for a given degree of exactness k for interpolation as in [156]. We use $\approx$ to denote strong equivalence of sequences. That is,

$$v_n \approx w_n \Leftrightarrow \lim_{n\to\infty} \frac{v_n}{w_n} = 1.$$

The next theorem characterizes how $n(k,d)$ grows as we increase our dimension d.

Theorem 2.2.1.1. *For some fixed value of k and as $d \to \infty$,*

$$n(k,d) \approx \frac{2^k d^k}{k!}, \tag{2.34}$$

thus the number of nodes for Smolyak's algorithm grows polynomially in d. In this sense, Smolyak's algorithm helps alleviate the curse of dimensionality.

Proof. We will denote the number of nodes in a set as $n(\chi)$. Since $\chi_j^0 = \emptyset$ and $n(\chi_j^2 \setminus \chi_j^1) = 2$, we have

$$\bigcup_{|\mathbf{i}|=q,\, i_r \leq 2} (\chi_1^{i_1} \setminus \chi_1^{i_1-1}) \times \ldots \times (\chi_d^{i_d} \setminus \chi_d^{i_d-1}). \tag{2.35}$$

Assuming $k \leq d$ gives us a lower bound on the number of nodes

$$n(k,d) \geq \left(\begin{array}{c} d \\ k \end{array} \right) \cdot 2^k. \tag{2.36}$$

Now, if $x \in \mathcal{H}(q,d)$ then there may only be up to k coordinates of x which are not members of the respective χ_j^1. Let $J = \{j_1, \ldots, j_v\}$ be a set of directions so that $\{x_j \notin \chi_j^1 | \forall\ j \in J\}$. Assume that $x \in \mathcal{H}(q,d)$ with $x_j \notin \chi_j^1$ if and only if $j \in J$. If $v = k$, then

$$x_j \in \chi_j^2 \setminus \chi_j^1\ \forall j \in J$$

and therefore

$$n(\{x \in \mathcal{H}(q,d) | x_j \notin \chi_j^1 \text{ iff } j \in J\}) \leq 2^k.$$

If $v < k$ then

$$x_j \in \bigcup_{i=1}^{k} \chi_j^i\ \forall\ j \in J$$

and therefore, recalling that m_i is the number of nodes needed at the i^{th} level in one dimension,

$$n(\{x \in \mathcal{H}(q,d) | x_j \notin \chi_j^1 \text{ iff } j \in J\}) \leq \left(\sum_{i=1}^{k} m_i \right)^k = c_k.$$

Finally, we have

$$n(k,d) \leq \sum_{v=0}^{k-1} \left(\begin{array}{c} d \\ v \end{array} \right) \cdot c_k + \left(\begin{array}{c} d \\ k \end{array} \right) \cdot 2^k \leq k \cdot d^{k-1} \cdot c_k + \left(\begin{array}{c} d \\ k \end{array} \right) \cdot 2^k. \tag{2.37}$$

Equations 2.36 and 2.37 give us the desired result. □

To illustrate this drastic reduction in grid points, Table 2.1 compares the number of grid points needed for tensor product interpolation with 5 points in each dimension and Smolyak's sparse grid interpolation.

Table 2.1: The number of grid points for tensor product grids with 5 points in each dimension and for Smolyak sparse grids.

d	Tensor Product	Sparse Grid		
		$k=1$	$k=2$	$k=3$
1	5	3	5	9
2	25	5	13	29
5	3,125	11	61	241
10	9,765,625	21	221	1,581
15	30,517,578,125	31	481	5,021

2.2.2 Error

General error and cost bounds for Smolyak's algorithm were derived by Smolyak in [198] and more explicitly by Wasilkowski and Wozniakowski in [222]. In [199] Sprengel derived error estimates for interpolation on Gauss-Chebyshev grids, but restricted to functions from certain kinds of Besov-type spaces. Here we prove error bounds as in [13] for functions with continuous mixed derivatives. First, however, we state and prove one of the most important properties of Smolyak's interpolation algorithm: polynomial exactness.

Exactness

In one dimension Lagrange interpolation with $k+1$ distinct nodes will exactly interpolate polynomials of degree k. There is an analogous result for Smolyak interpolation, which is stated in the following theorem. The proof is similar to the one from [155] where the authors proved the result for quadrature formulas. We will denote the space of polynomials in one variable of degree m or less as $\mathbb{P}_m$.

Theorem 2.2.2.1. *$\mathcal{A}(q,d)(f)$ will exactly reproduce all polynomials of the form*

$$\sum_{|\mathbf{i}|=q} \left(\mathbb{P}_{m_{i_1}-1} \otimes \ldots \otimes \mathbb{P}_{m_{i_d}-1} \right)$$

where $\mathbb{P}_m$ is the space of one-dimensional polynomials of degree less than or equal to m.

Proof. As in [155] we will proceed by induction on d. For $d = 1$, $\mathcal{A}(q, 1) = U^q$ which is exact. Assume $d > 1$ and the function f is a product of univariate polynomials

$$f = f_{i_1} \otimes \ldots \otimes f_{i_{d+1}} \tag{2.38}$$

where $|\mathbf{i}| = q$ and $i_1 + \ldots + i_d = m$ (note that we have $m + i_{d+1} = q$). We will use the fact that $\mathcal{A}(q, d+1)$ can be written in terms of $\mathcal{A}(\ell, d)$ by

$$\begin{aligned} \mathcal{A}(q, d+1) &= \sum_{\ell=d}^{q-1} \mathcal{A}(\ell, d) \otimes (U^{q-\ell} - U^{q-\ell-1}) \\ \Rightarrow \mathcal{A}(q, d+1)f &= \sum_{\ell=d}^{q-1} \mathcal{A}(\ell, d)(f_{i_1} \otimes \ldots \otimes f_{i_d}) \cdot (U^{q-\ell} - U^{q-\ell-1}) f_{i_{d+1}}. \end{aligned}$$

Since

$$U^{q-\ell} f_{i_{d+1}} = U^{q-\ell-1} f_{i_{d+1}} = f_{i_{d+1}}$$

for $d \leq \ell \leq m-1$ and from our inductive hypothesis we have that

$$\mathcal{A}(\ell, d)(f_{i_1} \otimes \ldots \otimes f_{i_d}) = f_{i_1} \otimes \ldots \otimes f_{i_d}$$

for $m \leq \ell \leq q-1$, we obtain

$$\begin{aligned} \mathcal{A}(q, d+1)f &= \sum_{\ell=m}^{q-1} f_{i_1} \otimes \ldots \otimes f_{i_d} \cdot (U^{q-\ell} - U^{q-\ell-1}) f_{i_{d+1}} \\ &= f_{i_1} \otimes \ldots \otimes f_{i_d} \cdot U^{q-m} f_{i_{d+1}} \\ &= f_{i_1} \otimes \ldots \otimes f_{i_{d+1}} = f. \end{aligned}$$

Thus $\mathcal{A}(d+k, d)$ is exact for all polynomials of degree less than or equal to k. □

Barthelmann et al. also derive error bounds for Smolyak's algorithm in [13] and we summarize their work here. Starting with $d = 1$, the interpolation operator U^i is exact on $\mathbb{P}(m_i - 1)$, the space of polynomials with degree at most $m_i - 1$. They apply the general error bound formula

$$\|f - U^i(f)\|_\infty \leq E_{m_i - 1}(f) \cdot (1 + \Lambda_{m_i}) \tag{2.39}$$

where E_m is the error of the best approximation by polynomials $p \in \mathbb{P}(m)$ and Λ_{m_i} is the

Lebesgue constant for Chebyshev nodes defined as

$$\Lambda_{m_i} = \max_{x\in[-1,1]} \left[\sum_{j=1}^{m_i} |\ell_j^i(x)| \right] \tag{2.40}$$

where ℓ_j^i is defined in Equation 2.19. From [110, 62] we have that

$$\Lambda_{m_i} \leq \frac{2}{\pi} \log(m_i - 1) + 1 \tag{2.41}$$

for $m \geq 2$.

Consider the space

$$F_1^k = C^k([-1,1])$$

with the norm

$$\|f\| = \max \left\{ \|D^\alpha f\|_\infty \,|\alpha = 0, \ldots, k \right\}$$

for $d = 1$ and the space

$$F_d^k = \left\{ f : [-1,1]^d \to \mathbb{R} | D^\alpha f \text{ continuous if } \alpha_i \leq k \;\forall\; i \right\}$$

with norm

$$\|f\| = \max \left\{ \|D^\alpha f\|_\infty \,|\alpha \in \mathbb{N}_0^d,\ \alpha_i \leq k \right\}$$

for $d > 1$. Finite linear combinations of functions $g = \bigotimes_{i=1}^{d} f_i$ with $f_i \in F_1^k$ are dense in F_d^k and

$$\|g\| = \prod_{i=1}^{d} \|f_i\|.$$

We will let I_d denote the embedding $F_d^k \hookrightarrow C([-1,1]^d)$ and

$$\|S\| = \sup \left\{ \|S(f)\|_\infty \,|f \in F_d^k,\ \|f\| \leq 1 \right\}$$

for $S : F_d^k \to C([-1,1]^d)$. In the following we will use $c_{d,k}$ to denote constants that only depend on d and k. Equations 2.39, 2.41, and the well-known Jackson estimate

$$E_n(f) \leq c_{1,k} \cdot \|f\| \cdot n^{-k}$$

give us the error bound

$$\|I_1 - U^i\| \leq \hat{c}_{1,k} \cdot (\log m_i) \cdot m_i^{-k} \tag{2.42}$$

for any $f \in F_1^k$ and $i > 1$. This is an optimal bound for every k up to the logarithmic factor. Barthelmann, Novak and Ritter use this one-dimensional estimate to prove error bounds for higher dimensions. We state and prove the theorem as in [13].

Theorem 2.2.2.1. *Let $n = n(q,d)$ be the number of nodes required by $\mathcal{A}(q,d)$. Then for the space F_d^k,*

$$\|I_d - \mathcal{A}(q,d)\| \leq c_{d,k} n^{-k} (\log n)^{(k+2)(d-1)+1}. \tag{2.43}$$

Proof. Let $D = 2^{-k}$. First, note that $\|I_1\| = 1$. For $i > 1$, Equation 2.42 yields

$$\begin{aligned}
\|I_1 - U^i\| &\leq \hat{c}_{1,k} \cdot (\log m_i) \cdot m_i^{-k} \\
&= \hat{c}_{1,k} \cdot (\log(2^{i-1}+1)) \cdot (2^{i-1}+1)^{-k} \\
&\leq \hat{c}_{1,k} \cdot \log(2^i) \cdot (2^{i-1})^{-k} \\
&= \hat{c}_{1,k} \cdot \frac{\log_2(2^i)}{\log_2(e)} \cdot 2^{k(1-i)} \\
&= CiD^i
\end{aligned}$$

where $C = \hat{c}_{1,k} \cdot \frac{2^k}{\log_2(e)} > 0$ and $D = 2^{-k}$. Similarly,

$$\left\|\Delta^i\right\| \leq EiD^i$$

for some $E > 0$. Defining

$$p(s,j) = \begin{cases} 1, & \text{if } j = 0 \\ \sum_{\mathbf{i} \in \mathbb{N}^j, |\mathbf{i}| = s} \prod_{\nu=1}^{j} i_\nu & \text{otherwise} \end{cases}$$

for $j \in \mathbb{N}_0$ and $s \geq j$, we claim that for every $q \geq d$ (recall $q = d + k$)

$$\|I_d - \mathcal{A}(q,d)\| \leq CD^{k+1} \sum_{j=0}^{d-1} (ED)^j \sum_{s=j}^{k+j} (k+j+1-s) p(s,j).$$

This estimate is proved in [13] but omitted here.

Let $B = \max\left\{E, D^{-1}\right\}$. Now we have

$$\begin{aligned}\|I_d - \mathcal{A}(q,d)\| &\leq CD^{k+1}\sum_{j=0}^{d-1}\max(1,ED)^{d-1}\sum_{s=j}^{k+j}(k+j+1-s)p(s,j)\\ &\leq CB^{d-1}D^q\sum_{j=0}^{d-1}\sum_{s=j}^{k+j}(k+j+1-s)p(s,j).\end{aligned}$$

If we let

$$\Gamma = \sum_{j=1}^{d-1}\sum_{s=j}^{k+j}(k+j+1-s)p(s,j),$$

then

$$\begin{aligned}\sum_{j=0}^{d-1}\sum_{s=j}^{k+j}(k+j+1-s)p(s,j) &= \Gamma + \sum_{s=0}^{k}(k+1-s)\\ &= \Gamma + \frac{1}{2}(k+1)(k+2)\end{aligned}$$

and

$$\begin{aligned}\Gamma &= \sum_{j=1}^{d-1}\sum_{s=j}^{k+j}(k+j+1-s)\sum_{\mathbf{i}\in\mathbb{N}^j,|\mathbf{i}|=s}\prod_{\nu=1}^{j}i_\nu\\ &\leq \sum_{j=1}^{d-1}\sum_{s=j}^{k+j}(k+j+1-s)\left(\begin{array}{c}s-1\\ j-1\end{array}\right)\left(\frac{s}{j}\right)^j\\ &\leq \sum_{j=1}^{d-1}\sum_{s=j}^{k+j}(k+j+1)\left(\begin{array}{c}q-2\\ j-1\end{array}\right)(q-1)^{d-1}\\ &\leq c_d\cdot q^{2d-1}.\end{aligned}$$

We now have

$$\|I_d - \mathcal{A}(q,d)\| \leq c_{d,k}\cdot 2^{-kq}q^{2d-1},$$

but we wish to relate this bound to the number of nodes, n. From [155], n can be bounded above by

$$n \leq c_d\cdot q^{d-1}\cdot 2^q$$

if $m_i = 2^{i-1} + 1$, ensuring the nodes are nested with increasing level i. Thus,

$$\begin{aligned} \|I_d - \mathcal{A}(q,d)\| &\leq c_{d,k} \cdot (n/q^{d-1})^{-k} q^{2d-1} \\ &\leq c_{d,k} \cdot n^{-k} q^{(k+2)(d-1)+1} \\ &\leq c_{d,k} \cdot n^{-k} (\log n)^{(k+2)(d-1)+1} \end{aligned}$$

and the theorem is proved. □

Error Estimation

The error of Smolyak's interpolating polynomial can be estimated using the same approach as Runge-Kutta methods for ordinary differential equations. Runge-Kutta methods estimate the error of an approximate solution by comparing the current order's approximation to a higher order approximation [173]. Similarly, the error of Smolyak's interpolant with polynomial exactness k can be estimated with Smolyak's interpolant with polynomial exactness $k-1$. Since sparse grids are nested one only needs to compare the difference of the two interpolants at the nodes unique to the order k interpolant, i.e. $x \in \mathcal{H}(d+k,d) \setminus \mathcal{H}(d+k-1,d)$. In the sparse grid literature these differences are known as *hierarchical surpluses* and are used for error estimation in adaptive sparse grid regimes [38, 35, 123]. Since the function we approximate in this work is far too expensive for adaptive interpolation algorithms, we use the hierarchical surpluses as a posterior error estimate.

Consider interpolating the function $f : \mathbb{R}^d \to \mathbb{R}$ with polynomial exactness k. Recalling that $q = d + k$, define the set $D = \mathcal{H}(q,d) \setminus \mathcal{H}(q-1,d)$ and let $\hat{f}^q(x)$ denote Smolyak's interpolating polynomial $\mathcal{A}(q,d)[f](x)$. Our relative interpolation error estimate is

$$\epsilon_{rel} = \max_{x \in D} \frac{\|\hat{f}^q(x) - \hat{f}^{q-1}(x)\|}{\|\hat{f}^q(x)\|}. \tag{2.44}$$

Note that if we have already computed $\hat{f}^q(x)$ the nested structure of Smolyak's sparse grids allows us to easily compute $\hat{f}^{q-1}(x)$. Also, for $x \in D$ $\hat{f}^q(x) = f(x)$, so the numerator of Equation 2.44 is indeed the absolute error for the interpolant of order $k-1$.

2.3 Efficient Implementation

In this section we present a new implementation of the sparse grid polynomial interpolation algorithm that is based on a reformulation of Smolyak's algorithm by Judd and coworkers in [118]. To our knowledge, previous research has focused on efficient algorithms for evaluating Equation 2.25 simultaneously at one or very few points [122, 149, 150]. Our goal, however, is

to evaluate the interpolant and its gradient at several thousand points simultaneously, and an algorithm that is efficient for, say, 5 points may not be efficient for 10^5 points depending on the algorithm's scalability.

As noted by Judd and coworkers in [118], Smolyak's algorithm as written in Equation 2.25 is inefficient because the linear combination causes several basis functions to be evaluated more than once. As we will see, the reformulation of Smolyak's algorithm eliminates redundant calculations of basis functions by using disjoint set generators instead of the conventional nested set generators for the Smolyak grids and basis functions. The reformulation from [118] is motivated by derivative-free dynamic economic models, where Smolyak's interpolating polynomial is repeatedly evaluated at a large number of points. On the other hand, our application requires evaluating the interpolant and its gradient at a large number of points simultaneously. As such, we extend the work of Judd and coworkers in two ways: first, our implementation is capable of evaluating the analytical gradient of Smolyak's interpolating polynomial, and second, the implementation is designed to quickly evaluate the interpolating polynomial and its gradient at a large number of points simultaneously. In the rest of this section we will review the implementation from [118] and describe our improvements.

2.3.1 Reformulation of Smolyak's Algorithm

Smolyak's algorithm as written in Equation 2.25 is inefficient in that the linear combination causes several basis functions to be evaluated more than once. Judd and coworkers devised a method in [118] to avoid repeated calculations by using disjoint sets of nodes to generate the unidimensional basis functions instead of the nested sets used in the conventional algorithm. In this section we summarize the reformulation from [118] and compare it to Smolyak's original formulation.

As an example, consider interpolating a function on $[-1,1]^2$ with $k=2$. Using the conventional Smolyak method, the nested sets of unidimensional nodes are

$$\begin{aligned} \chi^1 &= \{0\}, \\ \chi^2 &= \{-1,0,1\}, \\ \text{and } \chi^3 &= \left\{-1,-\frac{\sqrt{2}}{2},0,\frac{\sqrt{2}}{2},1\right\}. \end{aligned}$$

With Judd et al.'s disjoint sets, redundancies are removed and our sets of unidimensional nodes

become

$$\begin{aligned} \chi^1 &= \{0\}, \\ \chi^2 &= \{-1, 1\}, \\ \text{and } \chi^3 &= \left\{-\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}\right\}. \end{aligned}$$

The size of χ^n is now $m_n - m_{n-1} = 2^{n-2}$ for $n \geq 3$ where m_n is defined as before to be $m_n = 2^{n-1} + 1$. The size of χ^2 and χ^1 is 2 and 1, respectively. Table 2.2 shows the appropriate disjoint sparse grid points for each level i for Smolyak's algorithm with $d = k = 2$. The set of sparse grid nodes corresponding to any multi index $\mathbf{i}$ can be obtained by taking the union of the sets in the corresponding row and column in Table 2.2. Note that the sparse grids themselves are still nested; it is the sets that generate the sparse grids that are not.

Table 2.2: Sets of disjoint grid nodes corresponding to each level of sparse grid.

	$i_2 = 1$	$i_2 = 2$	$i_2 = 3$
$i_1 = 1$	$(0, 0)$	$(0, -1), (0, 1)$	$\left(0, -\frac{\sqrt{2}}{2}\right), \left(0, \frac{\sqrt{2}}{2}\right)$
$i_1 = 2$	$(-1, 0), (1, 0)$	$(-1, -1), (-1, 1),$ $(1, -1), (1, 1)$	$\left(-1, -\frac{\sqrt{2}}{2}\right), \left(-1, \frac{\sqrt{2}}{2}\right),$ $\left(1, -\frac{\sqrt{2}}{2}\right), \left(1, \frac{\sqrt{2}}{2}\right)$
$i_1 = 3$	$\left(-\frac{\sqrt{2}}{2}, 0\right), \left(\frac{\sqrt{2}}{2}, 0\right)$	$\left(-\frac{\sqrt{2}}{2}, -1\right), \left(\frac{\sqrt{2}}{2}, -1\right),$ $\left(-\frac{\sqrt{2}}{2}, 1\right), \left(\frac{\sqrt{2}}{2}, 1\right)$	$\left(-\frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2}\right), \left(-\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}\right),$ $\left(\frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2}\right), \left(\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}\right)$

The unidimensional basis functions $\phi(x)$ can be organized in a similar way. Following notation from [118] we let A_i be the set containing the appropriate unidimensional basis functions for interpolation level i. This yields

$$\begin{aligned} A_1 &= \{1\} \\ A_2 &= \{\phi_2(x), \phi_3(x)\} \\ A_3 &= \{\phi_4(x), \phi_5(x)\} \\ &\vdots \end{aligned}$$

where each $\phi_j(x)$ is a Lagrange basis polynomial constructed in the usual way utilizing all m_i

nodes of the level i to which it belongs. For example, $\phi_4(x)$ is the first basis function for level $i = 3$ and corresponds to the Lagrange basis polynomial centered at $x_2 = -\frac{\sqrt{2}}{2}$. So, recalling notation from Equations 2.19 and 2.20, we have

$$\phi_4(x) = \ell_2^3(x) = \prod_{j=1,\, j\neq 2}^{5} \frac{x - x_j^3}{x_2^3 - x_j^3}. \tag{2.45}$$

If we let $(x, y) \in [-1, 1]$, Table 2.3 contains the disjoint multidimensional tensor product basis functions for $d = k = 2$.

Table 2.3: Sets of disjoint basis functions corresponding to each level of sparse grid.

	$i_2 = 1$	$i_2 = 2$	$i_2 = 3$
$i_1 = 1$	1	$\phi_2(y),\ \phi_3(y)$	$\phi_4(y),\ \phi_5(y)$
$i_1 = 2$	$\phi_2(x),\ \phi_3(x)$	$\phi_2(x)\phi_2(y),\ \phi_2(x)\phi_3(y),$ $\phi_3(x)\phi_2(y),\ \phi_3(x)\phi_3(y)$	$\phi_2(x)\phi_4(y),\ \phi_2(x)\phi_5(y),$ $\phi_3(x)\phi_4(y),\ \phi_3(x)\phi_5(y)$
$i_1 = 3$	$\phi_4(x),\ \phi_5(x)$	$\phi_4(x)\phi_2(y),\ \phi_4(x)\phi_3(y),$ $\phi_5(x)\phi_2(y),\ \phi_5(x)\phi_3(y)$	$\phi_4(x)\phi_4(y),\ \phi_4(x)\phi_5(y),$ $\phi_5(x)\phi_4(y),\ \phi_5(x)\phi_5(y)$

With these unidimensional basis sets in hand, Smolyak's algorithm takes on a new form. First, for $x \in [-1, 1]^d$ and multi index $\mathbf{i} \in \mathbb{N}^d$, we define the tensor product

$$\tilde{U}^{\mathbf{i}}[f](x) = \sum_{j_1 = m_{i_1 - 1} + 1}^{m_{i_1}} \cdots \sum_{j_d = m_{i_d - 1} + 1}^{m_{i_d}} b_{j_1 \ldots j_d} \phi_{j_1}(x_1) \ldots \phi_{j_d}(x_d) \tag{2.46}$$

analogous to Equation 2.22. Next, we define a sum of these tensor products for a single index $i \in \mathbb{N}$ as

$$\tilde{U}^{i}[f](x) = \sum_{\alpha \in \mathbb{N}^d \,|\, |\alpha| = i} \tilde{U}^{\alpha}[f](x).$$

The coefficients $b_{j_1 \ldots j_d}$ in Equation 2.46 are found by solving the M by M system of linear equations

$$\begin{bmatrix} \Phi_1(x_1) & \ldots & \Phi_M(x_1) \\ \vdots & \ddots & \vdots \\ \Phi_1(x_M) & \ldots & \Phi_M(x_M) \end{bmatrix} \begin{bmatrix} b_1 \\ \vdots \\ b_M \end{bmatrix} = \begin{bmatrix} f(x_1) \\ \vdots \\ f(x_M) \end{bmatrix} \tag{2.47}$$

where M is the total number of sparse grid points, $f(x)$ is the function we are interpolating, and each $\Phi_i(x)$ is a product of the disjoint basis functions ϕ_{i_r}. Now Smolyak's algorithm operator

can efficiently be written as

$$\mathcal{A}(q,d) = \sum_{d \le i \le q} \tilde{U}^i. \tag{2.48}$$

In this form each basis function $\Phi_i(x)$ appears only once and thus leads to large savings in computation time. Note that this operator (Equation 2.48) and Smolyak's original operator (Equation 2.25) are equivalent; the reformulation only regroups terms to avoid redundant calculations of the tensor product in Equation 2.22. Also, the coefficients $b_{j_1 \dots j_d}$ need only be solved for once and can be stored for repeated evaluation of the interpolant.

Now we quantify the computational savings of this reformulation by comparing the number of terms in each from of Smolyak's algorithm. From [118], the number of terms evaluated in the original Smolyak formula (Equation 2.25) is

$$N_S(q,d) = \sum_{\max(d,k+1) \le |\mathbf{i}| \le q} \left[\prod_{j=1}^{d} m_{i_j} \right],$$

and the number of terms evaluated in the new Smolyak formula (Equation 2.48) is

$$N_J(q,d) = \sum_{d \le |\mathbf{i}| \le q} \left[\prod_{j=1}^{d} [m_{i_j} - m_{i_j - 1}] \right].$$

Note that $N_J(q,d) = \#\mathcal{H}(q,d)$ where $\#S$ denotes the cardinality of a set S. Defining

$$R(q,d) = \frac{N_S(q,d)}{N_J(q,d)},$$

Figure 2.5 shows $R(q,d)$ for various values of d and k (recall $q = d + k$), from which the amount of computational savings is clear, especially for larger values of k. As such, the implementation of Smolyak's algorithm in this work employs the formulation in Equation 2.48.

2.3.2 Recursive Lagrange Basis Polynomials

Further computational savings can be found in the construction of the univariate Lagrange basis polynomials. Note that our concept of recursive Lagrange polynomials differs from that of the hierarchical Lagrange basis polynomials as summarized in [37]. Here we construct global Lagrange basis polynomials using all m_i nodes of level i instead of the hierarchical Lagrange basis polynomials which are defined locally using a hierarchical structure. Now, since the denominator of Equation 2.19 does not depend on x, we can precompute the denominator for each basis function ℓ_j, a task which only needs to be completed once. On the other hand, to efficiently evaluate the numerator of Equation 2.19 we take advantage of the fact that the sets

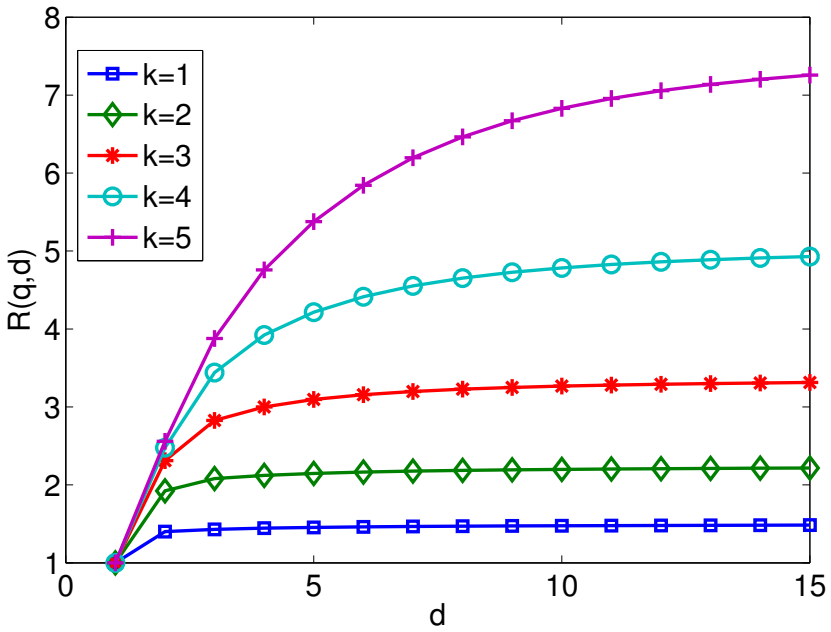


Figure 2.5: The ratio $R(d,k)$ of the number of terms evaluated in the original Smolyak formula to the number of terms evaluated in the new Smolyak formula.

of Chebyshev nodes χ^i are nested. Consider the one-dimensional interpolation problem and let i be the current level for which we want to evaluate the basis polynomials. Instead of explicitly computing each level's basis polynomials, we can use the basis polynomials from level $i-1$.

Let χ^i be the set of m_i nodes for the ith level and define

$$\chi^i_+ = \chi^i \setminus \chi^{i-1} \tag{2.49}$$

so that χ^i_+ contains only the new nodes added to level i. If we first define

$$L^i(x) = \prod_{k=1}^{m_i}(x - x_k), \tag{2.50}$$

the formula for the j^{th} Lagrange basis polynomial of level i is

$$\ell_j^i(x) = \begin{cases} \ell_j^{i-1}(x) \cdot \prod\limits_{x_k \in \chi_+^i} \dfrac{x - x_k}{x_j - x_k} & \text{if } x_j \notin \chi_+^i, \\ L^{i-1}(x) \cdot \prod\limits_{\substack{x_k \in \chi_+^i \\ k \neq j}} \dfrac{x - x_k}{x_j - x_k} & \text{if } x_j \in \chi_+^i. \end{cases} \tag{2.51}$$

Note that each $L^i(x)$ can also be computed recursively as

$$L^i(x) = L^{i-1}(x) \prod_{x_k \in \chi_+^i} (x - x_k). \tag{2.52}$$

We use a similar recursive scheme to compute the analytic derivatives of the one-dimensional Lagrange polynomials. Noting that the derivative of the Lagrange polynomial can be written as

$$\frac{d}{dx}\ell_j^i(x) = \ell_j^i(x) \sum_{k \neq j}^{m_i} \frac{1}{x - x_k}, \tag{2.53}$$

it can be computed in a way similar to Equation 2.51. By computing the Lagrange basis polynomials and their derivatives in this way we are able to reduce the overall computational cost of Smolyak's algorithm.

2.3.3 Numerical Tests

Klimke and Wohlmuth developed the robust Sparse Grid Interpolation Toolbox for MATLAB (see [121] for documentation). In [122] they describe the Toolbox's algorithm for piecewise multilinear hierarchical interpolation on sparse grids, but the Toolbox is also capable of using other bases including the global Lagrange polynomial basis used in Section 2.2. By default the Sparse Grid Toolbox uses a dimensionally adaptive sparse grid algorithm to approximate the given function to within specified error tolerances. The adaptive refinement scheme can be turned off, however, and options can be set to make algorithm perform as detailed in Section 2.2. The Sparse Grid Interpolation Toolbox also contains a gradient option that computes an analytic gradient of the interpolating polynomial. The Sparse Grid Interpolation Toolbox's algorithms involve barycentric Lagrange interpolation and the Discrete Cosine Transform to evaluate Smolyak's formula and compute gradients of the interpolant [120, 121]. We use this Toolbox as a benchmark for our own algorithm. While more state-of-the-art sparse grid packages are available, e.g. SG++ [165], these packages employ spatially adaptive algorithms that would be prohibitively time-consuming considering the cost of the function we wish to approximate. For our application it is more feasible to generate all of the sparse grid points at once and

compute all of the corresponding function evaluations in parallel. The MATLAB Sparse Grid Interpolation Toolbox still represents the state-of-the-art for the computing regime to which we are restricted. Finally, we also compare running times of Judd et al.'s reformulation with and without our recursive Lagrange basis polynomial construction. All tests were performed with MATLAB 2014a on a machine running Mac OS X 10.9.4 with two 2.93 GHz Quad-Core Intel Xeon processors and 16GB RAM.

Consider the task of evaluating Smolyak's interpolation of the function $f(x)$, which we denote by $\mathcal{A}(q,d)[f](x)$, at the point $x \in \mathbb{R}^d$. Our implementation and the Toolbox's both work in two steps. In the first step, everything that can be calculated without knowledge of x is computed. This step takes as input the function $f(x)$, the interpolation domain, and the degree of polynomial exactness k, and computes the sparse grid points, the multi-index set $Q(q,d)$, the coefficients for Smolyak's algorithm, and any bookkeeping data structures. The second step evaluates Smolyak's interpolant at x and involves computing the univariate Lagrange basis polynomials and the tensor products (Equation 2.22) dictated by Smolyak's algorithm.

As detailed in Chapter 4, our application requires that we integrate dynamics on the interpolating polynomial $\mathcal{A}(q,d)[f](x)$ continuously, meaning we must evaluate the same interpolating polynomial and its derivative several thousand times during simulations. With this in mind, we compare performance times of the MATLAB Sparse Grid Interpolation Toolbox and our own implementation only for the second step of the implementation process. The first step is a one-time computational cost and is negligible compared to the cost of simulating dynamics on the interpolant. We note, however, that for larger values of d and k the cost of computing the Smolyak coefficients via Equation 2.47 for our implementation can be quite cumbersome. As one would expect, the associated matrix of this system becomes larger and more ill-conditioned as d and k increase. Computing the coefficients in this way is acceptable for our application, though, since d and k remain relatively small ($d \leq 5$ and $k \leq 6$).

In all Figures shown the method presented in this thesis is referred to as "New Method" and the MATLAB Toolbox is referred to as "Toolbox." All times are given in seconds. Figure 2.6 shows performance scalings in dimension d for evaluating Smolyak's interpolant and its gradient at one point with $k = 4$, and Figure 2.7 shows performance scalings in degree of exactness k for evaluating Smolyak's interpolant and its gradient at one point in 4 dimensions. Tables 2.4 and 2.7 tabulate the data in Figures 2.6 and 2.7, respectively, and also report percent reductions in computation time.

It is clear that the New Method outperforms the MATLAB Toolbox for all reported dimensions d and scale similarly with increasing dimension, with approximately 50-60% reductions in computation time. The New Method also out performs the Toolbox for all degrees of exactness k except for the $k = 1$ case which is insignificant since both methods evaluate the interpolant in less than 0.01 seconds. Here the two methods scale slightly differently, resulting in the increase

in percent reduction in computation time in Table 2.5.

Most important to our application to reaction path following, Figure 2.8 shows performance scalings in the simultaneous evaluation of N points, where N increases in powers of 10 from from 1 to 10^5. Each computation was performed with $d = 4$ and $k = 5$ and also evaluated gradients. Table 2.6 tabulates this data with percent reductions in computation time. Here the New Method greatly outperforms the Toolbox, with 86-99% reductions in computation time. While the Toolbox is useful for the simultaneous evaluation of a few points, it was not designed to approximate a function at a large number of points. The code does not vectorize interpolant evaluation of points, and thus the simultaneous computation of several thousand points is quite expensive.

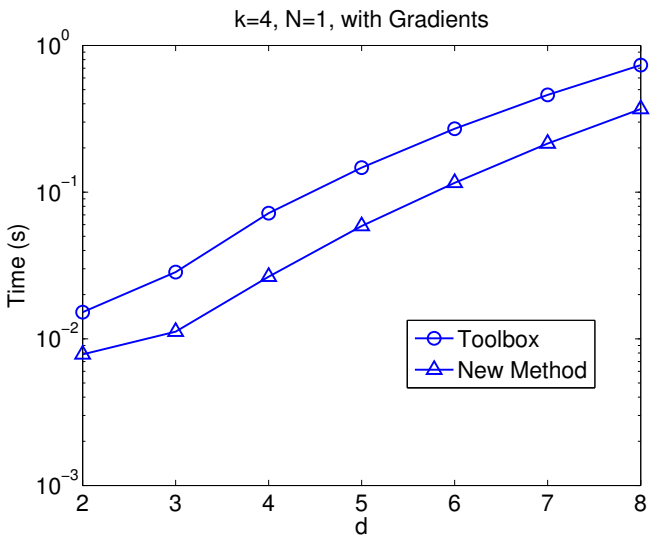


Figure 2.6: Comparison results for MATLAB's Sparse Grid Interpolation Toolbox ("Toolbox") and the implementation presented in this thesis ("New Method"): computation time (in seconds) vs. dimension d for evaluating 1 point with $k = 4$.

Table 2.4: Computation time (in seconds) and percent reduction for MATLAB's Sparse Grid Interpolation Toolbox ("Toolbox") and the implementation presented in this thesis ("New Method") for evaluating 1 point with a degree of exactness of $k = 4$ and increasing dimension d.

d	Toolbox (s)	New Method (s)	Percent Reduction (%)
2	1.52e-2	6.65e-3	56.2
3	2.85e-2	1.05e-2	63.2
4	7.18e-2	2.53e-2	64.8
5	1.47e-1	5.51e-2	62.5
6	2.70e-1	1.10e-1	59.4
7	4.60e-1	2.04e-1	55.8
8	7.34e-1	3.51e-1	52.2

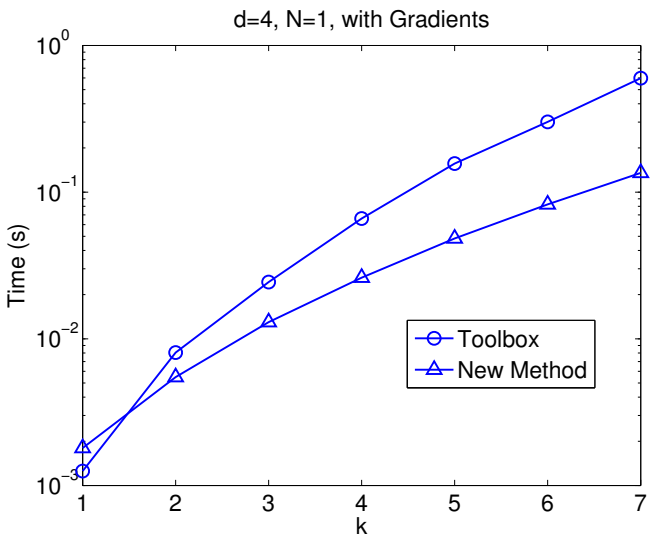


Figure 2.7: Comparison results for MATLAB's Sparse Grid Interpolation Toolbox ("Toolbox") and the implementation presented in this thesis ("New Method"): computation time (in seconds) vs. degree of exactness k for evaluating 1 point in $d = 4$.

Table 2.5: Computation time (in seconds) and percent reduction for MATLAB's Sparse Grid Interpolation Toolbox ("Toolbox") and the implementation presented in this thesis ("New Method") for evaluating 1 point in $d = 4$ dimensions with increasing degree of exactness k.

k	Toolbox (s)	New Method (s)	Percent Reduction (%)
1	1.25e-3	1.73e-3	-37.7
2	8.05e-3	5.50e-3	31.7
3	2.44e-2	1.22e-2	50.1
4	6.61e-2	2.48e-2	62.4
5	1.57e-1	4.60e-2	70.6
6	3.01e-1	7.86e-2	73.9
7	5.98e-1	1.29e-1	78.4

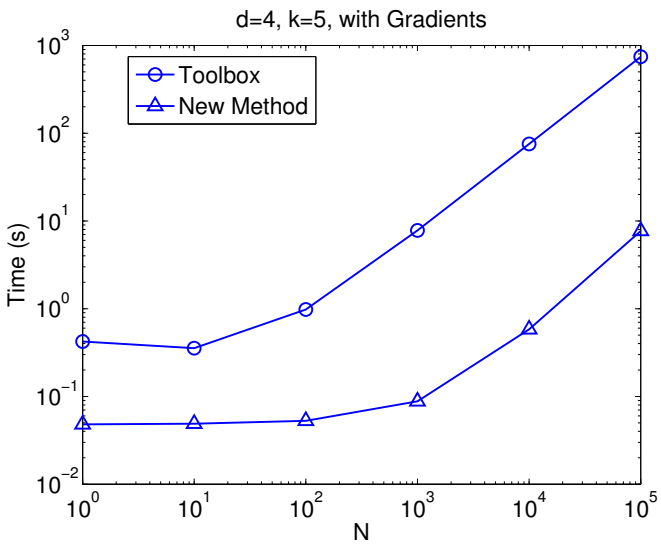


Figure 2.8: Comparison results for MATLAB's Sparse Grid Interpolation Toolbox ("Toolbox") and the implementation presented in this thesis ("New Method"): computation time (in seconds) vs. the number of evaluation points N in $d = 4$ dimensions with $k = 5$, with gradients.

Table 2.6: Computation time (in seconds) and percent reduction for MATLAB's Sparse Grid Interpolation Toolbox ("Toolbox") and the implementation presented in this thesis ("New Method") for evaluating N points with a degree of exactness of $k = 5$ in $d = 4$ dimensions.

N	Toolbox (s)	New Method (s)	Percent Reduction (%)
1	4.22-1	4.68e-2	88.9
10	3.55e-1	4.68e-2	86.8
10^2	9.80e-1	5.08e-2	94.8
10^3	7.80e+0	8.40e-2	98.9
10^4	7.54e+1	5.65e-1	99.3
10^5	7.43e+2	7.5e+0	99.0

Finally, we will compare our implementation with the recursive Lagrange polynomial construction to our implementation without the recursive construction. We will compare the computation time of constructing the univariate Lagrange basis polynomials and their derivatives for the same three cases as the MATLAB Toolbox comparison. First, we fix $k = 4$ and $N = 10^3$ and vary the dimension d. Second, we fix $d = 4$ and $N = 10^3$ and vary the degree of exactness k. Third, we fix $d = 4$ and $k = 5$ and vary the number of simultaneous evaluations N. Figures 2.9 and 2.10 show the results for performance scalings in d and k, respectively, and Figure 2.11 shows the results for performance scalings in N. Tables 2.7-2.9 tabulate the same data as Figures 2.9-2.11, respectively, and also report percent reductions in computation time.

As expected, the implementation with the recursive construction outperforms the implementation without for all test cases. While the computational savings appear to be small for a single evaluation, these savings accumulate over the number of time steps required by integrating systems of differential equations. As discussed in the next section, our application involves continuously following the steepest descent path, a task which requires integrating such a system. As such, any improvement in the cost of a single evaluation of Smolyak's interpolant and gradient could yield large computational savings during an integration routine.

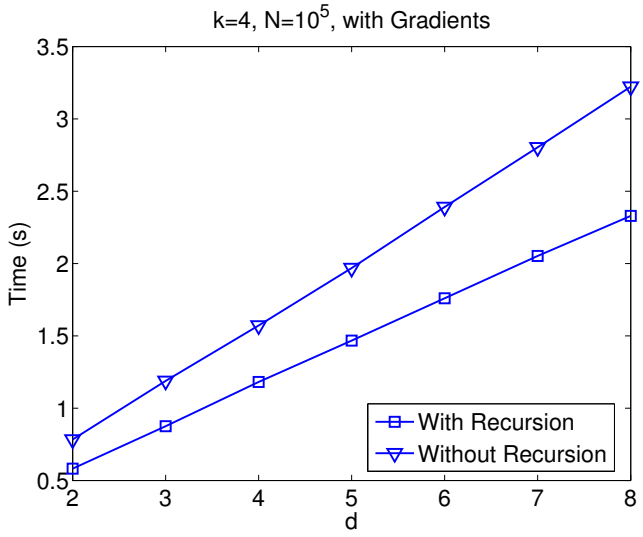


Figure 2.9: Comparison results for our new implementation with and without the recursive Lagrange construction: computation time (in seconds) vs. dimension d for evaluating 10^3 points with $k = 4$.

Table 2.7: Computation time (in seconds) and percent reduction for our new implementation with and without the recursive Lagrange construction for evaluating 10^3 points with a degree of exactness of $k = 4$ and increasing dimension d.

d	Without Recursion (s)	With Recursion (s)	Percent Reduction (%)
2	7.84e-1	5.82e-1	25.8
3	1.19e+0	8.76e-1	26.3
4	1.57e+0	1.18e+0	24.8
5	1.97e+0	1.47e+0	25.5
6	2.39e+0	1.76e+0	26.4
7	2.80e+0	2.05e+0	26.8
8	3.22e+0	2.33e+0	27.7

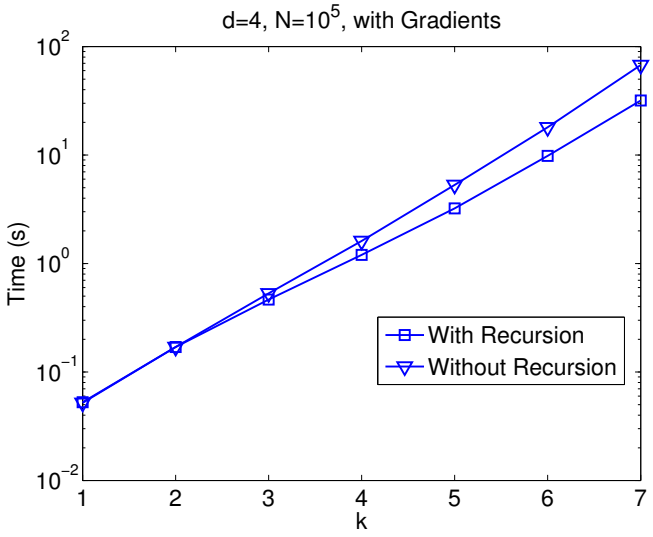


Figure 2.10: Comparison results for our new implementation with and without the recursive Lagrange construction: computation time (in seconds) vs. degree of exactness k for evaluating 10^3 points in $d = 4$.

Table 2.8: Computation time (in seconds) and percent reduction for our new implementation with and without the recursive Lagrange construction for evaluating 10^3 points in $d = 4$ dimensions with increasing degree of exactness k.

k	Without Recursion (s)	With Recursion (s)	Percent Reduction (%)
1	5.19e-2	5.26e-2	-1.32
2	1.70e-1	1.70e-1	0.00
3	5.30e-1	4.65e-1	12.4
4	1.61e+0	1.20e+0	25.5
5	5.30e+0	3.22e+0	39.2
6	1.80e+1	9.83e+0	45.4
7	6.75e+1	3.13e+1	52.9

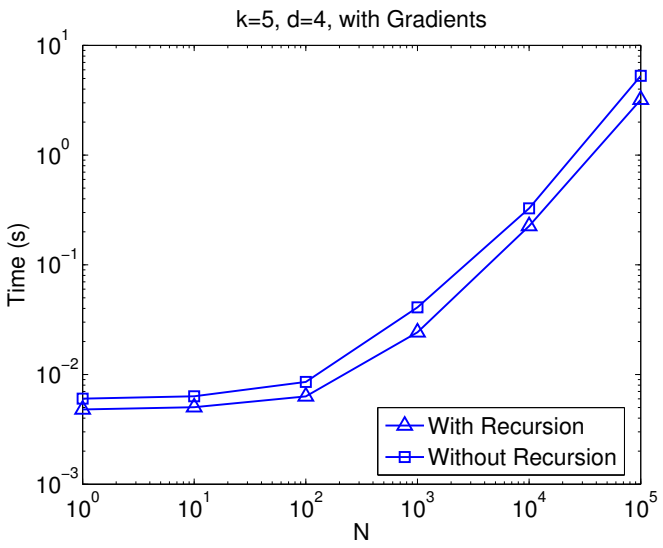


Figure 2.11: Comparison results for our new implementation with and without the recursive Lagrange construction: computation time (in seconds) vs. the number of evaluation points N in $d = 4$ dimensions with $k = 5$, with gradients.

Table 2.9: Computation time (in seconds) and percent reduction for our new implementation with and without the recursive Lagrange construction for evaluating N points with a degree of exactness of $k = 5$ in $d = 4$ dimensions.

N	Without Recursion (s)	With Recursion (s)	Percent Reduction (%)
1	6.03-3	4.81e-3	20.3
10	6.33e-3	5.03e-3	20.5
10^2	8.56e-3	6.31e-3	26.2
10^3	4.10e-2	2.42e-2	40.9
10^4	3.27e-1	2.26e-1	31.0
10^5	5.29e+0	3.20e+0	39.4

2.3.4 Chebyshev Polynomial Basis

Judd et al.'s implementation from [118] uses orthogonal Chebyshev basis polynomials as in [127] and [139] instead of the global Lagrange polynomials from [13] which have been utilized thus far in this Chapter. Traditionally, Lagrange basis polynomials are advantageous since the coefficient associated with each basis polynomial centered at a node x is the function value $f(x)$ as in Equation 2.19. Consequently, there is no need to construct the Vandermonde system of equations to solve for the coefficients. However, this is no longer the case with Judd et al.'s reformulation of Smolyak's algorithm, as we are forced to construct the Vandermonde system (Equation 2.47) to solve for each coefficient b_i. As such, the advantage of using Lagrange basis polynomials is lost.

In this section we consider using Chebyshev basis polynomials instead of the global Lagrange polynomials as a foundation for Smolyak's algorithm. As we will see, Chebyshev basis polynomials have a simple recursive relation that is much easier to implement than the analogous Lagrange basis polynomial recursive relation given in Equation 2.51, and may offer computational savings for our implementation of Smolyak's algorithm.

On the interval $[-1, 1]$, the Chebyshev polynomials are defined as

$$T_k(x) = \cos\left(k \arccos(x)\right) \tag{2.54}$$

for $k = 0, 1, 2, \ldots$. With $T_0(x) = 1$ and $T_1(x) = x$, they can also be defined by the two-term recursive relation

$$T_k(x) = 2xT_{k-1}(x) - T_{k-1}(x) \tag{2.55}$$

for $k = 1, 2, 3, \ldots$ [173]. The derivatives $T_k'(x)$ are also straightforward to compute with this recursive formula. Figure 2.12 shows the first five Chebyshev polynomials defined on $[-1, 1]$. One can define Chebyshev polynomials on an arbitrary interval $[a, b]$ with the mapping

$$s = \frac{2x - (b + a)}{b - a} \tag{2.56}$$

where $x \in [-1, 1]$. Then, as before, $T_0(s) = 1$, $T_1(s) = s$, and

$$T_k(s) = 2sT_{k-1}(s) - T_{k-1}(s). \tag{2.57}$$

For the rest of this section we will assume we are on the interval $[-1, 1]$. For Chebyshev basis polynomials ϕ_{j_i} in Equation 2.46 is simply the jth Chebyshev polynomial $T_j(x)$ in the ith dimension. The cumbersome relabeling of Lagrange basis polynomials as demonstrated in Equation 2.45 is no longer needed.

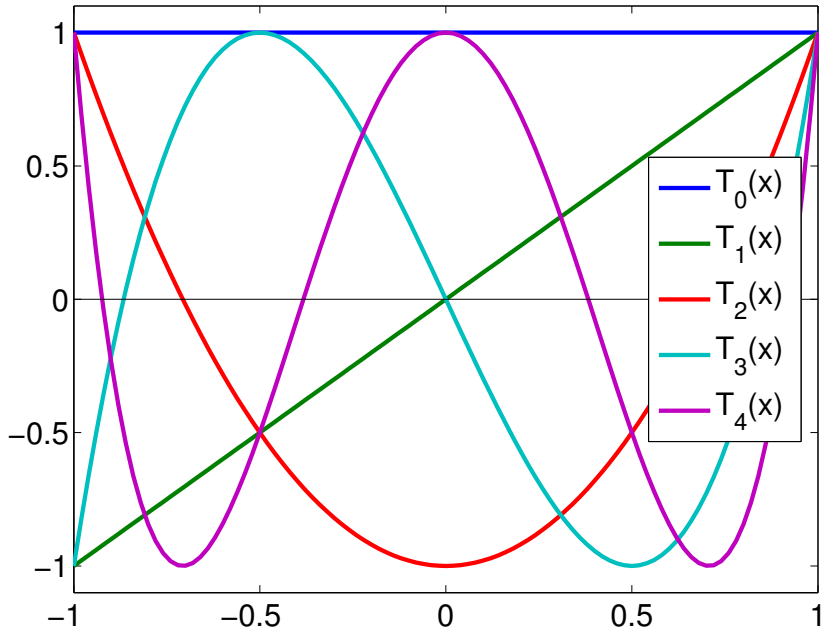


Figure 2.12: Chebyshev polynomials $T_k(x)$ on $[-1, 1]$ for $k = 0, 1, 2, 3, 4$.

We now compare our implementation of Smolyak's algorithm with Lagrange basis polynomials and Chebyshev basis polynomials. We perform the same three numerical tests as before: Table 2.10 shows computation times of $N = 1$ point for the two implementations vs. dimension d with degree of exactness $k = 4$. Table 2.11 shows computation times of $N = 1$ point for the two implementations vs. degree of exactness k in $d = 4$ dimensions. Finally, Table 2.12 shows computation times for the two implementations vs. the number of points N in $d = 4$ dimensions with degree of exactness $k = 5$.

Table 2.10: Computation times (in seconds) for our new implementation with Lagrange and Chebyshev basis polynomials. Here, we evaluate Smolyak's polynomial and its gradient at one point with $k = 4$.

d	Lagrange	Chebyshev
2	6.65e-3	3.85e-3
3	1.05e-2	8.39e-3
4	2.53e-2	2.25e-2
5	5.51e-2	5.17e-2
6	1.10e-1	1.06e-1
7	2.04e-1	2.00e-1
8	3.51e-1	3.52e-1

Table 2.11: Computation times (in seconds) for our new implementation with Lagrange and Chebyshev basis polynomials. Here, we evaluate Smolyak's polynomial and its gradient at one point in $d = 4$ dimensions.

k	Lagrange	Chebyshev
1	1.73e-3	1.45e-3
2	5.50e-3	4.46e-3
3	1.22e-2	1.07e-2
4	2.48e-2	2.27e-2
5	4.60e-2	4.30e-2
6	7.86e-2	7.56e-2
7	1.29e-1	1.25e-1

Table 2.12: Computation times (in seconds) for our new implementation with Lagrange and Chebyshev basis polynomials. Here, we evaluate Smolyak's polynomial and its gradient in $d = 4$ dimensions with $k = 5$.

N	Lagrange	Chebyshev
1	4.68e-2	4.34e-2
10	4.68e-2	4.38e-2
10^2	5.08e-2	4.58e-2
10^3	8.40e-2	6.25e-2
10^4	5.65e-1	3.54e-1
10^5	7.53e+0	4.75e+0

It is clear that the implementation with Chebyshev polynomials performs no worse than the implementation of Lagrange polynomials for the values of degree of exactness k and dimension d required by our application. The implementation with Chebyshev polynomials outperforms the implementation with Lagrange polynomials as we increase the number of simultaneous evaluations N. This is especially important considering that our application requires evaluating Smolyak's interpolating polynomial and its gradient at several thousand points simultaneously.

To summarize, the advantages of using Chebyshev basis polynomials are threefold. First, their simple recursive definition is much easier to implement than the recursive definition of Lagrange basis polynomials in this section. Second, the bookkeeping associated with the relabeling of Lagrange basis polynomials as in Equation 2.45 is avoided. Finally, as shown in Tables 2.10-2.12, evaluating Smolyak's algorithm with Chebyshev basis polynomials is no slower and in some cases faster than evaluating Smolyak's algorithm with Lagrange basis polynomials. For these reasons we implement Smolyak's algorithm with Chebyshev basis polynomials as in [118].

2.4 Anisotropic and Adaptive Sparse Grids

To conclude this chapter we briefly mention adaptive sparse grids. Considering the number of grid points and basis functions for all variables, Smolyak's algorithm and conventional sparse grid algorithms treat all dimensions equally. Anisotropic and adaptive sparse grids, on the other hand, allow for a differential treatment of variables. To define an anisotropic sparse grid, let k_j denote an approximation level in dimension j. The maximum index admitted via Smolyak's algorithm is then $i_j^{\max} = k_j + 1$. A sparse grid is "anisotropic" if there exists a dimension j such that $k_j \neq k_i$ for all $i \neq j$. Otherwise, the sparse grid is called "isotropic". The latter of these two types of grids is what is used in Smolyak's original algorithm. Figure 2.13 shows an example

of an anisotropic sparse grid with $i_1^{\max} = 3$, $i_2^{\max} = 2$, and $|\mathbf{i}| \leq 5$. Nonadaptive anisotropic sparse grids are a double-edged sword. On one hand, they can decrease the overall number of grid points and thus reduce the computational expense, but on the other hand the quality of approximation is not as good [118]. Adaptive sparse grids, however, refine the index set, grid points, or both until a sufficiently accurate approximation is attained.

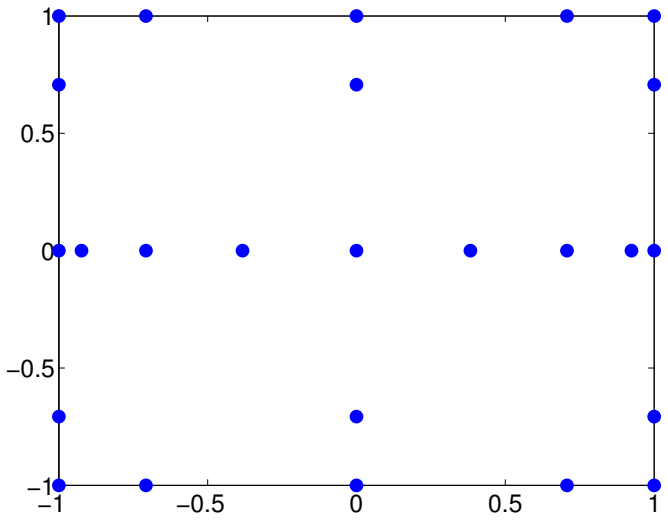


Figure 2.13: An anisotropic sparse grid with $i_1^{\max} = 3$, $i_2^{\max} = 2$, and $|\mathbf{i}| \leq 5$.

The goal of adaptive sparse grid algorithms is to choose k_j for each dimension $j = 1, \ldots, d$ so that a relative or absolute error tolerance is reached. Dimensionally adaptive sparse grids were originally proposed in the context of numerical quadrature by Gerstner and Griebel in [76] and Hegland in [101], and spatially adaptive sparse grids were first used by Griebel in [86] for the solution to partial differential equations. Adaptive sparse grids usually employ flexible piecewise linear or higher order basis functions to allow for consecutive refinements of the solution in areas where higher accuracy is needed. Conversely, our method (reviewed in Section 2.2) uses less flexible global Lagrange and Chebyshev polynomial basis functions which restrict refinement to specific dimensions and not specific areas of the domain.

We turn our attention first to dimensionally adaptive sparse grids. Spatially adaptive sparse grids are related and based on the same algorithm, but are understandably more complicated.

2.4.1 Dimensionally Adaptive Sparse Grids

Gerstner and Griebel generalized the sparse grid construction in [76] by considering a special set of admissible indices. An index set $\mathcal{I}$ is called admissible if for all $\mathbf{i} \in \mathcal{I}$,

$$\mathbf{i} - \mathbf{e}_j \in \mathcal{I} \text{ for } 1 \leq j \leq d,\ i_j > 1 \tag{2.58}$$

where $\mathbf{e}_j$ is the jth unit vector. As noted in [76], the admissibility condition ensures the validity of the telescope sum expansion of the general sparse grid technique using the difference operators $\Delta^1_{i_j}$. For an admissible index set $\mathcal{I} \in \mathbb{N}^d$ the general sparse grid construction becomes

$$\mathcal{A}(d, \mathcal{I})[f] = \sum_{\mathbf{i} \in \mathcal{I}} (\Delta^{i_1} \otimes \ldots \otimes \Delta^{i_d})[f]. \tag{2.59}$$

The goal of dimensionally adaptive sparse grids is to iteratively construct the index set $\mathcal{I}$ to minimize the interpolation or integration error ϵ for a set number of function evaluations. Here we summarize the algorithm from [76]. The dimensionally adaptive sparse grid algorithm starts with the root index set $\{\mathbf{1}\} = \{(1, \ldots, 1\}$ and successively adds indices such that first, the resulting index sets remain admissible, and second, a possibly large error reduction is achieved. To estimate the error for a given index $\mathbf{i}$ the error indicator $g_\mathbf{i}$ is computed from the difference operator

$$\Delta_\mathbf{i}[f] = (\Delta^{i_1} \otimes \ldots \otimes \Delta^{i_d})[f] \tag{2.60}$$

and from other values attributed to $\mathbf{i}$ such as the number of function evaluations required to evaluate $\Delta_\mathbf{i}[f]$. In [124], for example, Klimke suggests the error indicator

$$g_\mathbf{i} = \frac{1}{n_\mathbf{i}} \sum_\mathbf{j} |w^\mathbf{i}_\mathbf{j}| \tag{2.61}$$

where $w^\mathbf{i}_\mathbf{j}$ are the hierarchical surpluses of the sub-grid $\chi_\mathbf{i} = \chi^{i_1}_\Delta \times \ldots \times \chi^{i_d}_\Delta$, $j_k = 1, \ldots, m^\Delta_{i_k}$, and $n_\mathbf{i}$ is the number of new function evaluations required by the set of nodes $\chi_\mathbf{i}$.

For any index $\mathbf{i}$ we define its forward neighborhood as the set of d indices

$$\mathbf{i}_F = \{\mathbf{i} + \mathbf{e}_j,\ 1 \leq j \leq d\} \tag{2.62}$$

and similarly its backward neighborhood as the set of d indices

$$\mathbf{i}_B = \{\mathbf{i} - \mathbf{e}_j,\ 1 \leq j \leq d\}. \tag{2.63}$$

Gerstner and Griebel partition the index set $\mathcal{I}$ into an active set of indices $\mathcal{A}$ and an old set of

indices $\mathcal{O}$. The active set $\mathcal{A}$ contains the indices of $\mathcal{I}$ whose error indicators have been computed but the error indicators of all their forward neighbors have not yet been considered. The old index set $\mathcal{O}$ contains all other indices of the index set $\mathcal{I}$. Note that $\mathcal{A} \cup \mathcal{O} = \emptyset$. The error indicators associated with $\mathbf{i} \in \mathcal{A}$ can act as an global estimate of the error γ. Again from [124], Klimke uses the estimate

$$\gamma = \max_{\mathbf{i} \in \mathcal{A}, \mathbf{j}} |w_{\mathbf{j}}^{\mathbf{i}}|. \tag{2.64}$$

At each iteration of the adaptive algorithm, the index $\mathbf{i}$ with the largest associated error indicator is selected from the active set $\mathcal{A}$ and put in the old set $\mathcal{O}$. The error indicators of the admissible forward neighbors of $\mathbf{i}$ are then computed and their indices are put into the active index set $\mathcal{A}$. The corresponding values of $\Delta_{\mathbf{i}}$ are then added to the current approximation result and the global error is recalculated. If either the global error estimate falls below a specified tolerance or the number of allowed function evaluations is exceeded, then the algorithm is stopped and the current approximation is returned. Otherwise, the algorithm repeats this process with the new index with the largest error indicator, and so on.

As noted in [76], error estimation is a crucial part of the algorithm, and different choices of error indicator functions $g_{\mathbf{i}}$ can lead to different adaptive grids. If the error indicator for an index $\mathbf{i}$ is very small, there will be no future refinement in its forward neighborhood. This may cause the algorithm to stop prematurely if one of the foreword neighbors has a significantly larger error indicator. To tackle this problem Gerstner and Griebel employ the error indicator

$$g_{\mathbf{i}} = \max \left\{ \lambda \frac{|\Delta_{\mathbf{i}} f|}{|\Delta_{\mathbf{1}} f|}, (1-\lambda) \frac{1}{n_{\mathbf{i}}} \right\} \tag{2.65}$$

where $\lambda \in [0, 1]$. With $\lambda = 1$ the algorithm becomes greedy and disregards the number of function evaluations. A greedy algorithm would be ideal in the case where the function is known to be very smooth and the error estimates would decay with increasing indices anyway. Classical sparse grids are employed by selecting $\lambda = 0$.

On the other hand, Klimke proposes a similar approach in [124] that introduces the parameter λ in a separate indicator that relates the largest index $\mathbf{i} \in \mathcal{I}$ and the smallest index $\mathbf{i} \in \mathcal{A}$. In this way, the foreword neighbors of an index $\mathbf{i}^{act} = \text{argmin}_{\mathbf{i} \in \mathcal{A}} |\mathbf{i}|$ if

$$\frac{|\mathbf{i}^{act}|}{\max_{\mathbf{i} \in \mathcal{I}} |\mathbf{i}|} \leq (1 - \lambda) \tag{2.66}$$

regardless of the error indicator $g_{\mathbf{i}}$, $\mathbf{i} \in \mathcal{A}$.

As noted in both [124] and [76], efficient data structures must be used in order to store the indices in such a way that it is easy to: insert and remove indices from $\mathcal{A}$, insert indices into $\mathcal{O}$, find the index in the active index set with the largest error, and check if an index

is admissible. The details of such data structures is omitted here, but can be found in [124] and [76]. Finally, we note that the MATLAB Sparse Grid Interpolation Toolbox [124, 121] utilizes Klimke's implementation of the dimensionally adaptive sparse grid algorithm and is freely available online at `http://www.ians.uni-stuttgart.de/spinterp`.

2.4.2 Spatially Adaptive Sparse Grids

Spatially, or locally, adaptive sparse grids were first proposed by Griebel in [86] for the solution of elliptic partial differential equations, and have since been used for a variety of other applications [136, 137, 138, 166, 165, 38]. As opposed to the generalized sparse grid method of the preceding section, locally adaptive sparse grids attempt to reduce the number of points in a sparse grid by refining the grid in rapidly varying or discontinuous regions. The method is implicitly dimension-adaptive but is capable of adding necessary points in dimensions that the dimensionally adaptive algorithm might deem unimportant [111]. Returning to the sparse grid construction and notation from Section 2.1.1, from [86] the locally adaptive sparse grid representation of a function $u(x)$ is given by

$$u^{\epsilon,\|\cdot\|}(x) = \sum_{(\mathbf{l},\mathbf{i})\in\mathcal{A}(u,\epsilon,\|\cdot\|)} v_{\mathbf{l},\mathbf{i}} \cdot \phi_{\mathbf{l}.\mathbf{i}}(x) \tag{2.67}$$

where ϵ is the the tolerance for hierarchical surpluses, and $\mathcal{A}(u,\epsilon,\|\cdot\|)$ denotes the set of active indices. Indices $(\mathbf{l},\mathbf{i}) \in \mathcal{A}(u,\epsilon,\|\cdot\|)$ satisfy either of the two requirements

1. $\mathbf{i} \in I_1, \|v_{\mathbf{l},\mathbf{i}} \cdot \phi_{\mathbf{l}.\mathbf{i}}(x)\| \geq \epsilon$
2. $\exists(\mathbf{k},\mathbf{j}) : \mathbf{k} \geq \mathbf{l}, \|v_{\mathbf{k},\mathbf{j}} \cdot \phi_{\mathbf{k},\mathbf{j}}(x)\| \geq \epsilon,\ \mathrm{supp}(\phi_{\mathbf{k},\mathbf{j}}) \cap \mathrm{supp}(\phi_{\mathbf{l},\mathbf{i}}) \neq \emptyset$

where $\mathrm{supp}(\phi) = \{x : \phi(x) > 0\}$ is the open support of ϕ. Naturally, one cannot compute the infinite expansion of $u(x)$ and then remove the indices whose error indicator is within the given tolerance. Instead, Griebel proceeds in a top down approach that starts from the coarsest level and refines recursively level by level [86].

In [111] Jakeman and Roberts incorporate a locally adaptive procedure into the generalized sparse grid algorithm. For each index $\mathbf{i}$ they define the active point set $\mathcal{A}_\mathbf{i}$ and the redundant point set $\mathcal{R}_\mathbf{i}$. The active point set contains all admissible points associated with the index $\mathbf{i}$ with an error indicator $\gamma_{\mathbf{i},\mathbf{j}} \geq \epsilon$, and the redundant point set contains all admissible points with an error indicator $\gamma_{\mathbf{i},\mathbf{j}} < \epsilon$. A point is admissible if one of its d possible ancestors exists in the grids associated with the $\mathbf{i}_B$. Jakeman and Roberts employ the error indicator

$$\gamma_{\mathbf{i},\mathbf{j}} = |v_{\mathbf{l},\mathbf{i}} \cdot w_{\mathbf{l},\mathbf{i}}| \tag{2.68}$$

for each sparse grid point $x_{\mathbf{l},\mathbf{i}}$ where the weights

$$w_{\mathbf{l},\mathbf{i}} = \int_{I_x} \phi_{\mathbf{i},\mathbf{j}}(x) d\mu(x) \tag{2.69}$$

can be easily calculated without any extra function evaluations. The point $x_{\mathbf{l},\mathbf{i}}$ is added to the active point set $\mathcal{A}_{\mathbf{i}}$ if $\gamma_{\mathbf{i},\mathbf{j}} \geq \epsilon$, otherwise it is added to the redundant index set $\mathcal{R}_{\mathbf{i}}$.

Pflüger has implemented a spatially adaptive sparse grid algorithm in the software package SG++ [165]. The libraries are written in C++ and Python using the algorithm and data structures as described in [86]. Documentation and code can be found at `http://www5.in.tum.de/SGpp/releases/index.html`.

Chapter 3

Background Chemistry

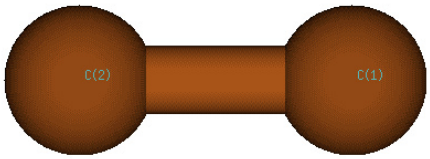
3.1 Molecules

A molecule is a group of atoms that are chemically bonded together and form when two or more atoms bond by sharing electrons and become connected by attractive forces. Molecules can be made from atoms of the same element (e.g. O_2) or atoms of different elements (e.g. H_2O). A molecule consists of the nuclei of the atoms and electrons, which are described by one-electron wavefunctions called orbitals. Figure 3.1 shows an acrolein molecule (C_3H_4O) with and without electron orbitals.

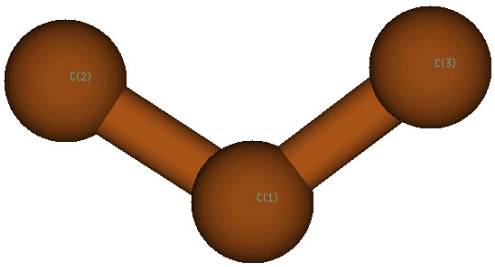
Figure 3.1: An acrolein (C_3H_4O) molecule without electron orbitals (left) and with electron orbitals (right).

The physical arrangement of the atoms within a molecule is called its molecular geometry. Molecular geometry can be specified by the location of the nuclei in Cartesian coordinates,

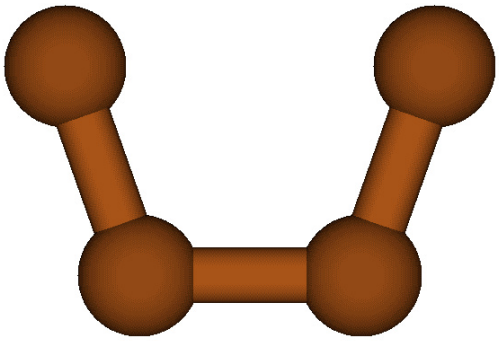
but it is more useful to describe the geometry via internal coordinates [163]. The three types of internal coordinates are: bond length, bond angle and dihedral angle. A bond length is the distance between two bonded nuclei and is measured by the atomic unit angstroms, Å, where 1 Å$= 10^{-10}$ meters. A bond angle is the angle formed by three nuclei connected by two bonds. A dihedral angle, or torsion angle, requires 4 nuclei connected by three bonds and is measured as the angle formed by the projection of the two outer bonds onto the plane perpendicular to the center bond. Bond angles and dihedral angles are measured in degrees. An example of each of these types of coordinates is shown in Figure 3.2. A molecule with N atoms is uniquely



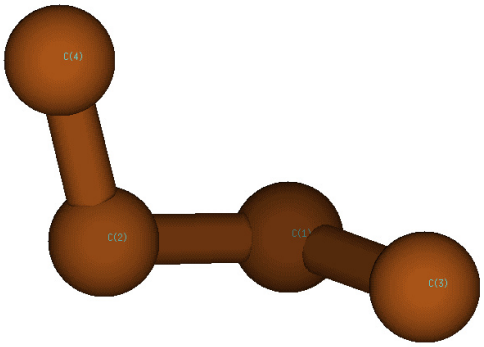
Bond length



Bond angle of 120°



Dihedral angle of 0°



Dihedral angle rotated 90°

Figure 3.2: Examples of molecular geometry coordinates.

determined by $3N-6$ coordinates: $N-1$ bond lengths, $N-2$ bond angles and $N-3$ dihedral angles. These coordinates are collected into a table called a Z-matrix.

Example 3.1.0.1. *Ethylene Z-matrix. Table 3.1 shows a sample Z-matrix for ethylene, C_2H_4. Row i of the Z-matrix gives the internal coordinates for atom i within the molecule. Atom1, Atom2, and Atom3 are the numbers of previously-specified atoms and are used to define atom i's position. The position of the atom i is specified by giving the length of the bond connecting Atom1 and atom i, the bond angle formed by this bond and the bond connecting Atom1 and Atom2, and the dihedral angle created by the plane containing atom i, Atom2, and Atom3, and the plane containing atom i, Atom1, and Atom2.*

Table 3.1: Example Z-matrix for Ethylene, C_2H_4.

Atom	Element	Atom1	Bond Length	Atom2	Bond Angle	Atom3	Dihedral Angle
1	C	-	-	-	-	-	-
2	C	1	1.3 Å	-	-	-	-
3	H	1	1.1 Å	2	120°	-	-
4	H	1	1.1 Å	2	120°	3	180°
5	H	2	1.1 Å	1	120°	3	0°
6	H	2	1.1 Å	1	120°	3	180°

Figure 3.3 shows the Ethylene molecule as described by the Z-matrix in Table 3.1 and the molecule if we change the last dihedral angle from 180 to 100 degrees.

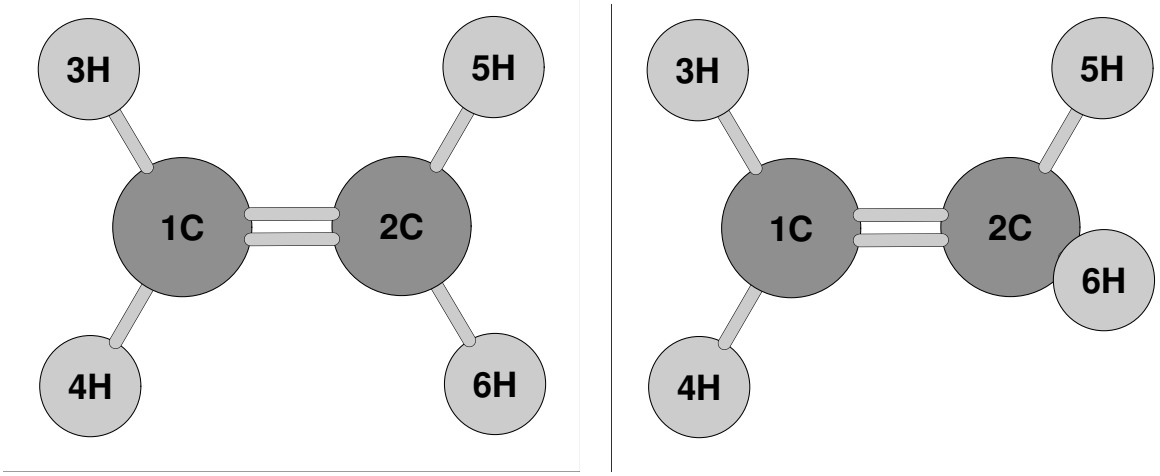


Figure 3.3: Ethylene as described in Table 3.1 with 180 degree (left) and 100 degree (right) dihedral angle defined by atoms 6, 2, 1 and 3 (left).

3.2 Quantized Operators

While the Z-matrix contains information about the nuclei of the atoms in a molecule, it does not give any information about the electrons' orbitals. Fortunately, the coordinates within the Z-matrix are sufficient for uniquely determining these orbitals [224]. The orbitals are described by the solutions to the time-independent Schrödinger equation

$$\hat{H}\psi = E\psi \tag{3.1}$$

where E is the energy, ψ is the wavefunction that completely describes the state of the system, and $\hat{H}$ is the Hamiltonian operator for a molecule of k atoms with n electrons with fixed nuclei. The assumption of fixed nuclei is appropriate since nuclei are much heavier than electrons. Even though the electrons' velocities are greater than the nuclei's, the nuclei's kinetic energy is far greater than surrounding electrons' kinetic energy. Since we are interested in studying the electrons we neglect the nuclei's kinetic energy. This approximation is commonly known as the Born-Oppenheimer Approximation [25] and is central to modern quantum chemistry. The wavefunction ψ gives us a probabilistic description of the system; $|\psi(\mathbf{x})|^2$ is the probability density for finding an electron at the coordinates in $\mathbf{x}$. Wavefunctions ψ and energies E that are solutions to Equation 3.1 are eigenfunction/eigenvalue pairs of the operator $\hat{H}$. As a function of geometry $\mathbf{x}$, $E(\mathbf{x})$ is called the *potential energy surface* (PES), a concept that will be more thoroughly introduced in Chapter 4.

The Hamiltonian $\hat{H}$ is the sum of the kinetic energy operators for the electrons and potential energy operators that result from particle attraction and repulsion [132]. The kinetic energy operator for a single electron is

$$\hat{T} = -\frac{h^2}{8\pi^2 m_e}\nabla^2. \tag{3.2}$$

where h is Planck's constant ($6.6 \times 10^{-34}\ J \cdot s$), m_e is the mass of an electron ($9.109 \times 10^{-31}\ kg$), and ∇ is the Laplace operator. Defining

$$\hbar = \frac{h}{2\pi},$$

the kinetic energy operator can also be written as

$$\hat{T} = -\frac{\hbar^2}{2m_e}\nabla^2. \tag{3.3}$$

Recall that for two electrons separated by a distance r, the electric potential energy is given

by

$$\frac{e^2}{4\pi\epsilon_0 r} = \frac{e'^2}{r}$$

where $e = 1.6 \times 10^{-19}\ C$ is the charge of a proton, $\epsilon_0 = 8.854 \times 10^{-12} \frac{C^2}{Nm^2}$ is the permittivity of vacuum and $e' = \frac{e}{\sqrt{4\pi\epsilon_0}}$. Thus for our system of k nuclei and n electrons the potential energy operator is

$$\hat{V} = \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} \frac{Z_i Z_j e'^2}{r_{ij}} - \sum_{i=1}^{n} \sum_{j=1}^{k} \frac{Z_j e'^2}{r_{ij}} + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{e'^2}{r_{ij}}$$

where Z_j is the number of protons in the j^{th} nucleus. The first term is the potential energy of the repulsions among the nuclei, the second term is the potential energy of the attractions among the electrons and nuclei and the last term is the potential energy of the repulsions among the electrons. Combining the kinetic and potential energy terms, we have

$$\hat{H} = \hat{T} + \hat{V} = -\frac{\hbar}{2m_e}\nabla^2 + \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} \frac{Z_i Z_j e'^2}{r_{ij}} - \sum_{i=1}^{n} \sum_{j=1}^{k} \frac{Z_j e'^2}{r_{ij}} + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{e'^2}{r_{ij}}. \tag{3.4}$$

As we will see, applying this Hamiltonian $\hat{H}$ to the time-independent Schrödinger equation (Equation 3.1) yields discrete values of the energy E. An operator with this property is called a quantized operator. In the following example we will explicitly solve a problem from quantum mechanics to demonstrate this property.

Example 3.2.0.1. *Particle in a Box: here we will consider a single particle with mass m in a one-dimensional box where the potential energy is defined as*

$$V(x) = \begin{cases} 0 & \text{if } 0 \leq x \leq a \\ \infty & \text{otherwise} \end{cases}$$

and is depicted in Figure 3.4.

The Hamiltonian for this system is

$$\hat{H} = -\frac{\hbar^2}{2m}\frac{d^2}{dx^2} + V(x),$$

thus our Schrödinger equation for $x \in \mathbb{R}$ is

$$-\frac{\hbar^2}{2m}\frac{d^2}{dx^2}\psi(x) + V(x)\psi = E\psi. \tag{3.5}$$

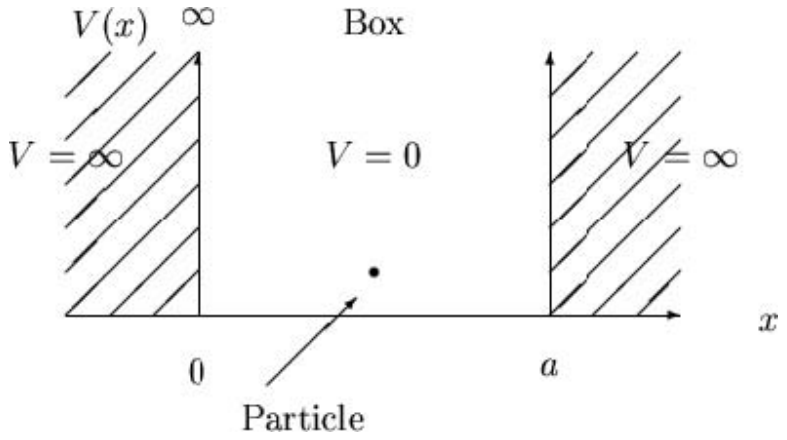


Figure 3.4: Potential energy function for a particle in a one-dimensional box.

Taking limits outside of the box, Equation 3.5 becomes

$$-\frac{\hbar^2}{2m}\frac{d^2}{dx^2}\psi = (E-\infty)\psi$$

which implies

$$\psi = \frac{1}{\infty}\frac{d^2}{dx^2}\psi$$

since $E - \infty = -\infty$. *From this we can conclude that* $\psi(x) = 0$ *outside of the box, so we only need to solve Equation 3.5 inside the box. For* $0 \leq x \leq a$ *Equation 3.5 is simplified to*

$$-\frac{\hbar^2}{2m}\psi = E\psi.$$

From [132], general solutions to this type of differential equation are

$$\psi(x) = c_1 \cos\left(\frac{\sqrt{2mE}}{\hbar}x\right) + c_2 \sin\left(\frac{\sqrt{2mE}}{\hbar}x\right).$$

Since $|\psi(x)|^2$ *is the probability density for finding the particle at* $x \in \mathbb{R}$ *and the probability of finding the particle outside the box is zero, our boundary conditions are* $\psi(0) = 0$ *and* $\psi(a) = 0$. *Enforcing these conditions, we find that* $c_1 = 0$ *and*

$$0 = c_2 \sin\left(\frac{\sqrt{2mE}}{\hbar}a\right).$$

Since we are not interested in the trivial solution $\psi(x) = 0$, we assume $c_2 \neq 0$ and thus

$$\frac{\sqrt{2mE}}{\hbar}a = \pm n\pi, \;\; n = 1, 2, \ldots$$

Finally, solving for the energy of the system yields

$$E = \frac{n^2h^2}{8ma^2}, \;\; n = 1, 2, \ldots \tag{3.6}$$

To solve for our wavefunction ψ, we use the normalization requirement of probability densities to solve for c_2. This yields

$$\begin{aligned} \int_{-\infty}^{\infty} |\psi|^2 dx &= 1 \\ \Rightarrow \int_0^a \left| c_2 \sin\left(\frac{n\pi x}{a}x\right)\right|^2 dx &= 1 \\ \Rightarrow c_2 &= \sqrt{\frac{2}{a}} \end{aligned}$$

and thus our wavefunction is

$$\psi = \sqrt{\frac{2}{a}} \sin\left(\frac{n\pi x}{a}\right), \;\; n = 1, 2 \ldots.$$

Note that $E = 0$ is not a solution since it yields the wavefunction $\psi = 0$, so $n = 0$ is not allowed. Equation 3.6 shows that only discrete values of E are allowed, and these values depend on the so-called quantum number n. Therefore the energy of the system is quantized and $\hat{H}$ is a quantized operator. The lowest energy state of a system is called the ground state, and any higher energy state is called an excited state.

3.3 Hydrogen Atom

The only chemical systems where a wavefunction may be computed analytically is the family of hydrogen-like ions, or atoms made up of any atomic nucleus and just one electron. Examples include hydrogen itself, He^+ (helium with one less electron) and Li^{2+} (lithium with two fewer electrons). We will let Z be the number of protons in the nucleus of an atom. $Z = 1$ for hydrogen and $Z = 3$ for lithium, for example. The Hamiltonian for a hydrogen-like ion is simplified from Equation 3.4 to

$$\hat{H} = -\frac{\hbar^2}{2\mu}\nabla^2 - \frac{Ze'^2}{r},$$

where

$$\mu = \frac{m_e m_n}{m_e + m_n}$$

is the reduced mass of the atom, m_n is the mass of the nucleus, and

$$e' = \frac{e}{\sqrt{4\pi\epsilon_0}}$$

where again $e = 1.6 \times 10^{-19}$ C is the charge of a single proton. Before solving the Schrödinger equation with this Hamiltonian, we will first introduce another operator that will simply things for us later. In quantum mechanics the square of the orbital-angular-momentum operator $\hat{L}^2$ is given by [132]

$$\hat{L}^2 = -\hbar^2 \left(\frac{\partial^2}{\partial\theta^2} + \cot\theta \frac{\partial}{\partial\theta} + \frac{1}{\sin^2\theta} \frac{\partial^2}{\partial\phi^2} \right).$$

For the purposes of this thesis we only define the operator mathematically and give its eigenfunctions and eigenvalues. We refer the reader to [132] for a more rigorous treatment of orbital-angular-momentum operators. In regards to the eigenvalue problem

$$\hat{L}^2 Y(\theta, \phi) = cY(\theta, \phi),$$

the eigenvalues are

$$c = l(l+1)\hbar^2, \;\; l = 0, 1, 2, \ldots$$

and eigenfunctions are given by

$$Y_l^m(\theta, \phi) = \frac{1}{\sqrt{2\pi}} S_{l,m}(\theta) e^{im\phi}$$

where $S_{l,m}(\theta)$ are associated Legendre functions given by

$$S_{l,m}(\theta) = sin^{|m|}\theta \sum_j a_j \cos^j\theta$$

with the coefficients given by the recursion relation

$$a_{j+2} = \frac{[(j+|m|)(j+|m|+1) - l(l+1)]}{(j+1)(j+2)} a_j.$$

The sum is over even (starting at 0) or odd values of j depending on whether $l - |m|$ is even or odd. a_0 and a_1 can be found by imposing normalization constraints on the wavefunction. The possible values for the quantum number m are

$$m = -l, -l+1, \ldots, l-1, l.$$

The functions $Y_l^m(\theta, \phi)$ are called spherical harmonics and are the wavefunctions for a quantum two-particle rigid rotor system. In this system, two particles are held at a fixed distance apart from each other and one is allowed to revolve around the other. With the Born-Oppenheimer approximation, this is very much like an electron orbiting around a fixed nucleus.

Now, to solve the Schrödinger equation for the hydrogen-like ions we will reformulate the problem in spherical coordinates. From [132], the Laplacian operator in spherical coordinates is

$$\nabla^2 = \frac{\partial^2}{\partial r^2} + \frac{2}{r}\frac{\partial}{\partial r} - \frac{1}{r^2\hbar^2}\hat{L}^2,$$

so our Hamiltonian becomes

$$\hat{H} = -\frac{\hbar^2}{2\mu}\left(\frac{\partial^2}{\partial r^2} + \frac{2}{r}\frac{\partial}{\partial r}\right) + \frac{1}{2\mu r^2}\hat{L}^2 - \frac{Ze'^2}{r}.$$

Since the potential energy of a Hydrogen atom is a function of r only, the properties of eigenfunctions allow us to write our wavefunction ψ as a spherical harmonic $Y(\theta, \phi)$ multiplied by a radial function $R(r)$,

$$\psi(r, \theta, \phi) = R(r)Y_l^m(\theta, \phi).$$

Applying the Hamiltonian and dividing both sides of the Schrödinger equation by $Y_l^m(\theta, \phi)$ yields the differential equation

$$R''(r) + \frac{2}{r}R'(r) + \left(\frac{2E}{ae'^2} + \frac{2Z}{ar} - \frac{l(l+1)}{r^2}\right)R(r) = 0 \tag{3.7}$$

where $a = \hbar^2/\mu e'^2$.

Solving Equation 3.7 is tedious and is omitted here for the sake of simplicity, but it is possible to find an analytic solution. From [132],

$$R(r) = R_{nl}(r) = r^l e^{-Zr/na}\sum_{j=0}^{n-l-1} b_j r^j,$$

where the coefficients b_j are found by the recursion relation

$$b_{j+1} = \frac{2Z}{na}\frac{j+l+1-n}{(j+1)(j+2l+2)}b_j$$

where b_0 can be found by enforcing normalization requirements. Thus the complete hydrogen wavefunctions are

$$\psi_{nlm} = \frac{1}{\sqrt{2\pi}}R_{nl}(r)S_{lm}(\theta)e^{im\phi} \tag{3.8}$$

with quantum numbers $n = 1, 2, \ldots$, $l = 0, 1, \ldots, n-1$, and $m = -l, -l+1, \ldots, l-1, l$. The

associated energies are given by

$$E = -\frac{Z^2 \mu e'^4}{2n^2 \hbar^2}. \tag{3.9}$$

Finally, note that for any value of n we can have n different values of l and $2l+1$ values of m, implying that each energy level has n^2 independent wavefunctions. Energy levels that have more than one independent wavefunction are called degenerate.

3.4 Spin

To obtain a complete description of atoms and molecules, we must consider an additional property of electrons called spin. Electrons have angular momentum due to their motion just like all other particles from classical mechanics, but they also have an intrinsic spin angular momentum. We must modify our wavefunction for an electron to reflect this property. The spin of an electron has two possible values, $m_s = \frac{1}{2}$ (spin "up") and $m_s = -\frac{1}{2}$ (spin "down"). The components of the Hamiltonian operator that involve spin do not interact with the spatial variables, so we can separate the one-electron wavefunction as

$$\psi(x, y, z, m_s) = \psi(x, y, z) g(m_s).$$

A wavefunction that is the product of a spatial wavefunction and a spin function $g(m_s)$ is called a spin-orbital. Spin functions are commonly denoted

$$g\left(\frac{1}{2}\right) = \alpha \text{ and } g\left(-\frac{1}{2}\right) = \beta.$$

While spin has no effect on the energy of the system, it does impose additional constraints on the wavefunction due to the requirement of indistinguishability of identical particles in quantum mechanics [132]. The Pauli exclusion principle [162] states that no two electrons can occupy the same spin-orbital. Mathematically, this means that the wavefunction of a system of electrons must be antisymmetric with respect to the interchange of any two electrons. In other words, if we exchange the ordering of any two electrons in our system, the wavefunction must be multiplied by -1. If we let $q_i = (x_i, y_i, z_i, m_{si})$ denote the Cartesian coordinates and spin of the i^{th} electron, our wavefunctions must satisfy

$$\hat{P}_{ij} \psi(q_1, \ldots, q_n) = -\psi(q_1, \ldots, q_n)$$

where $\hat{P}_{ij}$ is the exchange operator and is defined by

$$\hat{P}_{ij} f(q_1, \ldots, q_i, \ldots, q_j, \ldots, q_n) = f(q_1, \ldots, q_j, \ldots, q_i, \ldots, q_n).$$

This requirement on the wavefunction is called the Pauli exclusion principle [162]. As we will see in the next section, there is a simple way to construct wavefunctions that obey this antisymmetry requirement.

3.4.1 Slater Determinants

As we we have just seen, electrons occupy spin-orbitals that can be represented by the product of a spatial function and a spin function. We will denote these spin-orbitals by

$$\chi(q) = \psi(x, y, z)g(m_s)$$

where again $q = (x, y, z, m_s)$ is a vector of electron coordinates. Because we have two different values of spin (see Section 3.4), each spatial function $\psi(x, y, z)$ yields two different spin-orbitals, $\chi(q) = \psi(x, y, z)\alpha$ or $\chi(q) = \psi(x, y, z)\beta$.

With this in mind, we can construct an antisymmetric wavefunction by taking a linear combination of spin-orbitals via Slater determinants [195]. A Slater determinant is a determinant of a matrix whose entries are different spin-orbitals. All the elements in a given column of a Slater determinant use the same spin-orbital, whereas elements in the same row all involve the same electron [132]. For an n-electron system, the Slater determinant to construct a normalized wavefunction is defined as [206]

$$\psi(q_1, \ldots, q_n) = \frac{1}{\sqrt{n!}} \begin{vmatrix} \chi_1(q_1) & \chi_2(q_1) & \cdots & \chi_n(q_1) \\ \chi_1(q_2) & \chi_2(q_2) & \cdots & \chi_n(q_2) \\ \vdots & \vdots & & \vdots \\ \chi_1(q_n) & \chi_2(q_n) & \cdots & \chi_n(q_n) \end{vmatrix}. \tag{3.10}$$

One can easily check from the properties of determinants that this construction ensures that our wavefunction will be antisymmetric. Furthermore, if two electrons violated the Pauli exclusion principle by occupying the same spin-orbital, then the two corresponding columns of the matrix would be identical and make the determinant zero.

3.5 Calculating Energy

Although only the wavefunction and energy for a single-electron atoms can be solved for analytically, we must approximate solutions to more complicated systems. We will introduce two methods for doing so in this section. First, the variation method assumes a structure for the wavefunction that is dependent on certain parameters and then optimizes over those parameters. Second, perturbation theory splits the Hamiltonian operator into two parts: one that has

a known wavefunction and another that slightly increases the complexity of the Hamiltonian.

Before we proceed we will introduce bracket notation. Following notation from [132], for any operator $\hat{A}$ and any two, possibly complex, functions f_m and f_n we will use the abbreviation

$$\int f_m^* \hat{A} f_n d\tau = \left\langle f_m | \hat{A} | f_n \right\rangle$$

where f_m^* is the complex conjugate of f_m. In the absence of an operator this notation denotes the standard inner product

$$\int f_m^* f_n d\tau = \langle f_m | f_n \rangle .$$

Finally, a linear operator $\hat{A}$ is said to be a Hermitian operator if it satisfies

$$\left\langle f_m | \hat{A} | f_n \right\rangle = \left\langle f_n | \hat{A} | f_m \right\rangle^* . \tag{3.11}$$

Hamiltonian operators for quantum molecular systems are Hermitian operators. This fact will be useful in proving important results in the following subsections.

3.5.1 Variation Method

The first method we will discuss is the variation method. The method seeks to find an optimal wavefunction from a class of trial functions of our choosing. These trial functions usually have the same form but vary continuously with respect to one or more parameters. Before we outline the method, we will state and prove the Variation Theorem, an important result that validates the variation method as a reasonable method for approximating our solution.

Theorem 3.5.1.1. *Variation Theorem. Let $\hat{H}$ be a time-independent Hamiltonian with lowest eigenvalue E_1. Then for any wavefunction ϕ,*

$$W = \frac{\left\langle \phi | \hat{H} | \phi \right\rangle}{\langle \phi | \phi \rangle} \geq E_1. \tag{3.12}$$

Proof. We will prove the theorem for the case that ϕ is normalized. If ϕ is not normalized we can multiply ϕ by the appropriate constant to normalize it. Let $\{\psi_i\}$ be the complete set of normalized eigenfunctions for $\hat{H}$. Expand our wavefunction ϕ in terms of our complete set of eigenfunctions

$$\phi = \sum_i c_i \psi_i. \tag{3.13}$$

Note that since ϕ is normalized we have

$$\begin{aligned} 1 &= \langle\phi|\phi\rangle = \left\langle \sum_i c_i\psi_i \middle| \sum_j c_j\psi_j \right\rangle \\ &= \sum_i \sum_j c_i c_j \langle\psi_i|\psi_j\rangle \\ &= \sum_i |c_i|^2. \end{aligned}$$

Then plugging Equation 3.13 into the left hand side of Equation 3.12 yields

$$\begin{aligned} \left\langle \sum_i c_i\psi_i \middle| \hat{H} \middle| \sum_j c_j\psi_j \right\rangle &= \left\langle \sum_i c_i\psi_i \middle| \sum_j c_j\hat{H}\psi_j \right\rangle \\ &= \left\langle \sum_i c_i\psi_i \middle| \sum_j c_j E_j\psi_j \right\rangle \\ &= \sum_i \sum_j c_i^* E_j c_j \langle\psi_i|\psi_j\rangle \\ &= \sum_i |c_i|^2 E_i \\ &\geq \sum_i |c_i|^2 E_1 = E_1 \end{aligned}$$

since $E_1 \leq E_i$ for all $i = 2, 3, \ldots$. □

We must be sure that our class of trial functions ϕ satisfy any boundary conditions and properties that we would like our solution to satisfy. The simplest use of the Variation Theorem is to make ϕ depend on a parameter, α, and then minimize

$$W(\alpha) = \frac{\left\langle \phi_\alpha|\hat{H}|\phi_\alpha \right\rangle}{\langle\phi_\alpha|\phi_\alpha\rangle}$$

by solving $\frac{d}{d\alpha}W(\alpha) = 0$, but a common choice for the form of the test function is a linear combination of linearly independent basis functions $\{f_i\}$,

$$\phi = \sum_{i=1}^{n} c_i f_i.$$

This is called the linear variation method [132]. Our goal is to solve for the coefficients c_j.

Letting $c = (c_1, \ldots, c_n)$ first note that

$$W(c) = \frac{\left\langle \phi \left| \hat{H} \right| \phi \right\rangle}{\langle \phi | \phi \rangle} = \frac{\sum_{j=1}^{n} \sum_{k=1}^{n} c_j c_k H_{jk}}{\sum_{j=1}^{n} \sum_{k=1}^{n} c_j c_k S_{jk}}$$
$$\Rightarrow W(c) \sum_{j=1}^{n} \sum_{k=1}^{n} c_j c_k S_{jk} = \sum_{j=1}^{n} \sum_{k=1}^{n} c_j c_k H_{jk}$$

where $S_{jk} = \langle f_j | f_k \rangle$ and $H_{jk} = \left\langle f_j \left| \hat{H} \right| f_k \right\rangle$. Differentiating both sides with respect to c_i yields

$$\begin{aligned}
\frac{\partial W}{\partial c_i} \sum_{j=1}^{n} \sum_{k=1}^{n} c_j c_k S_{jk} + W \frac{\partial}{\partial c_i} \left(\sum_{j=1}^{n} \sum_{k=1}^{n} c_j c_k S_{jk} \right) &= \frac{\partial}{\partial c_i} \left(\sum_{j=1}^{n} \sum_{k=1}^{n} c_j c_k H_{jk} \right) \\
\Rightarrow \frac{\partial W}{\partial c_i} \sum_{j=1}^{n} \sum_{k=1}^{n} c_j c_k S_{jk} + W \frac{\partial}{\partial c_i} \left(\sum_{j=1}^{n} \sum_{k=1}^{n} c_j c_k S_{jk} \right) &= \frac{\partial}{\partial c_i} \left(\sum_{j=1}^{n} \sum_{k=1}^{n} c_j c_k H_{jk} \right) \\
\Rightarrow \frac{\partial W}{\partial c_i} \sum_{j=1}^{n} \sum_{k=1}^{n} c_j c_k S_{jk} + 2W \sum_{k=1}^{n} c_k S_{ik} &= 2 \sum_{k=1}^{n} c_k H_{ik} \\
\Rightarrow \frac{\partial W}{\partial c_i} \sum_{j=1}^{n} \sum_{k=1}^{n} c_j c_k S_{jk} &= 2 \sum_{k=1}^{n} [H_{ik} - W S_{ik}] c_k.
\end{aligned}$$

Setting $\frac{\partial W}{\partial c_i} = 0$ for $i = 1, \ldots, n$ we conclude that

$$\sum_{k=1}^{n} [H_{ik} - W S_{ik}] c_k = 0,$$

a system of linear homogeneous equations. For there to be a nontrivial solution ($c_i \neq 0$ for some i), we must ensure

$$\det(H_{ik} - S_{ik} W) = 0. \tag{3.14}$$

Equation 3.14 is called the *secular equation.*

To solve this system we expand the determinant and obtain an algebraic equation in terms of W of degree n with coefficients α_i

$$\alpha_0 + \alpha_1 W + \ldots + \alpha_n W^n = 0.$$

Such an equation has n real roots, $W_1, \ldots W_n$, and we can arrange them such that $W_1 \leq \ldots \leq W_n$. One can show that $W_i \geq E_i$ for any $i = 1, \ldots, n$. In this way we can approximate energies for ground and excited states of molecules. Note, though, that the Variation Theorem only

guarantees that our approximation is greater than or equal to the true energy, it gives no upper bound on the error of our approximation.

We will now demonstrate the variation method by returning to the particle in a box problem.

Example 3.5.1.1. *Consider again the particle in a box problem with $V(x) = 0$ for $0 \leq x \leq a$. We will attempt to approximate the wavefunction with*

$$\psi(x) = x(a - x) \text{ for } 0 \leq x \leq a.$$

For our choice of trial function Equation 3.12 yields

$$\begin{aligned} W &= \frac{\left\langle \psi \left| \hat{H} \right| \psi \right\rangle}{\langle \psi | \psi \rangle} \\ &= \frac{\int_0^a x(a-x) \frac{-\hbar^2}{2m} \frac{d^2}{dx^2} x(a-x) dx}{\int_0^a x^2 (a-x)^2 dx} \\ &= \frac{30\hbar^2 a^3}{6ma^5} \\ &= \frac{5\hbar^2}{ma^2} \approx 0.1266515 \frac{h^2}{ma^2}. \end{aligned}$$

The variation theorem guarantees that our approximated energy is greater than or equal to the lowest eigenvalue E_1 of the particle in a box Hamiltonian. Returning to the particle in a box example, we find that $E_1 = h^2/8ma^2 \approx 0.125h^2/ma^2$, thus we have successfully approximated the energy to within 1.3% error.

3.5.2 Perturbation Theory

The second method used to approximate the energy of a system is called perturbation theory. Again, our goal is to solve the time-independent Schrödinger equation

$$\hat{H}\psi_n = E_n \psi_n \tag{3.15}$$

by finding the eigenfunctions ψ_n and eigenvalues E_n. For perturbation theory we will assume that our Hamiltonian $\hat{H}$ is slightly different from a Hamilton $\hat{H}^0$ of an equation we can solve analytically,

$$\hat{H}^0 \psi_n^{(0)} = E_n^{(0)} \psi_n^{(0)}, \tag{3.16}$$

which we will refer to as the unperturbed system. We will assume that the eigenvalues of the unperturbed problem are nondegenerate, that is, $E_n^{(0)} \neq E_m^{(0)}$ for all $n \neq m$. So now our system

in Equation 3.15 only differs from Equation 3.16 by a perturbation, $\hat{H}'$. That is,

$$\hat{H} = \hat{H}^0 + \hat{H}'. \tag{3.17}$$

For perturbation theory we insert a parameter $\lambda \in [0, 1]$ into Equation 3.17 to get

$$\hat{H} = \hat{H}^0 + \lambda\hat{H}'. \tag{3.18}$$

By doing this we can vary the perturbation continuously from unperturbed ($\lambda = 0$) to fully perturbed ($\lambda = 1$). The idea of perturbation theory is to approximate a solution to Equation 3.15 by relating ψ_n and E_n to our known eigenpairs from Equation 3.16.

First we expand our wavefunction, ψ_n, and energy, E_n, in a power series about λ and evaluate at $\lambda = 0$. Letting $\psi_n^{(k)} = \frac{1}{k!}\frac{\partial^k \psi_n}{\partial \lambda^k}|_{\lambda=0}$ and $E_n^{(k)} = \frac{1}{k!}\frac{\partial^k E_n}{\partial \lambda^k}|_{\lambda=0}$, this yields

$$\psi_n = \psi_n^{(0)} + \lambda\psi_n^{(1)} + \lambda^2\psi_n^{(2)} + \dots \tag{3.19}$$

and

$$E_n = E_n^{(0)} + \lambda E_n^{(1)} + \lambda^2 E_n^{(2)} + \dots. \tag{3.20}$$

We assume the perturbation is small and the two series converge. Note that ψ_n and E_n are unknown, so we will use a nearby problem whose solution is known to approximate the wavefunction and energy.

Assume that we have orthonormal wavefunctions $\{\psi_n^0\}$ and corresponding energies $\{E_n^0\}$ to the zeroth order Schrödinger equation

$$\hat{H}^0\psi_n^0 = E_n^0\psi_n^0.$$

Further assume that the ψ_n satisfy the intermediate normalization condition

$$\left\langle \psi_n^{(0)} | \psi_n \right\rangle = 1,$$

which with Equation 3.19 implies that

$$\left\langle \psi_n^{(0)} | \psi_n^{(i)} \right\rangle = 0$$

for any $i \geq 1$. Now, substituting Equations 3.18, 3.19 and 3.20 into Equation 3.1, we have

$$\left(\hat{H}^0 + \lambda\hat{H}'\right)\left(\psi_n^{(0)} + \lambda\psi_n^{(1)} + \lambda^2\psi_n^{(2)} + \dots\right) = \left(E_n^0 + \lambda E_n^{(1)} + \lambda^2 E_n^{(2)} + \dots\right)\left(\psi_n^{(0)} + \lambda\psi_n^{(1)} + \lambda^2\psi_n^{(2)} + \dots\right).$$

Operating and combining terms with like powers of λ yields

$$\begin{aligned} \hat{H}^0\psi_n^{(0)} &= E_n^0\psi_n^{(0)}, \\ \hat{H}^0\psi_n^{(1)} + \hat{H}'\psi_n^{(0)} &= E_n^{(0)}\psi_n^{(1)} + E_n^{(1)}\psi_n^{(0)}, \\ &\vdots \end{aligned} \tag{3.21}$$

We now multiply Equation 3.21 by $\psi_m^{(0)*}$, rearrange and integrate to get

$$\left\langle \psi_m^{(0)} \left| \hat{H}^0 \right| \psi_n^{(1)} \right\rangle - E_n^{(0)} \left\langle \psi_m^{(0)} | \psi_n^{(1)} \right\rangle = E_n^{(1)} \left\langle \psi_m^{(0)} | \psi_n^{(0)} \right\rangle - \left\langle \psi_m^{(0)} \left| \hat{H}' \right| \psi_n^{(0)} \right\rangle. \tag{3.22}$$

The fact that $\hat{H}^0$ is a Hermitian operator allows us to rewrite the first term of Equation 3.22 as

$$\left\langle \psi_m^{(0)} \left| \hat{H}^0 \right| \psi_n^{(1)} \right\rangle = E_m^{(0)} \left\langle \psi_m^{(0)} | \psi_n^{(1)} \right\rangle.$$

This implies

$$\left(E_m^{(0)} - E_n^{(0)}\right) \left\langle \psi_m^{(0)} | \psi_n^{(1)} \right\rangle = E_n^{(1)}\delta_{mn} - \left\langle \psi_m^{(0)} \left| \hat{H}' \right| \psi_n^{(0)} \right\rangle \tag{3.23}$$

and thus

$$E_n^{(1)} = \left\langle \psi_n^{(0)} \left| \hat{H}' \right| \psi_n^{(0)} \right\rangle.$$

To find $\psi_n^{(1)}$ we use our set of known solutions $\left\{\psi_n^{(0)}\right\}$ to expand

$$\psi_n^{(1)} = \sum_m a_m \psi_m^{(0)}$$

where $a_m = \left\langle \psi_m^0 | \psi_n^{(1)} \right\rangle$. For $m \neq n$ Equation 3.23 becomes

$$\left(E_m^{(0)} - E_n^{(0)}\right) \left\langle \psi_m^{(0)} | \psi_n^{(1)} \right\rangle = - \left\langle \psi_m^{(0)} \left| \hat{H}' \right| \psi_n^{(0)} \right\rangle.$$

Substituting in our expansion for $\psi_n^{(1)}$ yields

$$\left(E_m^{(0)} - E_n^{(0)}\right) a_m = - \left\langle \psi_m^{(0)} \left| \hat{H}' \right| \psi_n^{(0)} \right\rangle.$$

Since we assumed that the energy levels were nondegenerate we are assured that $E_m^{(0)} - E_n^{(0)} \neq 0$ and we can safely write

$$a_m = \frac{\left\langle \psi_m^{(0)} \left| \hat{H}' \right| \psi_n^{(0)} \right\rangle}{E_n^{(0)} - E_m^{(0)}}.$$

Thus the first-order correction to the wavefunction is

$$\psi_n^{(1)} = \sum_{m \neq n} \frac{\left\langle \psi_m^{(0)} \middle| \hat{H}' \middle| \psi_n^{(0)} \right\rangle}{E_n^{(0)} - E_m^{(0)}} \psi_m^{(0)}$$

and

$$\psi_n \approx \psi_n^{(0)} + \sum_{m \neq n} \frac{\left\langle \psi_m^{(0)} \middle| \hat{H}' \middle| \psi_n^{(0)} \right\rangle}{E_n^{(0)} - E_m^{(0)}} \psi_m^{(0)}.$$

Degenerate levels of energy require some special care but are not discussed here. It should also be noted that it is possible to solve for higher-order energy and wavefunction corrections but the process is tedious. Since we assume our perturbations are small, it is usually sufficient to include only the first order corrections.

3.5.3 Hartree-Fock Self-Consistent Field Method

The Hartree-Fock method [177] is an application of the variation method which separates the inner product $\left\langle \psi \middle| \hat{H} \middle| \psi \right\rangle$ by electronic interactions. We will first introduce the Hartree self-consistent-field (SCF) method as discussed in detail by Blinder in [22].

Recall that the Hamiltonian operator for an n-electron atom is

$$\hat{H} = -\frac{\hbar^2}{2m_e} \sum_{i=1}^{n} \nabla_i^2 - \sum_{i=1}^{n} \frac{Ze'^2}{r_i} + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{e'^2}{r_{ij}}$$

where the first sum is the kinetic energy operators, the second sum is the potential energy for the attractions between the electrons and the nucleus of charge Ze' and the last sum is the potential energy of the inter-electronic repulsions. Note that this Hamiltonian is still incomplete since we do not account for spin-orbit and other interactions. Applying the perturbation method yields a zeroth-order wavefunction that is a product of n one-electron spin-orbitals

$$\psi^{(0)} = \prod_{i=1}^{n} f_i(r_i, \theta_i, \phi_i)$$

where each f is a hydrogen-like orbital of the form of Equation 3.8. This wavefunction, however, is quantitatively inaccurate for complex systems [132]. We can get a better approximation by using a variation function without restricting ourselves to any particular form of orbitals. We can take

$$\phi = \prod_{i=1}^{n} g_i(r_i, \theta_i, \phi_i)$$

and look for the functions g_i that minimize the variational integral given in Equation 3.12.

The method for determining the g_i's is called the Hartree self-consistent-field method and is outlined as follows. First, we guess a product wavefunction

$$\phi_0 = \prod_{i=1}^{n} s_i(r_i, \theta_i, \phi_i)$$

where each s_i is a normalized function $R(r)$ multiplied by a spherical harmonic $Y(\theta, \phi)$ (see Section 3.3). We now average out the instantaneous interactions between electron 1 and all other electrons, which we assume are homogonized to form a fixed distribution of electric charge through which electron 1 moves. The potential energy for such an interaction is

$$\bar{V}_1(r_1, \theta_1, \phi_1) = \sum_{j=2}^{n} e'^2 \int \frac{|s_j|^2}{r_{1j}} dv_j - \frac{Ze'^2}{r_1}.$$

We also make use of the central-field approximation, which assumes that the effective potential acting on an electron in an atom can be adequately approximated by a function of r only. With this assumption, the resulting potential energy of this interaction is given by [132] to be

$$V_1(r_1) = \frac{\int_0^{2\pi} \int_0^{\pi} \bar{V}_1(r_1, \theta_1, \phi_1) \sin\theta_1 d\theta_1 d\phi_1}{\int_0^{2\pi} \int_0^{\pi} \sin\theta d\theta d\phi}.$$

We then plug this potential energy into a one-electron Schrödinger equation

$$\left[-\frac{\hbar^2}{2m_e} \nabla_1^2 + V_1(r_r) \right] t_1(1) = \epsilon_1 t_1(1) \tag{3.24}$$

and solve for $t_1(1)$, an improved orbital for electron 1. We get a set of solutions that are products of a spherical harmonic and radial factor, and we choose the solution that corresponds to the orbital we are improving. We then iterate through the rest of the n electrons, regarding them as moving in a charge cloud density due to the other electrons and use the improved orbitals as we go. Once we have repeated this process for all n electrons, we repeat the entire process until we are satisfied with the quality of our orbitals. In the end we will have orbitals $g_i(i)$ and energies ϵ_i. The energy approximation for the system is

$$E = \sum_i \epsilon_i - \sum_i \sum_{j>i} \left\langle g_i(i) g_i(i) \left| \frac{e'^2}{r_{ij}} \right| g_j(j) g_j(j) \right\rangle.$$

Recalling that Hartree's method does not account for spin, The Hartree-Fock SCF method is similar to the Hartree SCF method but uses antisymmetrized spin-orbitals via Slater deter-

minants (see Section 3.4.1) to include spin explicitly in the wavefunction. The method solves

$$\hat{F}u_i = \epsilon_i u_i$$

where u_i is the i^{th} spin-orbital and the Fock operator $\hat{F}$ is similar to the operator in Equation 3.24 but includes some extra terms. ϵ_i is the orbital energy of u_i. As proposed by Roothaan in [177], the Hartree-Fock orbitals are written as linear combinations of a set $\{\chi_j\}$ of basis functions

$$g_i = \sum_j c_{ji}\chi_j$$

where the c_{ji}'s are found by the SCF iterative process and minimize the energy of the system. A Hartree-Fock SCF calculation for a many-electron atom requires a great deal of computation, but one can get very accurate results with astutely chosen basis functions, the subject of the next section.

3.5.4 Basis Sets

In performing ab initio computational chemistry calculations the choice of basis set plays a significant role in the accuracy of the results. Recalling from the previous section that molecular orbitals g_i are represented as a linear combination of basis functions $g_i = \sum_j c_{ji}\chi_j$, in this section we describe some popular choices for the basis set $\{\chi_j\}$. As we will see, an understanding of the molecular system in question is the first step towards choosing an appropriate basis set. Two large families of basis sets are built off of the so-called Slater-type orbitals (STOs) and Gaussian-type functions (GTFs).

Originally proposed by Slater in 1930, an STO centered on atom a has the form

$$\chi^{STO} = N r_a^{n-1} e^{-\zeta r_a} Y_l^m(\theta_a, \phi_a) \tag{3.25}$$

with quantum numbers n, l, and m [196]. The function $Y_l^m(\theta_a, \phi_a)$ is a spherical harmonic as defined in Equation 3.3, the parameter ζ is called the orbital exponent, and N is a normalization constant. The simplest of STO basis sets is called the minimal basis set and consists of one STO for each inner shell and valence shell atomic orbital of each atom. The size of the basis set can be increased by increasing the number of STO's per atomic orbital. For example, a double-zeta basis set refers to a minimal basis set that replaces each STO by two STO's that have different orbital exponents, and analogously a triple-zeta basis replaces each STO by three STO's with different orbital exponents.

Valence shell electrons, or electrons in the outermost shell, play an important role in molecular bonding, so one may wish to use more basis functions to describe these orbitals. A split-

valence basis set uses two or more STO's for each valence atomic orbital but only one STO for each inner orbital. To account for atomic orbital polarization, basis functions whose l quantum numbers are greater than the maximum l of the valence shell can be added to the basis set. Adding polarization functions allows the atomic orbitals to vary and shifts the charge density away from the nuclei and into the bonding regions within the molecule [132]. For situations where there is a significant electron density at large distances from nuclei, one can add diffuse functions, or functions with a very small orbital exponent, to the basis set. SCF calculations on moderately sized molecules can use anywhere to 40 to 400 basis functions, which consequently produce 300000 to 3×10^9 electron-repulsion integrals, the evaluations of which are very time-consuming [132].

The other family of basis sets use Gaussian-type functions as their building block, and were first proposed by Boys in 1950 to speed up molecular integral evaluation [29]. A cartesian Gaussian, or primitive Gaussian, centered on atom a at (x_a, y_a, z_a) is defined as

$$\chi^{PG} = N x_a^i y_a^j z_a^k e^{-\alpha r_a^2} \tag{3.26}$$

where $i, j, k \in \mathbb{Z}^+$, α is a positive orbital exponent, and N is again a normalizing constant. Since a single Gaussian function gives a poor representation of an atomic orbital for small values of r_a, in practice one forms each basis function by taking a normalized linear combination of GTFs. Such a basis function is called a contracted Gaussian-type function (CGTF) and has the form

$$\chi^{CG} = \sum_i \beta_i \chi_i^{PG}. \tag{3.27}$$

Creating a basis set in this way increases the number of integral evaluations during the SCF procedure, but Gaussian integrals require much less computational cost than Slater integrals [132]. Usually, each χ_i^{PG} is held fixed and only the β_i's are varied during the SCF variational procedure, reducing the number of parameters to be optimized and thus further reducing the computational cost. Furthermore, CGTF basis sets can be modified in the same vein as STO basis sets and are the most widely used for ab initio molecular calculations because of they are less computationally expensive than STO basis sets.

This work makes frequent use of the 6-31G split valence basis set, which uses six primitive Gaussians in each inner-shell CGTF and uses one CGTF with three primitives and one Gaussian with one primitive in each valence-shell orbital. This basis set is named using notation from [56], where split-valence basis sets are denoted X-YZG. X is the number of primitive Gaussians used for each inner-shell CGTF. The valence-shell orbitals are described by two CGTF functions made with Y and Z primitive Guassians, respectively. The 6-31G basis set can be modified to include polarization functions on heavy atoms by adding " * " and diffusion functions on heavy

atoms by adding " + ". For example, the 6-31+G* basis set is the 6-31G basis set supplemented by both diffuse and polarization functions. Two asterisks or two plus signs denote that the corresponding functions are also added to the light atoms: hydrogen and helium. Clearly, larger basis sets yield more accurate results but take more computation time. There is a vast number of basis sets to choose from when performing ab initio calculations. Many of the most commonly used basis sets and their corresponding literature references can be found at the Environmental Molecular Sciences Laboratory Gaussian Basis Set Exchange website (bse.pnl.gov/bse/portal).

3.5.5 Density Functional Theory

We now introduce a completely different method to calculate energy that is based on the fact that all information about a given system can be described by the electron density. For a system with N electrons, the electron density, $\rho(\bar{r})$, is defined as

$$\rho(\bar{r}) = N \int \dots \int |\Psi(\bar{x}_1, \dots, \bar{x}_N)|^2 \, ds_1 d\bar{x}_2 \dots d\bar{x}_N \tag{3.28}$$

and determines the probability of finding any of the N electrons with arbitrary spin within the specified volume $d\bar{r}_1$ while the other $N-1$ electrons have arbitrary positions and spin in the state represented by Ψ.

Density functional theory (DFT) was given a firm theoretical foundation by Hohenberg and Kohn in 1964 [107], in which they prove two theorems that lay the foundation for the theory. The first Hohenberg-Kohn theorem states that the external potential energy is, to within a constant, a unique functional of the electron density. Furthermore, this potential energy uniquely determines the Hamiltonian operator and thus all properties of the system. This allows one to write the ground state energy as a functional of the true electron density ρ_0,

$$\begin{aligned} E_0[\rho_0] &= T[\rho_0] + V_{ee}[\rho_0] + V_{Ne}[\rho_0] \\ &= T[\rho_0] + E_{ee}[\rho_0] + \int \rho_0(\bar{r}) V_{Ne} d\bar{r} \end{aligned}$$

where we split the energy into three parts: kinetic energy, electron-electron potential energy, and nuclei-electron potential energy. From this we define the Hohenberg-Kohn functional $F_{HK}[\rho_0]$ and write

$$E_0[\rho_0] = F_{HK}[\rho_0] + \int \rho_0(\bar{r}) V_{Ne} d\bar{r}. \tag{3.29}$$

It is important to note that the exact form of this functional that produces the correct ground state energy has not been determined. However, it does exist, and advances in density functional theory rely on developing functionals that satisfactorily approximate the true functional.

The second Hohenberg-Kohn theorem states that $F_{HK}[\rho]$ gives the lowest energy if and

only if the input density is the true ground state density, ρ_0. Recalling the variation principle introduced in Section 3.5.1, this means that for any trial density $\tilde{\rho}(\bar{r})$ the energy obtained from the functional in Equation 3.29 is an upper bound to the true ground state energy E_0. This theorem only applies to the exact functional, and since we are forced to approximate this functional, one should be careful interpreting energies calculated from density functional theory methods. In the Hartree-Fock approximation the variation principle holds for all wavefunctions, but this is not the case for density functional theory and all functionals.

The Kohn-Sham Method

In 1965 Kohn and Sham developed a method for finding the density ρ_0 and ground state energy E_0 in [126]. They employ what is called a noninteracting system, often denoted by the subscript s, of N noninteracting electrons that each experience the same external potential energy function $V_s(\bar{r}_i)$. This potential energy function is chosen to make the ground state electron probability density $\rho_s(\bar{r})$ of this fictitious system equal to the exact ground state electron density $\rho_0(\bar{r})$. Using notation from [125], the Hamiltonian for the noninteracting system is

$$\hat{H}_s = \sum_{i=1}^{N} \left[-\frac{1}{2}\nabla_i^2 + v_s(\bar{r}_i) \right] = \sum_{i=1}^{N} \hat{h}_i^{KS}. \tag{3.30}$$

The Kohn-Sham reference system can be related to the real system via

$$\hat{H}_\lambda = \hat{T} + \sum_{i=1}^{N} v_\lambda(\bar{r}_i) + \lambda \hat{V}_{ee} \tag{3.31}$$

where the $\lambda \in [0, 1]$ and v_λ is defined as the external potential that will make the ground state electron density of the λ-system equal to that of the real molecule's ground state [125]. Note that for $\lambda = 0$ we have the noninteracting system and for $\lambda = 1$ we have the real system. The ground state wave function $\psi_{s,0}$ is the Slater determinant of the Kohn-Sham spin orbitals $u_i^{KS} = \theta_i^{KS}\sigma_i$ and each spatial part θ_i^{KS} satisfies

$$\hat{h}_i^{KS}\theta_i^{KS} = \epsilon_i^{KS}\theta_i^{KS} \tag{3.32}$$

where the ϵ_i^{KS}'s are the Kohn-Sham orbital energies.

For simplicity we shall henceforth drop the subscript from ρ_0 and refer to the exact electron density as ρ. Kohn and Sham defined the functionals

$$\Delta T[\rho] = T[\rho] - T_s[\rho] \tag{3.33}$$

and

$$\Delta V_{ee}[\rho] = V_{ee}[\rho] - \frac{1}{2}\int\int \frac{\rho(\bar{r}_1)\rho(\bar{r}_2)}{r_{12}} d\bar{r}_1 d\bar{r}_2 \tag{3.34}$$

to quantify the differences in the average ground state electronic kinetic energy and electron-electron potential energy between the molecule and the reference system of noninteracting electrons. The sum of these two functionals is called the exchange-correlation energy functional and is defined as

$$E_{xc}[\rho] = \Delta T[\rho] + \Delta V_{ee}[\rho]. \tag{3.35}$$

We can now rewrite Equation 3.29 as

$$E_0[\rho] = \int \rho(\bar{r}) V_{Ne}(\bar{r}) d\bar{r} + T_s[\rho] + \frac{1}{2}\int\int \frac{\rho(\bar{r}_1)\rho(\bar{r}_2)}{r_{12}} d\bar{r}_1 d\bar{r}_2 + E_{xc}[\rho]. \tag{3.36}$$

The first three terms of Equation 3.36 are relatively easy to evaluate given the density ρ. The exchange-correlation term $E_{xc}[\rho]$, however, is not and is the key to accurate Kohn-Sham DFT methods [132]. The true form of $E_{xc}[\rho]$ is not known and various approximations have been proposed (see [160] for details). In this work we make use of the popular B3LYP (Becke, three-parameter, Lee-Yang-Parr) method which uses a combination of functionals from Becke [16] and Lee, Yang, and Parr [130].

Since the noninteracting system is defined to have the same electron density as the ground state of the molecule, we have that $\rho_s = \rho$. Thus, as shown in [132], the ground state electron density can be expressed can be expressed in terms of the Kohn-Sham spatial orbitals as

$$\rho = \sum_{i=1}^{N} |\theta_i^{KS}|^2. \tag{3.37}$$

The Kohn-Sham orbitals are found via a process similar to the method used to find the Hartree-Fock orbitals [160]. Once we have found the appropriate Kohn-Sham orbitals and a sufficient approximation to the exchange-correlation energy functional, we can calculate our approximation to the ground state energy E_0. One can calculate excited state energies via the time-dependent DFT method, a method first proposed by Runge and Gross in [178]. Finally, we note that we have only scratched the surface of density functional theory, as it is the subject of current and ongoing research. Excellent reviews of the theory can be found in [160] and [125].

3.6 Geometry Optimization

The equilibrium geometry of a molecule corresponds to the nuclear arrangement that minimizes the molecular electronic potential energy. Using the methods presented in the previous Section,

we can now compute the energy of a molecule for any molecular geometry with coordinates p. A molecule will conform to minimize its energy in any quantum state n, thus

$$\hat{E}_n = \min_p E_n(p).$$

This optimization can be performed with the computational chemistry software Gaussian 09 [72] using the Berny optimization algorithm, which is based on an algorithm developed by Schlegel in [182]. Schlegel's algorithm is a variation of Pulay mixing [171], a quasi-Newton method that constructs the consecutive iterates using a linear combination of potential solutions. Quasi-Newton methods are Newton optimization methods that use an approximate Hessian H_c instead of an analytic one, and updates H_c as the iterative optimization proceeds. At each iterate x_c quasi-Newton methods

1. Compute the Newton step $d = -H_c^{-1}\nabla f(x_c)$,
2. Compute the next iterate $x_+ = x_c + \lambda d$, where λ is a step length parameter,
3. Update H_c to H_+.

Newton methods can be difficult to implement because of the unavailability or cost of a gradient $\nabla f(x_c)$ and/or Hessian H_c. For $E_n(p)$, however, it is possible to compute an analytic gradient, $\nabla E_n(p)$, and even an analytic Hessian. In the following subsection, we derive an expression for the gradient of the energy. While analytic Hessians are possible, the default option within the Berny optimization algorithm is to use a finite difference Hessian [72].

3.6.1 Energy Gradient

One of the greatest advances in computational chemistry was the analytic expression for the gradient of the energy of a system, which can be obtained via the Hellmann-Feynman theorem [105, 69]. Before deriving the analytic expression for the gradient, we present and prove the Hellman-Feynman theorem.

Theorem 3.6.1.1. *Hellman-Feynman Theorem: For any energy state n,*

$$\frac{\partial E_n}{\partial \lambda} = \left\langle \psi_n(\lambda) \left| \frac{\partial \hat{H}}{\partial \lambda} \right| \psi_n(\lambda) \right\rangle \tag{3.38}$$

where ψ_n, E_n and $\hat{H}$ all depend parametrically on λ.

Proof. Consider the Schrödinger equation with normalized eigenfunctions. Because of normal-

ization and assuming well-behavedness of our integrand,

$$\begin{aligned} E_n &= \int \psi_n^* \hat{H} \psi_n d\tau \\ \Rightarrow \frac{\partial E_n}{\partial \lambda} &= \frac{\partial}{\partial \lambda} \int \psi_n^* \hat{H} \psi_n d\tau \\ &= \int \frac{\partial \psi_n^*}{\partial \lambda} \hat{H} \psi_n d\tau + \int \psi_n^* \frac{\partial}{\partial \lambda} \left(\hat{H} \psi_n \right) d\tau. \end{aligned}$$

Noting that

$$\frac{\partial}{\partial \lambda} \left(\hat{H} \psi_n \right) = \frac{\partial \hat{H}}{\partial \lambda} \psi_n + \hat{H} \frac{\partial \psi_n}{\partial \lambda},$$

we can write

$$\frac{\partial E_n}{\partial \lambda} = \int \frac{\partial \psi_n^*}{\partial \lambda} \hat{H} \psi_n d\tau + \int \psi_n^* \frac{\partial \hat{H}}{\partial \lambda} \psi_n d\tau + \int \psi_n^* \hat{H} \frac{\partial \psi_n}{\partial \lambda} d\tau.$$

Evaluating the first term using the Schrödinger equation and the fact that the Hamiltonian is Hermitian yields

$$\begin{aligned} \frac{\partial E_n}{\partial \lambda} &= E \int \frac{\partial \psi_n^*}{\partial \lambda} \psi_n d\tau + \int \psi_n^* \frac{\partial \hat{H}}{\partial \lambda} \psi_n d\tau + \int \frac{\partial \psi_n}{\partial \lambda} \left(\hat{H} \psi_n \right)^* d\tau \\ &= E \int \frac{\partial \psi_n^*}{\partial \lambda} \psi_n d\tau + \int \psi_n^* \frac{\partial \hat{H}}{\partial \lambda} \psi_n d\tau + E \int \psi_n^* \frac{\partial \psi_n}{\partial \lambda} d\tau \\ &= E \int \left(\frac{\partial \psi_n^*}{\partial \lambda} \psi_n + \psi_n^* \frac{\partial \psi_n}{\partial \lambda} \right) d\tau + \int \psi_n^* \frac{\partial \hat{H}}{\partial \lambda} \psi_n d\tau \\ &= E \frac{\partial}{\partial \lambda} \int \psi_n^* \psi_n d\tau + \int \psi_n^* \frac{\partial \hat{H}}{\partial \lambda} \psi_n d\tau \\ &= 0 + \int \psi_n^* \frac{\partial \hat{H}}{\partial \lambda} \psi_n d\tau \end{aligned}$$

since

$$\begin{aligned} \int \psi_n^* \psi_d \tau &= 1 \\ \Rightarrow \frac{\partial}{\partial \lambda} \int \psi_n^* \psi_d \tau &= 0. \end{aligned}$$

□

To compute one component of the energy gradient we will take $\lambda = x_p$ to be one of the Cartesian coordinates specifying molecular geometry.

For a system of k atoms and n electrons the Hamiltonian is

$$\hat{H} = -\frac{\hbar}{2m}\sum_{i=1}^{n}\nabla_i^2 - \sum_{i=1}^{n}\sum_{j=1}^{k}\frac{Z_j}{r_{ij}} + \sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\frac{1}{r_{ij}} + \sum_{i=1}^{k-1}\sum_{j=i+1}^{k}\frac{Z_i Z_j}{r_{ij}}.$$

Differentiating with respect to λ yields

$$\frac{\partial\hat{H}}{\partial\lambda} = -\frac{\partial}{\partial\lambda}\frac{\hbar}{2m}\sum_{i=1}^{n}\nabla_i^2 - \frac{\partial}{\partial\lambda}\sum_{i=1}^{n}\sum_{j=1}^{k}\frac{Z_j}{r_{ij}} + \frac{\partial}{\partial\lambda}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\frac{1}{r_{ij}} + \frac{\partial}{\partial\lambda}\sum_{i=1}^{k-1}\sum_{j=i+1}^{k}\frac{Z_i Z_j}{r_{ij}}.$$

Since the first and third terms do not depend on λ we have

$$\begin{aligned}\frac{\partial\hat{H}}{\partial\lambda} &= -\frac{\partial}{\partial\lambda}\sum_{i=1}^{n}\sum_{j=1}^{k}\frac{Z_j}{r_{ij}} + \frac{\partial}{\partial\lambda}\sum_{i=1}^{k-1}\sum_{j=i+1}^{k}\frac{Z_i Z_j}{r_{ij}} \\ &= -Z_p\sum_{i=1}^{n}\frac{x_i - x_p}{r_{ip}^3} + Z_p\sum_{i\neq p}^{k}\frac{Z_i(x_i - x_p)}{r_{ij}^3},\end{aligned}$$

and thus we can write the partial derivative of the wavefunction with respect to λ as

$$\frac{\partial E_n}{\partial\lambda} = \left\langle \psi_n(\lambda)\left| -Z_p\sum_{i=1}^{n}\frac{x_i - x_p}{r_{ip}^3} + Z_p\sum_{i\neq p}^{k}\frac{Z_i(x_i - x_p)}{r_{ij}^3}\right|\psi_n(\lambda)\right\rangle.$$

Now that we have an expression for the gradient, we will outline Pulay mixing [171], the method that motivated the Berny optimization algorithm.

3.6.2 Pulay Mixing

At the heart of Gaussian 09's optimization algorithm is a method called direct inversion of the iterative subspace (DIIS), also known as Pulay mixing [171] and closely related to Anderson acceleration [4]. Our goal is to solve the optimization problem

$$\min E(p_1, \ldots, p_n)$$

with respect to parameters or coordinates p_k for $k = 1, \ldots, n$. Pulay mixing begins with a set of coordinates, $\bar{p}^1 = (p_1^1, \ldots, p_n^1)$ and an approximate inverse Hessian H_0^{-1}. Pulay originally developed this approach as an improvement over what he termed "simple relaxation" (SR),

which, starting from an initial iterate $\bar{p}^1$, generates the next iterate via

$$\bar{p}^i = \bar{p}^1 - \sum_{j=1}^{i-1} H_0^{-1} \nabla E(\bar{p}^j).$$

In [171] Pulay posited that a much better approximation could be constructed from the first m iterated vectors $\bar{p}^1, \ldots, \bar{p}^m$. These m vectors form the $(m+1)^{th}$ iterate by writing $\bar{p}^{m+1}$ as a linear combination of the previous m vectors,

$$\bar{p}^{m+1} = \sum_{i=1}^{m} c_i \bar{p}^i.$$

The c_i's are found by solving

$$\min_{c_i} \sum_{i=1}^{m} c_i \Delta \bar{p}^i$$

subject to $\sum_{i=1}^{m} c_i = 1$ where $\Delta \bar{p}^i = \bar{p}^{i+1} - \bar{p}^i$. This minimization problem is solved via the Lagrange multiplier technique with Lagrange constant λ, a value which yields the squared norm of the residuum vector $\Delta \bar{p}^{m+1}$. This results in a system of $m+1$ linear equations

$$\begin{pmatrix} B_{11} & B_{12} & \ldots & B_{1m} & -1 \\ B_{21} & B_{22} & \ldots & B_{2m} & -1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ B_{m1} & B_{m2} & \ldots & B_{mm} & -1 \\ -1 & -1 & \ldots & -1 & 0 \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_m \\ \lambda \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ -1 \end{pmatrix}$$

where $B_{ij} = \left\langle \Delta p^i | \Delta p^j \right\rangle$. Once $\bar{p}^{m+1}$ is obtained and the convergence criteria is tested by a step of SR. If convergence has not been reached, $\bar{p}^{m+1}$ is added to the list of vectors and the DIIS process is repeated.

Once the new iterate p_{k+1} is determined, the method proceeds with the Broyden-Fletcher-Goldfarb-Shanno (BFGS) update scheme [30, 70, 79, 188] for the new approximate Hessian

$$H_+ = H_c + \frac{yy^T}{y^T s} - \frac{(H_c s)(H_c s)^T}{s^T H_c s}$$

where $s = x_+ - x_c$ and $y = \nabla f(x_+) - f(x_c)$.

Gaussian 09 [72] currently utilizes the Berny geometry optimization algorithm [182] which takes advantage of redundant internal coordinates [172]. Gaussian gives the user an option to set constraints for the optimization process. In this work we often freeze, or fix, one or several internal coordinates. Such coordinates are referred to as frozen variables. According to [72],

each step of the current implementation for finding a minimum takes the following actions:

1. Update the Hessian via BFGS.
2. Update the trust radius using the method of Fletcher and Bofill from [23].
3. Set components of the gradient vector corresponding to frozen variables to zero. This eliminaties their contribution to the next optimization step.
    - Perform a polynomial line search to find the step length. The degree of the polynomial is determined principally by the availability of second derivatives.
4. If the latest point is the best so far, a quadratic step is determined using the Rational Function Optimization (RFO) approach [194, 8].
5. Set components of the quadratic step vector corresponding to frozen variables to zero.
6. If the step exceeds the trust radius, reduce the step in length to the trust radius by searching for a minimum of the quadratic function on the sphere having the trust radius [78].
7. Test for convergence.

Berny geometry optimizations use more convergence criteria than standard quasi-Newton methods which normally terminate at small gradient [135]. The four additional criteria are

1. The root-mean square of $\nabla E(\bar{p})$ is less than 10^{-5}
2. The largest component of $\nabla E(\bar{p})$ is less than 1.5×10^{-5}
3. The root-mean square of the geometry displacement is less than 4×10^{-5}
4. The maximum single displacement is less than 6×10^{-5}.

Chapter 4

Reaction Path Following

The potential energy surface (PES) of a molecule describes the energy of an N-atom molecule as a function of its $3N-6$ geometric coordinates [220]. Local minima of these surfaces correspond to stable molecular geometries, and first order saddle points correspond to transition states [20]. As such, there is great interest in studying the global structure of these PESs, especially in regards to following reaction paths from one stable molecular geometry to another. Reaction path following from transition states to local minima has been studied extensively [17, 63, 183, 104, 190, 20, 24], but current algorithms are computationally burdensome for molecules composed of more than a handful of atoms. Molecules of interest can have hundreds of atoms and degrees of freedom so model and/or dimension reduction must be applied to efficiently compute the reaction path. Because of the computational expense, most approaches for reaction path following do not explicitly construct the PESs.

For many reactions only a few of the $3N-6$ molecular coordinates change significantly and the rest remain approximately constant. As such, a popular approach for dimension reduction is to follow reaction paths for only a small subset of the $3N-6$ molecular coordinates. In [20], for example, Birkholz and Schlegel use principal component analysis to identify the reduced number of internal coordinates with which to define the reaction path.

A popular approach for model reduction involves iteratively optimizing a finite set of points, or images, on the PES to find a minimum energy path between a minimum and a transition state. Two methods that work in this way are nudged elastic band [116] and the string method [223].

The model reduction technique we are interested in constructs surrogate PESs by means of interpolation. Ischtwan and Collins, for example, use an inverse distance weighted method known as modified Shepard interpolation [47, 46] to represent the PES [109]. Spline interpolation has been used to construct PESs for $d \leq 3$ dimensions [169, 181, 145], and has recently been applied to higher dimensions with non-uniform meshes [161]. These interpolation methods,

however, are restricted to small molecules because of the computational cost of constructing the PESs to within a sufficient accuracy. The number of grid points required for interpolation on full grids grows exponentially with dimension, a problem known as the *curse of dimensionality*.

In this thesis we present a method for reaction path following based on the work of Mokrauer, Kelley, and Bykhovski [146, 144] that applies both model and dimension reduction by constructing a surrogate model via interpolation on *sparse* grids [198, 13, 118]. Mokrauer's approach isolates a few, say d, molecular coordinates and then uses sparse interpolation to approximate the PESs in patches during the reaction simulation using a combination of trust-region and error estimation algorithms [144]. The numerous interpolation patches, however, introduce discontinuities in the PES that are unphysical. The method presented in this thesis is similar, but constructs a single global PES and then simulates the reaction. In this way one can visualize the entire interpolated PES landscape before simulating the reaction process.

Furthermore, this new reaction path following method allows one to track entire reaction paths on the ground and all specified excited state PESs. Knowledge of excited state dynamics is particularly useful in studying reaction paths of photoisomerization processes, for example, in which a molecule's absorption of a photon causes it to excite to a different electronic state and consequently change between isomers. Isomers are molecules with the same molecular formula but with different geometric configurations and possibly different properties [61].

4.1 Surrogate Model

4.1.1 Potential Energy Surface Approximation

In this section we describe how we approximate PESs using Smolyak's sparse grid interpolation algorithm [13, 118, 198]. The potential energy E_n of an N-atom molecule at any electronic state $n = 0, 1, 2 \ldots$, can be computed as a function of (redundant) internal coordinates $p \in \mathbb{R}^{3N-6}$ composed of bond lengths, bond angles, and dihedral angles. The first step of our method is to partition

$$p = \begin{pmatrix} x \\ \xi \end{pmatrix}$$

into a vector of design variables $x \in \mathbb{R}^d$ and a vector of remainder variables $\xi \in \mathbb{R}^{3N-6-d}$, where chemical knowledge or intuition of the system guides the appropriate choice of design variables x. The d-dimensional ground state PES is computed via the constrained optimization problem

$$E_0(x) = \min_{\xi} \mathcal{E}_0(x, \xi) \tag{4.1}$$

where the minimization is only over the remainder variables ξ. Points on the excited state PESs $E_n(x)$, $n \geq 1$, are calculated at optimized ground state geometries. Because Equation 4.1 is such an expensive optimization process, continuous dynamical simulations are often unfeasible. Smolyak's algorithm [198, 13, 118], however, allows one to simulate dynamics with a cheap surrogate potential energy function.

Smolyak's interpolating polynomial is constructed by evaluating the Equation 4.1 at each point of the sparse grid. We note that the interpolation domain for the ith dimension must be specified to match the corresponding design variable x_i. For example, $[0°, 360°]$ would be an appropriate domain for a dihedral angle. The nested structure of sparse grids (recall from Chapter 2 that $H(q,d) \subset H(q+1,d)$) means that one can reuse all of the function evaluations from the k sparse grid, which are in our case expensive electronic structure calculations, to obtain an approximation on the $k+1$ sparse grid. The nestedness also allows one to estimate interpolation errors [146]. Explicit error bounds for Smolyak's algorithm have been studied extensively (see Section 2.2.2), but in general depend on the dimension d, the degree of polynomial exactness k, the size of the interpolation domain, and the smoothness of the PESs $E_n(x)$.

To employ Smolyak's algorithm one must perform the electronic structure optimization in Equation 4.1 and any corresponding excited state energy calculations for each point x_i in the sparse grid $H(q,d)$. While the necessary electronic structure calculations are expensive, they are the sole computational burden of this method. With the sets of sparse grid points x_i and energy values $E_n(x_i)$ in hand, one can use Equation 2.25 to interpolate the PESs. We will denote this surrogate for the potential energy function by

$$E_n^s(x) = \mathcal{A}(q,d)[E_n](x) \tag{4.2}$$

for any electronic state $n \geq 0$.

4.1.2 Dynamics

The goal of our simulations is to successfully predict the natural excitation and relaxation of a molecule for a given sequence of m electronic states $\{n_1, \ldots, n_m\}$ and track the entire reaction path. The sequence can specify both multi-photon and single photon excitations, as well as any decay path. For example, a sequence of $\{0, 3, 0\}$ would represent a single photon excitation and relaxation between the ground state and the third excited state. The reaction path of a molecule moves in the direction of the negative gradient on PESs [143], and large steps across the PESs may pass over local minima or be physically unnatural. With this in mind, we track the reaction path and find local minima via continuous steepest descent[135, 49] by integrating the dynamics

$$\dot{x} = -\nabla E_n^s(x) \tag{4.3}$$

until $\|\nabla E_n^s(x)\|$ is smaller than a prescribed tolerance $\tau \approx 0$. These dynamics employ the Born-Oppenheimer approximation (see Section 3.2) and are referred to as *adiabatic* dynamics, or dynamics where interactions are represented on a single PES.

The solution $\mathbf{x}$ to 4.3 is the reaction path and is approximated using MATLAB's `ode15s` variable order solver [140, 187]. We stress that the time step is non-physical and in no way reflects the time scale of the physical reaction; we are only interested in the path the molecule takes. Continuous steepest descent would be unfeasible without the surrogate model. The gradient of the potential energy in Equation 4.1, while available in analytic form, is far too expensive to compute at each time step. On the other hand, the surrogate model $E_n^s(x)$ and its gradient, which can also be computed analytically, are both multidimensional polynomials and are much less expensive to evaluate than $E_n(x)$ and its gradient $\nabla E_n(x)$. For moderately sized molecules, the evaluation time can be reduced from several minutes to a fraction of a second.

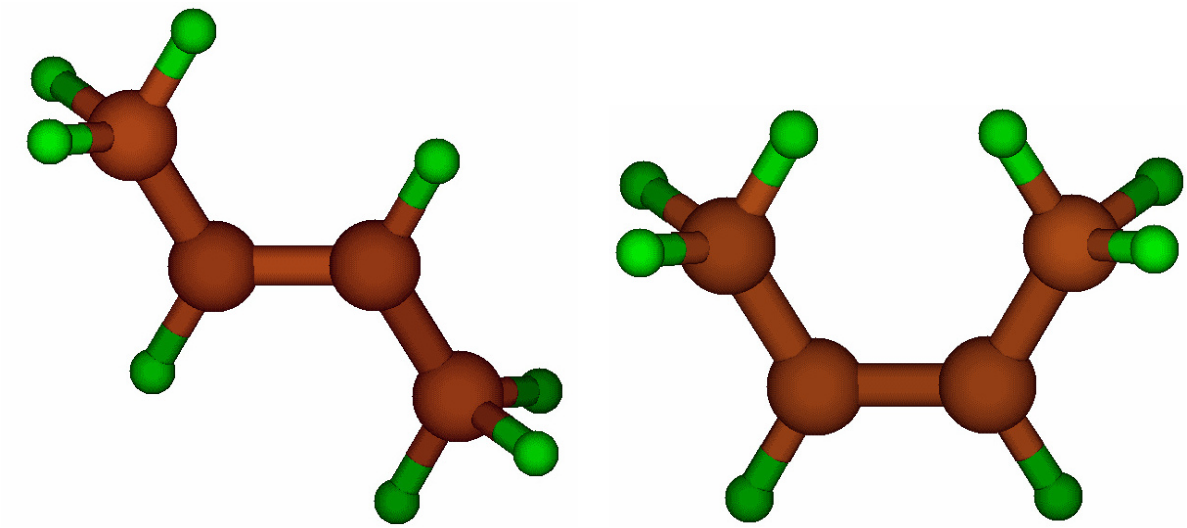
Each reaction path simulation begins with $n_1 = 0$ at an equilibrium geometry on the surrogate ground state PES, from which the molecule is excited to the next specified electronic state n_2. The steepest descent path is followed to find a local minimizer $\hat{x}^{n_2}$ on the PES, and then the molecule is excited to the next state. The simulation continues in this way until a local minimum on the highest specified electronic state, say n_i, has been reached. To simulate the relaxation of a molecule from one state to the next, thermal fluctuations and molecular vibrations are accounted for by reinitializing dynamics on the succeeding state at $\hat{x}^{n_i} + \gamma$ where γ is a d-dimensional vector of random numbers. The domain for γ_j depends on the type of molecular coordinate: $\gamma_j \in [-0.05, 0.05]$ Å if the j^{th} design variable is a bond length, and $\gamma_j \in [-30, 30]^\circ$ if the j^{th} design variable is a bond or dihedral angle. These intervals were chosen arbitrarily to approximate the effects of molecular vibrations and thermal fluctuations.

4.2 Examples

4.2.1 2-Butene

A good test molecule for our simulations is 2-butene because it has a known transition path from *cis*-2-butene to *trans*-2-butene via excitation to the first excited singlet state [19, 158, 226]. Both stable ground state geometries for 2-butene are shown in Figure 4.1 and the isomerization process is depicted in Figure 4.2 where we label each of the four C atoms with superscripts.

The first step of our reaction path method is to choose design coordinates and corresponding interpolation bounds. The main reaction coordinate for the isomerization of 2-butene is the dihedral angle formed by the four C atoms C^1-C^2=C^3-C^4, so we choose this angle to be x_1. The *cis*-2-butene and *trans*-2-butene geometries correspond to $x_1 = 0^\circ$ and $x_1 = 180^\circ$ and the transition state corresponds to $x_1 = 90^\circ$ [19, 158, 226], so we choose bounds of $[-190, 190]^\circ$ for



(A) (B)

Figure 4.1: The two stable ground state geometries of 2-butene: (A) *trans*-2-butene and (B) *cis*-2-butene.

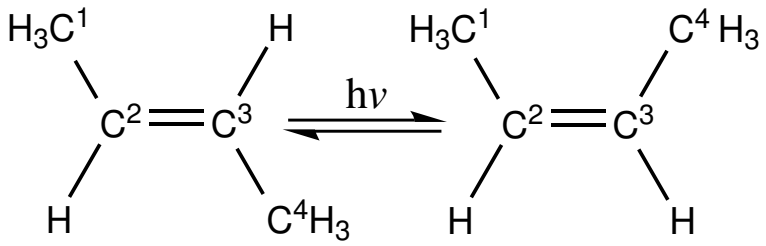


Figure 4.2: Photoisomerization of *trans*-2-butene (left) and *cis*-2-butene (right). Carbon atoms are labeled with superscripts.

x_1. A one dimensional simulation of the reaction path with this degree of freedom without the effects of randomness is shown in Figure 4.3 and reproduces results found in [226] and [146].

The second and third coordinates x_2 and x_3 are rotations of the methyl groups that are governed by the dihedral angles formed by the atoms C^3=C^2-C^1-H_3 and C^2=C^3-C^4-H_3. Only one C=C-C-H dihedral angle is frozen during the optimization, as we expect the methyl group to retain its symmetry. We do not expect these coordinates to change much during the simulation, so we choose bounds of $[-130, 130]$ degrees for both x_2 and x_3. Design coordinates x_1, x_2, and x_3 are shown in 4.4. Finally, we choose the three carbon-carbon bond lengths as additional

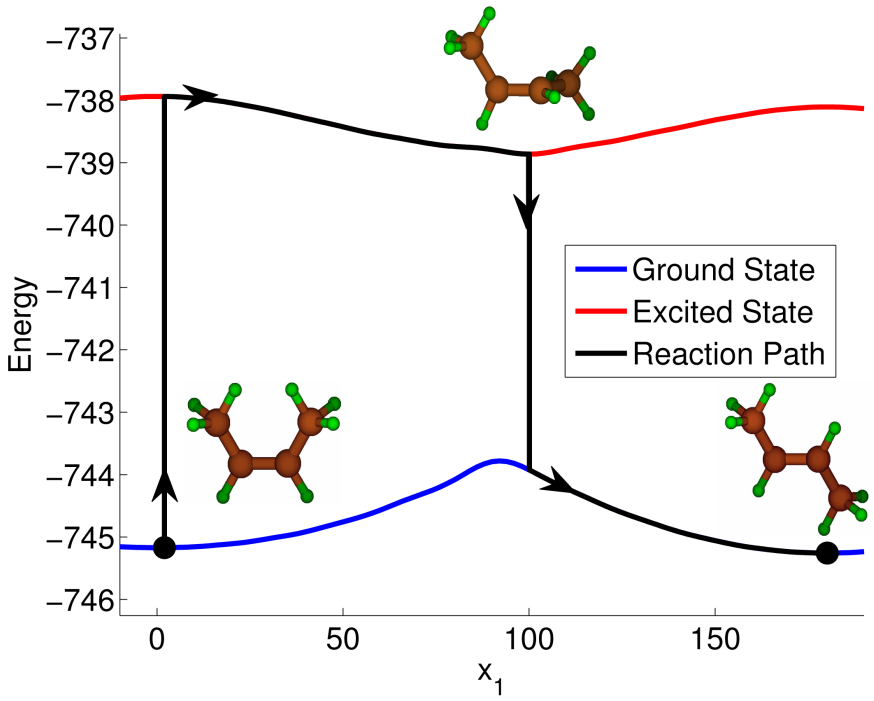


Figure 4.3: Reaction path of 2-butene for the photoisomerization of *cis*-2-butene to *trans*-2-butene. The ground state PES is labeled in blue, the first excited state PES is labeled in red, and the reaction path is labeled in black with arrows pointing in the direction of the path the molecule follows.

degrees of freedom. The C^2=C^3 bond length is x_4, the C^1-C^2 bond length is x_5, and C^3-C^4 bond length is x_6. Since we expect the double bond length (x_4) to change more than the single bond lengths (x_5 and x_6), we choose the domain $[1.3, 1.7]$ Å for x_4 and $[1.4, 1.7]$ Å for x_5 and x_6. In the following, all units for reported energies, angles, and bond lengths are electronvolts (eV), degrees, and angstroms (Å), respectively.

We know that the isomerization takes place via the first excited state [226], so our sequence of electronic states to model this phenomenon is $\{0, 1, 0\}$. While our method can be applied using any electronic structure method and basis set, each geometry optimization is calculated using the B3LYP hybrid functional [200, 119] with the CEP-31G* basis set [201, 202, 50]. All excitation energies are computed using the TD-DFT method [178]. We used the GAUSSIAN 09 software package [72] for all electronic structure calculations.

We present results for three simulations: Simulation 1 is a two-dimensional simulation with design coordinates x_1 and x_2, Simulation 2 is a three-dimensional simulation with design coordi-

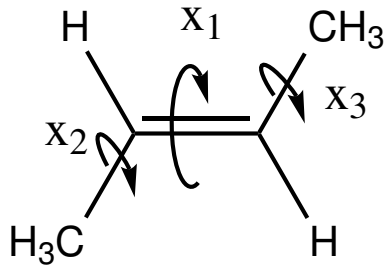


Figure 4.4: Design variables x_1, x_2, and x_3 for 2-butene simulations.

nates x_1, x_2, and x_3, Simulation 3 is a six-dimensional simulation with all six design coordinates. To our knowledge this is the first time 2-butene isomerization has been studied with more than 3 degrees of freedom. All three simulations are initialized at the stable *cis*-2-butene geometry, and the gradient norm tolerance for continuous steepest descent is set to $\tau = 10^{-10}$. We employ our own implementation of Smolyak's algorithm that is specifically designed to construct and follow multiple dynamics paths on PESs [152].

Refinement Study

For each simulation we must choose an appropriate degree of polynomial exactness k for Smolyak's algorithm. To do this, we perform a refinement study in k for each dimension $d = 2, 3,$ and 6 by comparing the local minima of the true PESs with those of the surrogate PESs. Tables 4.1-4.3 show minimizers for the fully optimized molecule and PESs approximated with $k = 1, 2, \ldots, 5$ for $d = 2$, $d = 3$, and $d = 6$. Since the interpolation domain is centered on the *cis* geometry for x_1, x_2, and x_3, the *cis* minimizer for surrogate PES aligns perfectly with the fully optimized structure for these degrees of freedom. Greater discrepancy is found between the *trans* minimizers, where a degree of exactness of $k = 4$ or $k = 5$ aligns with the fully optimized structure to within an acceptable degree of accuracy.

We also use the method of manufactured solutions to observe how the accuracy of our approximation increases with k. Henceforth we assume that the $k = 5$ PES is the "true" solution. This value of k was chosen since the minimizers of its surrogate PES sufficiently approximated the physical *cis* and *trans* minimizers (see Tables 4.1-4.3). We then quantify the error by calculating the largest relative error ϵ_{rel} at points in the sparse grid $\mathcal{H}(d+5, 5)$. If $\hat{E}_k(x)$

Table 4.1: Two-dimensional minimizers corresponding the the *trans*-2-butene geometry for the true PES and Smolyak's surrogate PES for various values of k.

PES	**x** (*cis*)	**x** (*trans*)
True	(0.0, 0.0)	(180.0, 0.0)
$k = 1$	(0.0, 0.0)	(190.0, 0.0)
$k = 2$	(0.0, 0.0)	(190.0, -5.0)
$k = 3$	(0.0, 0.0)	(168.8, 6.8)
$k = 4$	(0.0, 0.0)	(180.2, -6.3)
$k = 5$	(0.0, 0.0)	(178.9, 2.1)

Table 4.2: Three-dimensional minimizers corresponding the the *trans*-2-butene geometry for the true PES and Smolyak's surrogate PES for various values of k.

PES	**x** (*cis*)	**x** (*trans*)
True	(0.0, 0.0, 0.0)	(180.0, 0.0, 0.0)
$k = 1$	(0.0, 0.0, 0.0)	(190.0, 0.0, 0.0)
$k = 2$	(0.0, 0.0, 0.0)	(190.0, -5.1, -5.1)
$k = 3$	(0.0, 0.0, 0.0)	(169.0, 7.0, 7.0)
$k = 4$	(0.0, 0.0, 0.0)	(180.0, -6.4, -6.4)
$k = 5$	(0.0, 0.0, 0.0)	(178.9, 2.2, 2.3)

Table 4.3: Six-dimensional minimizers corresponding the the *trans*-2-butene geometry for the true PES and Smolyak's surrogate PES for various values of k.

PES	**x** (*cis*)	**x** (*trans*)
True	(0.0, 0.0, 0.0, 1.36, 1.52, 1.52)	(180.0, 0.0, 0.0, 1.36, 1.52, 1.52)
$k = 1$	(0.0, 0.0, 0.0, 1.35, 1.54, 1.54)	(190.0, 0.0, 0.0, 1.35, 1.54, 1.54)
$k = 2$	(0.0, 0.0, 0.0, 1.37, 1.52, 1.52)	(190.0, -4.7, -4.7, 1.36, 1.52, 1.43)
$k = 3$	(0.0, 0.0, 0.0, 1.37, 1.52, 1.52)	(170.7, 6.1, 6.1, 1.37, 1.52, 1.52)
$k = 4$	(0.0, 0.0, 0.0, 1.37, 1.52, 1.52)	(179.4, -6.4, -6.3, 1.36, 1.52, 1.52)
$k = 5$	(0.0, 0.0, 0.0, 1.37, 1.52, 1.52)	(179.0, 1.2, 1.2, 1.36, 1.52, 1.52)

is the PES approximated with a degree of polynomial exactness k, then the error indicator is

$$\epsilon_{rel}(k) = \max_{x \in \mathcal{H}(d+5,5)} \frac{\|\hat{E}_k(x) - \hat{E}_5(x)\|}{\|\hat{E}_5(x)\|} \tag{4.4}$$

$$= \max_{x \in \mathcal{H}(d+5,5)} \frac{\|\hat{E}_k(x) - E(x)\|}{\|E(x)\|} \tag{4.5}$$

since $\hat{E}_5(x) = E(x)$ for all $x \in \mathcal{H}(d+5, d)$.

Results of the refinement study for each dimension are shown in Figure 4.5. It is interesting to note that ϵ_{rel} is on the order of 10^{-3} for all degrees of exactness k in each dimension d. Furthermore, while ϵ_{rel} does decrease as we increase k, it does not decrease as much as one might expect.

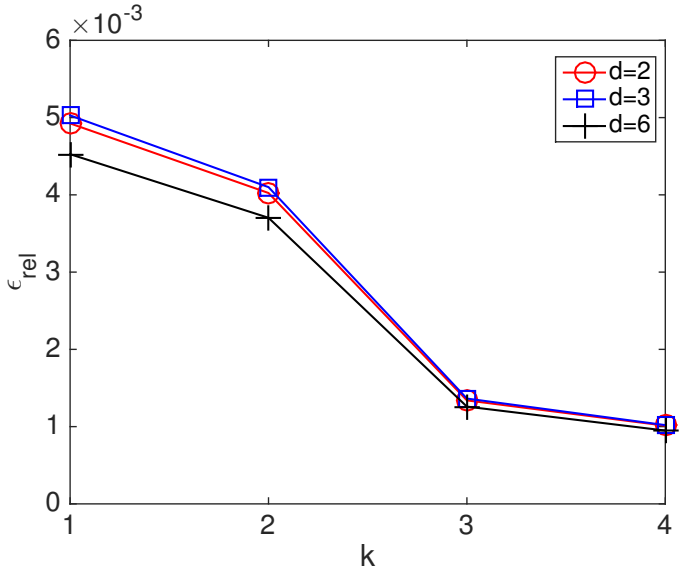


Figure 4.5: Refinement study results for 2-butene: ϵ_{rel} vs. k for $d = 2$, $d = 3$, and $d = 6$.

Simulation 1: $d = 2$

The interpolated PESs for both the ground and first excited state are shown in Figure 4.6 with each sparse grid point. One can visually observe that the global minimum of the first excited state lies directly above the global maximum of the ground state, so it is clear that randomness caused by molecular vibrations and thermal fluctuations could play an important role in the reaction. Also, as we expect, the potential energy does not vary much in the x_2 direction.

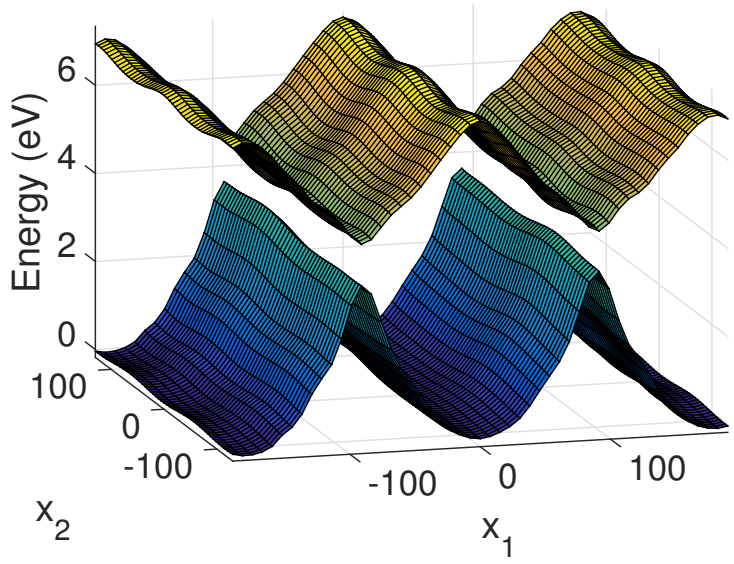


Figure 4.6: Ground and first excited state PESs approximated with $k = 5$.

Fifty different reaction paths are shown on the PESs in Figure 4.7. The reaction path is initialized on the ground state in the *cis* geometry at $x = (0.0, 0.0)^\circ$, and continuous steepest descent converges to a minimizer of $\hat{x}^1 = (-89.9, 49.5)^\circ$ on the first excited state PES. The approximated effects of thermal fluctuations and molecular vibrations upon relaxation do indeed either send the molecule back to its original geometry or towards its *trans*-2-butene counterpart. Note that that x_2 also converges to the two geometrically identical values of either 0° or -120°.

Simulation 2: $d = 3$

For this simulation we include our third degree of freedom x_3. The reaction path is initialized on the ground state at $x = (0.0, 0.0, 0.0)^\circ$, corresponding to the *cis*-2-butene geometry of our

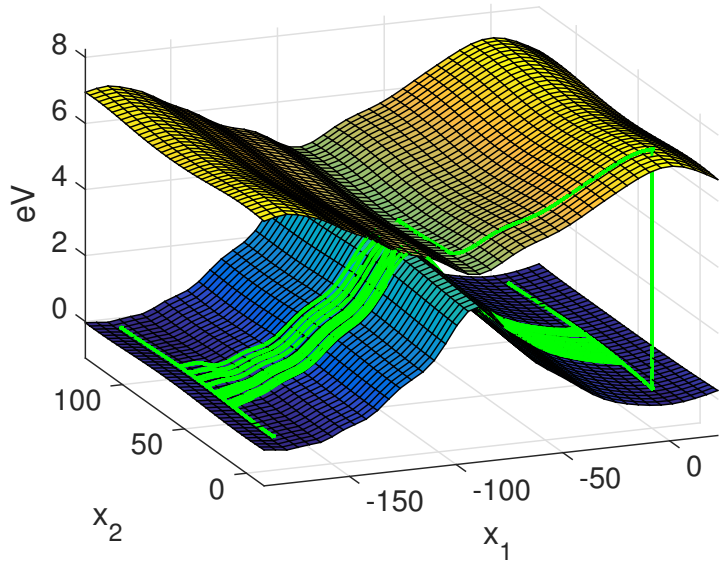


Figure 4.7: 50 simulation reaction paths on PESs approximated with $k = 5$. Here we show only the relevant portion of the larger PES shown in Figure 4.6.

surrogate PES. After excitation to the first excited singlet state, continuous steepest descent converges to a minimizer of $\hat{x}^1 = (-89.7, 43.6, 43.7)^\circ$. Upon relaxation, the effects of thermal fluctuations and molecular vibrations again either send the molecule back to a *cis* geometry or towards *trans*. Fifty different reaction paths are shown in Figure 4.8.

The simulation finds several different ground state local minima, but each has values sufficiently close to 0° or 180° for x_1. Dihedral angles x_2 and x_3 both change from 0° to approximately 44° while the molecule is in its first excited electronic state, and the simulated randomness upon relaxation sends trajectories to either 0° or 120° (note that these two values are geometrically equivalent for the rotation of the methyl groups). The energies of each reaction converge to two slightly different values, reflecting the fact that the *trans* geometry is a lower energy configuration than its *cis* counterpart.

Simulation 3: $d = 6$

Finally, we include the three carbon-carbon bond lengths for a total of six degrees of freedom. Simulations begin on the ground state at the local minimum $x = (0.0, 0.0, 0.0, 1.37, 1.52, 1.52)$ corresponding to the *cis* geometry and are excited to the first electronic state. Continuous steepest descent converges to a minimum of $x^1 = (-89.6, 49.7, 49.6, 1.42, 1.52, 1.52)$ on the first

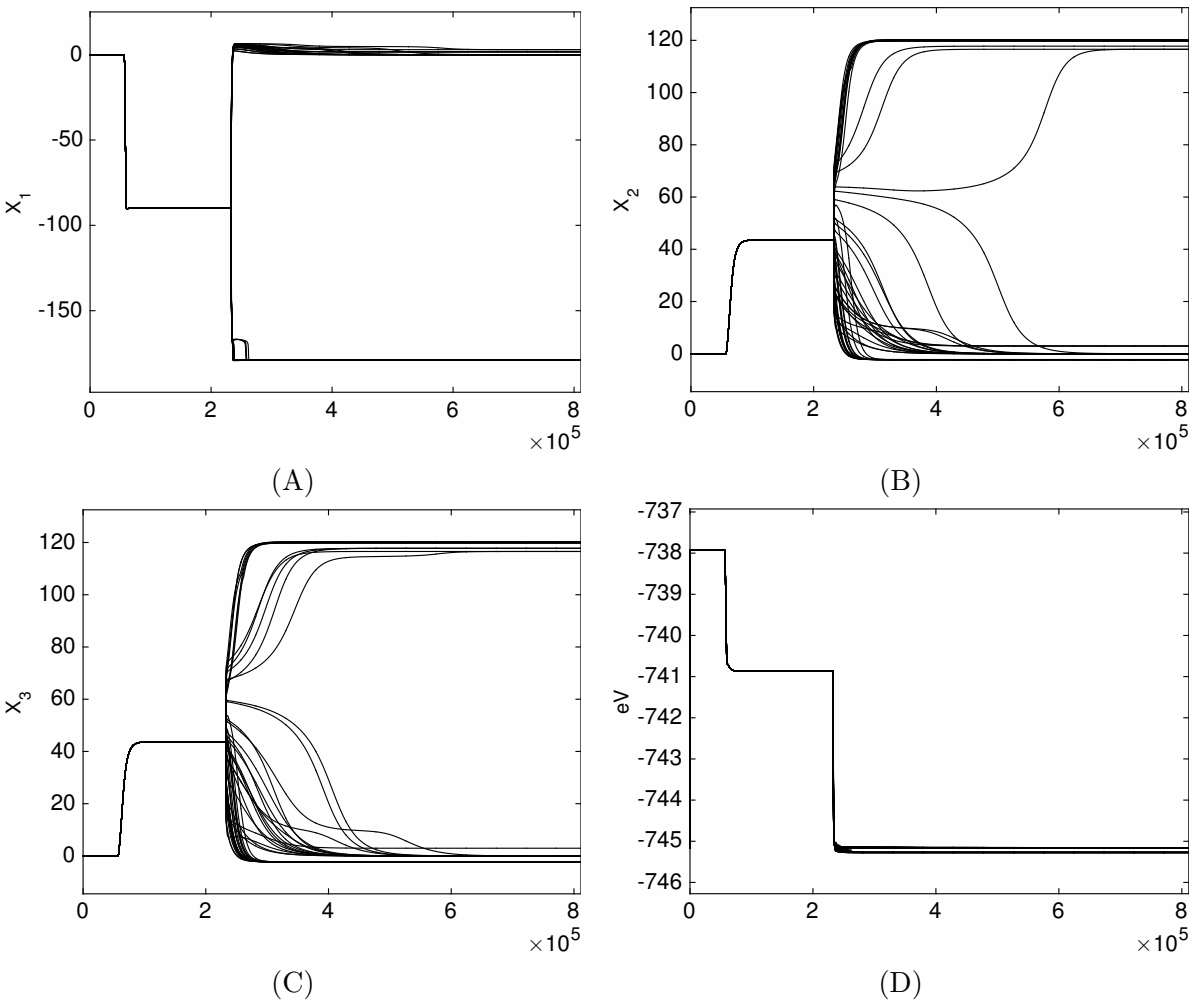


Figure 4.8: Simulation 3 results for 2-butene: (A) reaction paths for design variable x_1, (B) reaction paths for design variable x_2, (C) reaction paths for design variable x_3, (D) energies for reaction paths. The x-axis in each plot is our non-physical time variable.

excited state PES and 50 reaction paths are reinitialized on the ground state. Reaction paths for all 6 degrees of freedom are shown in Figure 4.9.

The dihedral angle x_1 behaves as expected, with trajectories converging sufficiently close to 0° or 180°. Similar to Simulation 2, dihedral angles x_2 and x_3 both change from 0 to 49.7° while the molecule is in its first excited electronic state, and the simulated randomness upon relaxation sends trajectories to either 0 or 120° (again, note that these two values are geometrically equivalent for the rotation of the methyl groups). The reaction paths for x_4 show that the C^2=C^3 bond length changes from its initial value of 1.37 Å to 1.59 Å in a comparatively small

number of time steps after excitation, but eventually decreases to its stable minimizer of 1.42 Å there after. After relaxation back to the ground state, x_4 converges to two slightly different bond lengths corresponding to *cis* and *trans* geometries. The single carbon bond lengths x_5 and x_6 vary as well, but by no more than 0.04 Å . Similar to x_4, the trajectories for x_5 and x_6 converge to two different bond lengths. Energy paths resemble those shown in Figure 4.8(D) for Simulation 2 and are not shown for this simulation.

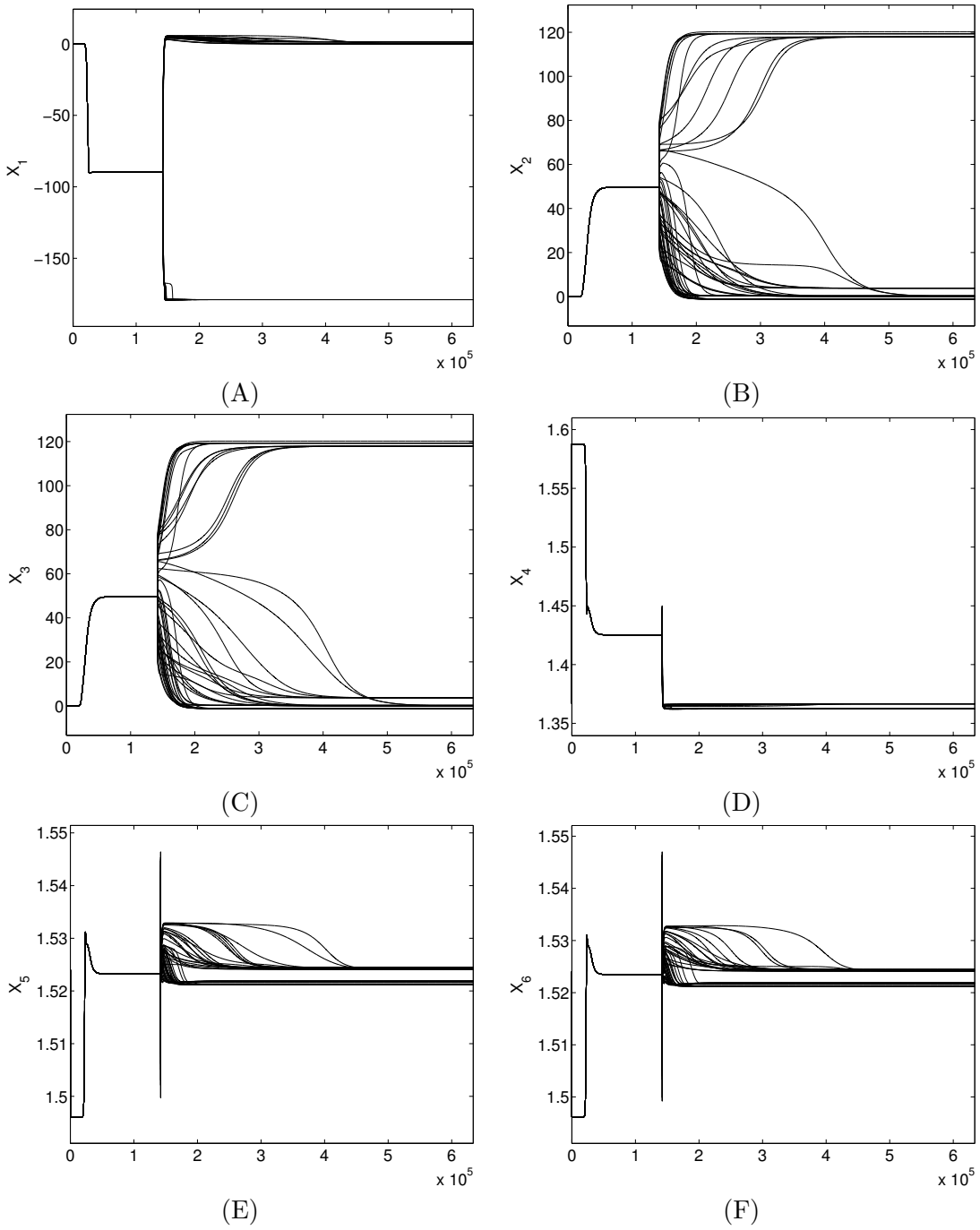
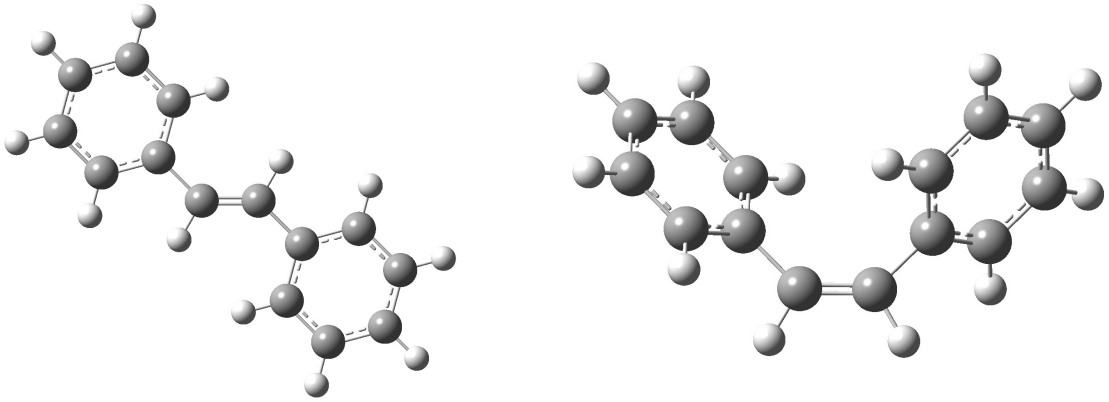


Figure 4.9: Simulation 4 results for 2-butene: (A) reaction paths for design variable x_1, (B) reaction paths for design variable x_2, (C) reaction paths for design variable x_3, (D) reaction paths for design variable x_4, (E) reaction paths for design variable x_5, (F) reaction paths for design variable x_6. The x-axis in each plot is our non-physical time variable.

4.2.2 Stilbene

Stilbene is considered a model for trans-cis isomerization studies, as they are ideal for molecular electronics due to their properties of organic conductors, photoswitches, organic displays, or biosensors [176]. Like 2-butene, stilbene isomerization has been studied extensively and a transition paths between *cis* and *trans* are known [193, 21, 216, 186, 219].

Both *cis* and *trans* geometries are shown in Figure 4.10.



(a): *trans*-stilbene (b): *cis*-stilbene

Figure 4.10: Stilbene molecule, $C_{14}H_{12}$.

3 Degrees of Freedom

To corroborate our results, we will focus on a recent study by Jiang et al. [114] where they model laser-induced *trans* to *cis* isomerization with a method called semiclassical electron-radiation-ion dynamics (see A.3 for details on this method).

For our simulations we chose three degrees of freedom: x_1 is a rotation about the central C=C bond, and x_2 and x_3 correspond to the two vinyl-phenyl dihedral angles. From [114], the *cis* geometry corresponds to $\bar{x} = (0, 50, 50)$ (this value differs from what is reported in [114] because of how they define the dihedral angle controlling this rotation - we have modified it appropriately) and the *trans* geometry corresponds to $\bar{x} = (180, 0, 0)$. The interpolation domain for each degree of freedom is: [-10, 190]°, [-80, 80]°, and [-80, 80]°, respectively. Figure 4.11 shows these degrees of freedom with the molecule at its *cis* stable geometry. We construct PESs with a degree of exactness of $k = 3$ and perform each geometry optimization at the B3LYP/6-31G* level of theory. The stilbene z-matrix used for the electronic structure calculations can be found

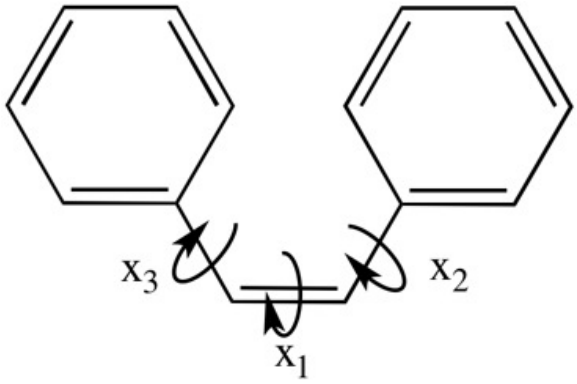


Figure 4.11: Degrees of freedom for stilbene simulations.

in Appendix A.6.

We initialize simulations in the *cis* geometry and excite the molecule to its first excited singlet electronic state. 50 different reaction paths are shown in Figure 4.12, where the artificial vibrations and fluctuations successfully simulate the *cis-trans* photoisomerization process.

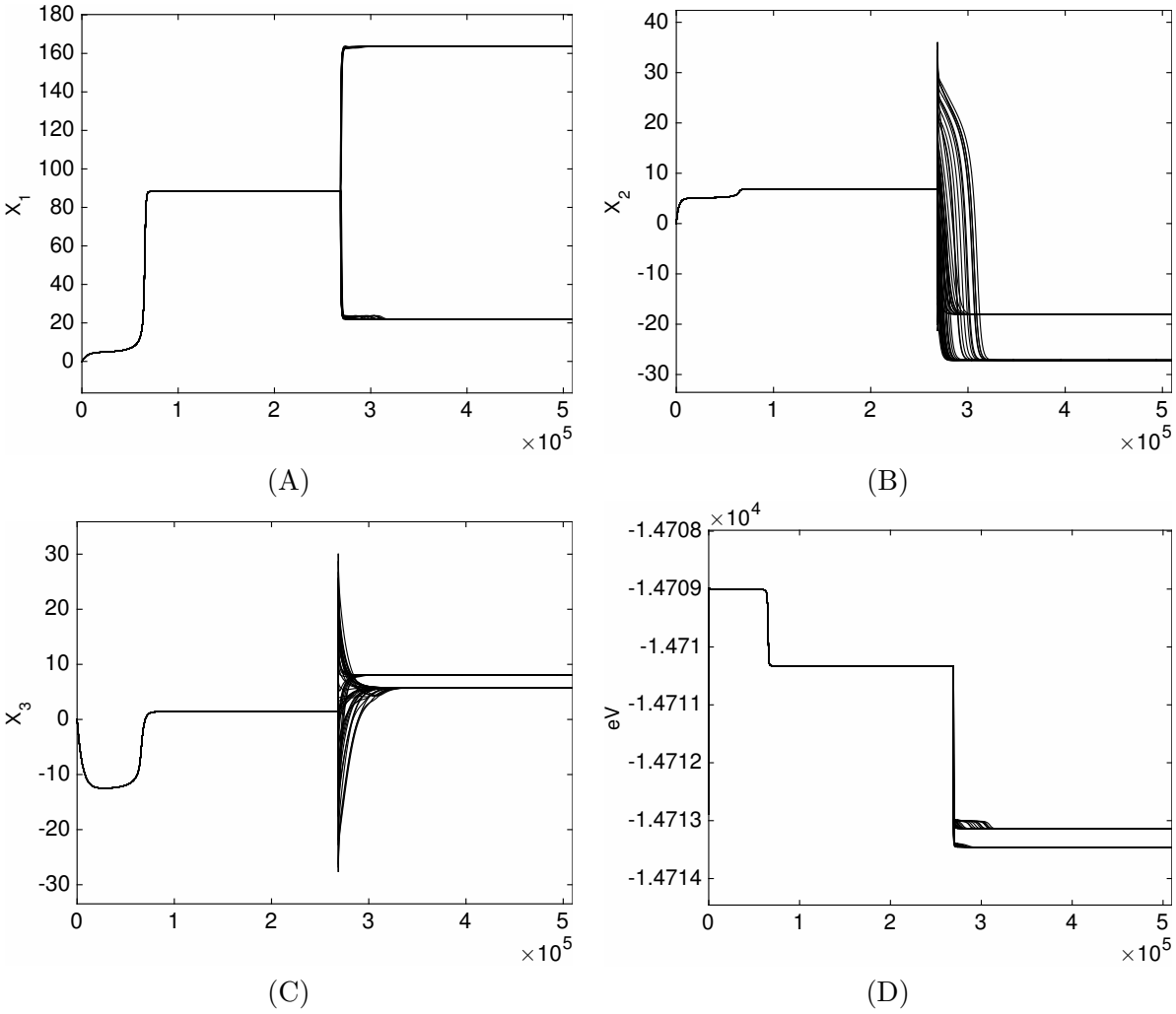


Figure 4.12: Simulation results for stilbene: (A) reaction paths for design variable x_1, (B) reaction paths for design variable x_2, (C) reaction paths for design variable x_3, (D) energies for reaction paths. In each plot the x-axis is the reaction path time step.

4.2.3 2-Pentene

Pentene (C_5H_{10}) is another molecule that can transition between *cis* and *trans* isomers (see Figure 4.13). The main reaction coordinate, which we set as x_1, is the dihedral angle governing the twisting of the carbon-carbon double bond. The second degree of freedom x_2 is a twisting of the single carbon-carbon bond in the middle of the molecule. Both degrees of freedom are labeled in Figure 4.14. The interpolation domain for both of these degrees of freedom is $[-190, 190]^\circ$. The energies are calculated at the B3LYP/6-31G* level of theory and the PESs are built with a degree of exactness $k = 5$.

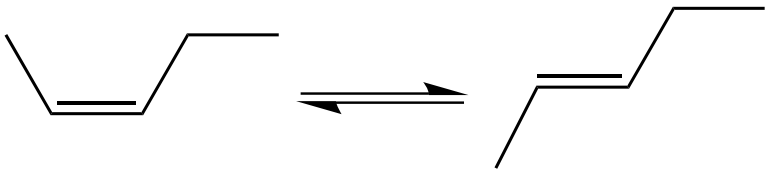


Figure 4.13: Isomerization between *cis*-2-pentene (left) and *trans*-2-pentene (right).

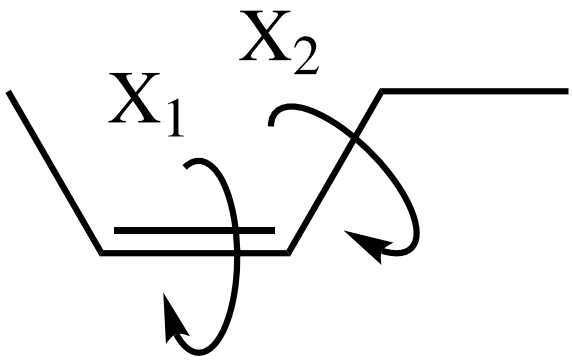
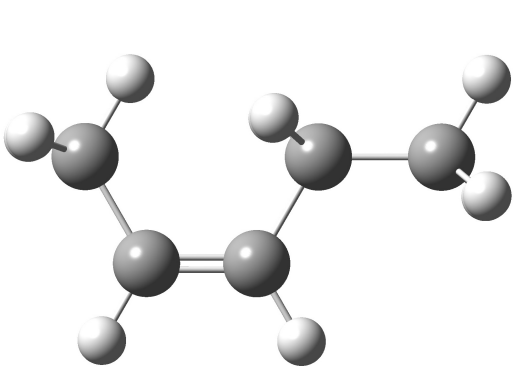


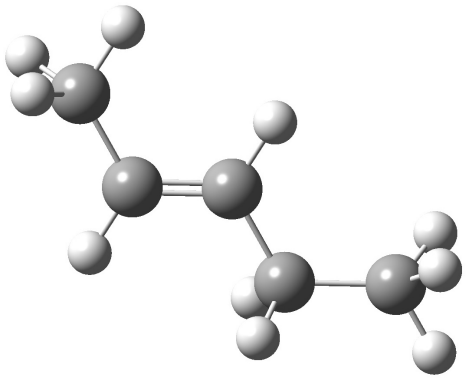
Figure 4.14: Degrees of freedom for pentene reaction path dynamics.

We initialize the simulations in the *cis* conformation with $x = (0.0, 0.0)$ and excite the molecule to the first excited singlet state. The reaction path is followed to the minimizer of the first excited singlet state, $x = (-89.9, -58.3)$, and the simulated randomness causes reaction

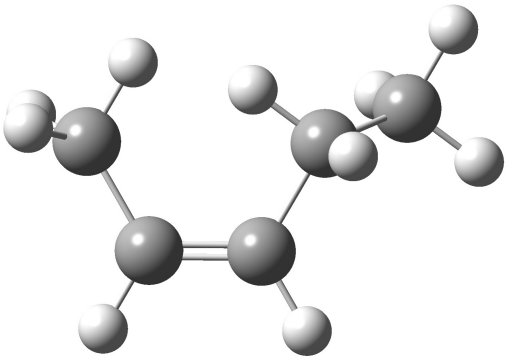
paths to split among the four stable ground state geometries listed in Table 4.4. The reaction paths for 50 different simulations are shown in Figure 4.16. The two isomers *cis-cis* and *trans-cis* (Figures 4.4a and 4.4b) are known, but our reaction path simulations also converge to two other stable geometries which we have labeled *cis-trans* and *trans-trans* (Figures 4.4c and 4.4d).



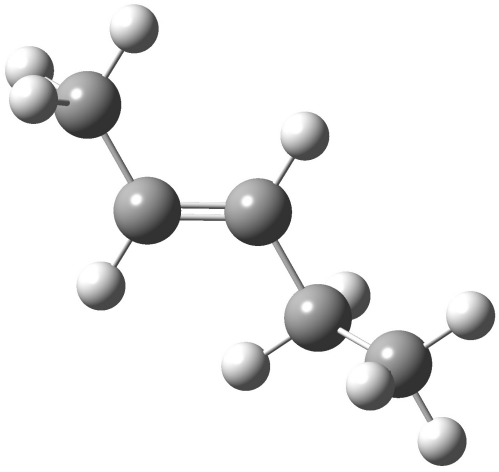
(a): Cis-cis-2-pentene



(b): Trans-cis-2-pentene



(c): Cis-trans-2-pentene



(d): Trans-trans-2-pentene

Figure 4.15: 2-Pentene molecule, C_5H_{10}.

Table 4.4: Two-dimensional minimizers of the approximated PES for 2-pentene.

Stable Geometry	Minimizer (x_1, x_2)
cis-cis	(0.0, 0.0)
cis-trans	(2.4, -119.7)
trans-cis	(-178.9, 8.1)
trans-trans	(-178.8, -119.7)

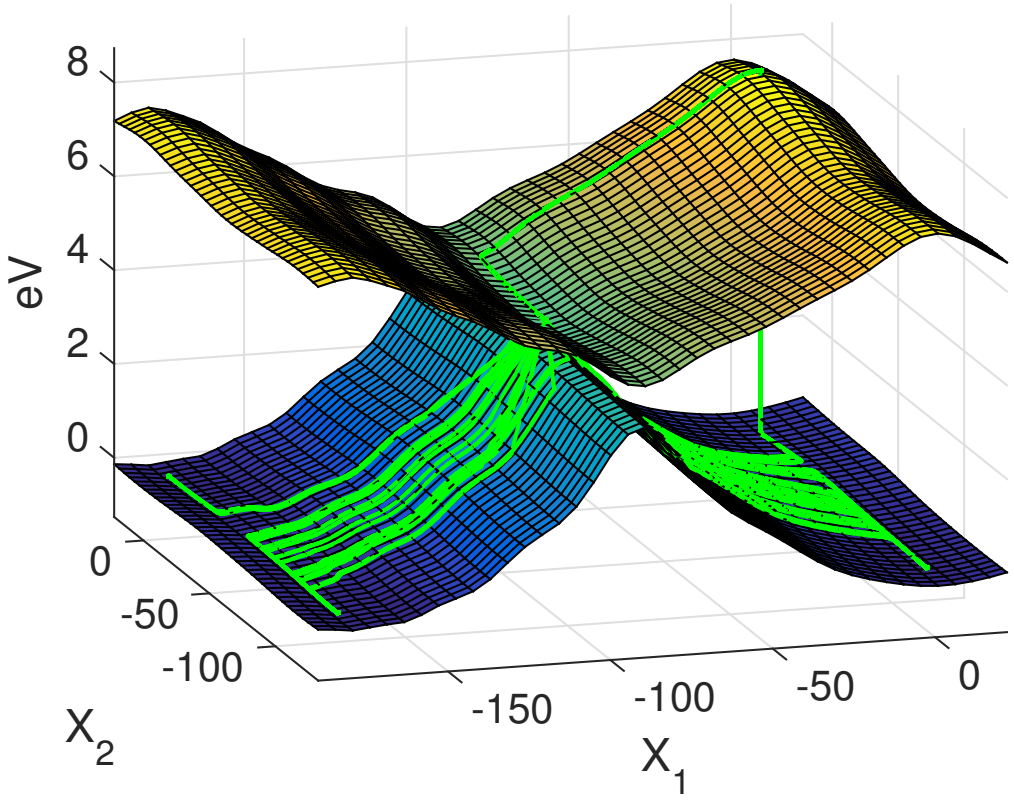


Figure 4.16: Simulations run on the ground and first excited singlet state of 2-pentene. The PESs were generated with $k = 5$.

Chapter 5

Potential Energy Surfaces and Nonadiabatic Dynamics of Fe-based Molecular Complexes

5.1 Introduction

Sunlight offers abundant amounts of energy that could be converted into electricity or other sources of fuel. In 2002, the total solar energy absorbed in one hour by Earth's atmosphere, oceans, and land masses was more energy than the world used in the entire year [148]. While advances in the field have been made, the efficient capture, storage, and transport of energy from sunlight are still significant challenges to properly harvesting this energy. Sunlight can be converted to electricity via dye-sensitized solar cells (DSSCs) [93, 85] or to chemical fuels via photocatalytic synthetic cells [209]. These cells contain either a single photoactive molecule or molecular array anchored to a semiconductor. This work focuses on the latter, where conversion of sunlight to electricity occurs takes place by the interfacial electron transfer (IET) between the molecule and the semiconductor after the molecule absorbs light [93].

The most successful class of molecules used in these cells is based on Ru(II)-polypyridine compounds. While efficient, ruthenium (Ru) is a relatively rare and expensive metal. More common metals such as iron [67, 68] and copper [48] have been investigated for use in these cells, and are ideal for the development of economical and sustainable solar cells or artificial photosynthetic systems because of their low cost and high abundance [60, 147, 134]. Unfortunately, compounds based on these metals are not as efficient as their ruthenium counterpart. The presence of a number of low-lying metal-centered excited states of various spin multiplicities inhibits the control the photochemical activity of compounds based on first row transition metals, i.e. iron [117]. Iron-based complexes have a weaker ligand-field compared to ruthenium-

based complexes, and therefore the metal-centered ligand-field (MC) states lie lower in energy than the lowest energy metal-to-ligand charge transfer (MLCT) state. As a result, successful interfacial electron transfer is impeded by a very fast intersystem crossing (ISC) into these MC states [117, 147, 108, 197]. An ISC is a radiationless transfer between two electronic states with different spin multiplicities. For example, an ISC occurs when an excited singlet state, in which all electron spins are paired (different spin), transitions to an excited triplet state, in which the excited electron is no longer paired with the ground state electron (same spin). Several models of intersystem crossing have been proposed [108, 197, 52, 217, 43, 239, 51], but the phenomenon is still not completely understood. Figure 5.1 shows an energy level diagram comparing the electronic states of an Fe(II)-polypyridine and a Ru(II)-polypyridine. In particular, observe that the ^{5}MC state lies lower in energy for the Fe(II)-polypyridine than it does for the Ru(II)-polypyridine.

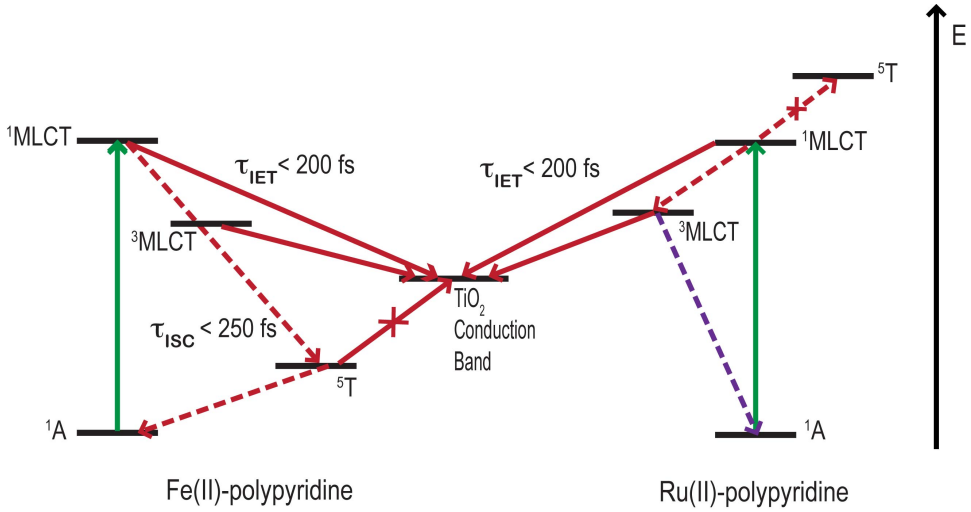


Figure 5.1: Energy level diagram comparing excited state dynamics of Fe(II)-polypyridine and Ru(II)-polypyridine compounds. Green lines represent photo excitation from the singlet ground state (^{1}A) to a singlet MLCT state (^{1}MLCT). Dashed red lines represent the ISC into the MC state (^{5}MC), and solid red lines represent IET into the TiO_2 conduction band. The dashed blue line represents radiative decay back into the ground state. *Image produced by Elena Jakubikova.*

Several factors make this an extremely challenging system to study: first, the complexes in question have upwards of 50 atoms and require a large amount of computational resources for electronic structure calculations. Second, the change in spin multiplicities during ISC cannot be described by adiabatic dynamics methods, such as the one presented in Chapter 4. Third,

the distortions in molecular geometry and the large number of electronic states involved in ISC require more computational overhead than the simple two-state reactions studied in Chapter 4.

One goal of this work is to model ISC events in these Fe-based systems. First, we must obtain a better understanding of the potential energy surfaces (PESs) of Fe(II)-polypyridines. Specifically, we will construct adiabatic PESs for the the ^{1}A, ^{1,3}MLCT, ^{3}MC, and ^{5}MC electronic states for the $[Fe(tpy)_2]^{2+}$ complex. One or two dimensional PES's for FE(II)-polypyridines have been studied in the past [52, 51, 157, 71, 40, 240], and $[Fe(tpy)_2]^{2+}$ has been studied experimentally in [44] and [112].

Second, we will locate minimum energy crossing points (MECPs) between pairs of PES's. A minimum energy crossing point is a point where two energy surfaces corresponding to different spin states intersect with minimum energy. Transitions between the two PESs of different spin are most likely to occur at these points [229, 230, 15]. We will also compute the entire intersection seam for the pairs of PESs that intersect. Because we utilize a sparse grid approximation for the PESs, visualization of the seams is computationally inexpensive.

Finally, we will introduce both nonadiabatic transition state theory [96, 97] and Tully's fewest-switches surface hopping nonadiabatic dynamics [210], two methods by which to simulate ISC and IET. The reaction path following method presented in Chapter 4 can only be applied to adiabatic dynamics - dynamics based on the Born-Oppenheimer approximation that can be represented on a single PES. The complicated changes in spin-states of Fe(II)-polypyridines, however, necessitates the application of nonadiabatic dynamics - dynamics where the Born-Oppenheimer approximation breaks down because of the speed of the reaction or changes in spin-state. The breakdown of the the Born-Oppenheimer approximation necessitates special methods to account for the splitting of the population among several electronic states [12].

In the subsequent sections we will present PES results for the $[Fe(tpy)_2]^{2+}$ complex, introduce nonadiabatic transition state theory and Tully's fewest-switches surface hopping nonadiabatic dynamics, and show how we could extend our reaction path following method from Chapter 4 to account for nonadiabatic dynamics.

5.2 Potential Energy Surfaces of $[Fe(tpy)_2]^{2+}$

$[Fe(tpy)_2]^{2+}$ is an iron complex that has two terpyridine ligands, and is of particular interest to the chemical community due to its unique photophysical properties [100, 175]. The lowest energy excited state for $[Fe(tpy)_2]^{2+}$ is a high-spin quintet (^{5}MC) state, which is populated via a very fast intersystem tossing cascade after excitation from the singlet ground state (^{1}A) [100, 159].

The observed lifetime of the ^{5}MC state at low temperatures increases with decreasing energy difference between the ^{1}A and ^{5}MC states, primarily due to the fact that the non-radiative decay

pathway ^{5}MC$\rightarrow^1$A can only occur via a tunneling mechanism without external perturbations [99]. At low temperatures, this behavior allows "light-induced excited state spin trapping" (LIESST), a process that can potentially be employed for molecular switches such as memory [99]. $[Fe(tpy)_2]^{2+}$ is unique because when it is embedded in a loose solid matrix, the lifetime of the ^{5}MC state drastically increases relative to similar complexes. This phenomenon is called "strong-field LIESST" (SF-LIESST), and is not fully understood. Elucidating the properties of $[Fe(tpy)_2]^{2+}$ that cause this phenomenon could establish new design principles for molecular switches and sensitizers.

Studying the PESs of $[Fe(tpy)_2]^{2+}$ could yield insights into the mechanism of SF-LIESST. As such, these surfaces of are particular interest to the chemical community and are the subjects of active research. It has been suggested that a single reaction coordinate model is insufficient to describe the spin-state transitions of $[Fe(tpy)_2]^{2+}$ [100]. In [159], for example, the authors study the phenomenon with respect to two degrees of freedom: the axial Fe-N bond lengths and the "biting" angle created by the NNN atoms of each ligand. The elongation of the Fe-N bond lengths occurs because two electrons are transferred from the nonbonding t_{2g} orbitals to the e_g^* type anti bonding orbitals, leading to the expansion of the system [159].

After inspecting fully optimized structures for the ^{1}A, ^{3}MC, and ^{5}MC states, we identified three degrees of freedom for our surrogate PESs: axial N-Fe bond lengths, equatorial N-Fe bond lengths, and a "rocking" angle. This work marks the first time that such an angle has been considered in analyzing the photochemical processes of Fe(II)-polypyridines. The axial bond lengths x_1 are the two bonds connecting the Fe atom to the N atom of each ligand's middle pyridine. The equatorial bond lengths x_2 are the four bonds connecting the Fe atom to the N atoms of each ligand's outer pyridines. The rocking angle x_3 is defined as the axial N-Fe-N bond angle's deviation from 180°. The interpolation domain for each of these degrees of freedom is [1.76, 2.37] Å. The rocking angle is defined as the axial N-Fe-N bond angle's deviation from 180°. The interpolation domain for x_3 is [-20, 20] degrees (°). The three degrees of freedom are shown in Figure 5.2. Figure 5.3 shows the molecule with a rocking angle of 0° and 20°. To simplify notation and clarify discussion, we relabel our coordinates $(R_{ax}, R_{eq}, \Theta) := (x_1, x_2, x_3)$ in the following.

We will construct diabatic PESs for the the ground singlet (^{1}A), lowest-lying metal-to-ligand transfer (^{1}MLCT), lowest-lying triplet (^{3}MC), and lowest lying (^{5}MC) electronic states for the $[Fe(tpy)_2]^{2+}$ complex. For our initial studies, we employed Smolyaks algorithm with a degree of exactness of k=1, but found that a degree of exactness of k=3 was necessary to achieve a satisfactory approximation. As previously mentioned, the nested structure of sparse grids allowed us to reuse calculations from the k=1and k=2 grids when constructing the k=2 and k=3 grids, respectively. For example, the k=1 sparse grid contains 7 grid points and the k=2 sparse grid contains 25 grid points. If we already have the energy values at the 7 grid points

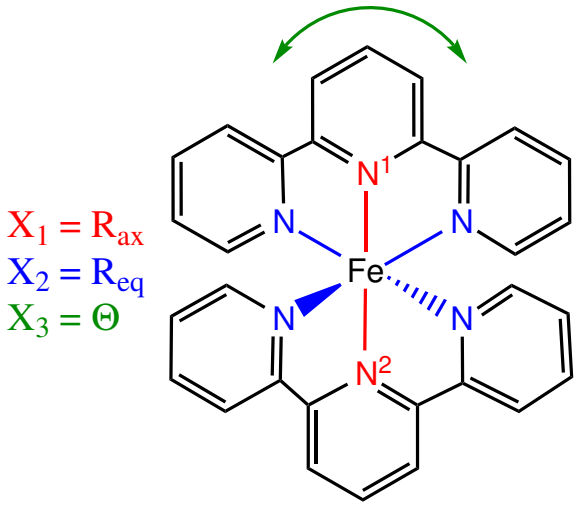


Figure 5.2: The 3 degrees of freedom for $[Fe(tpy)_2]^{2+}$.

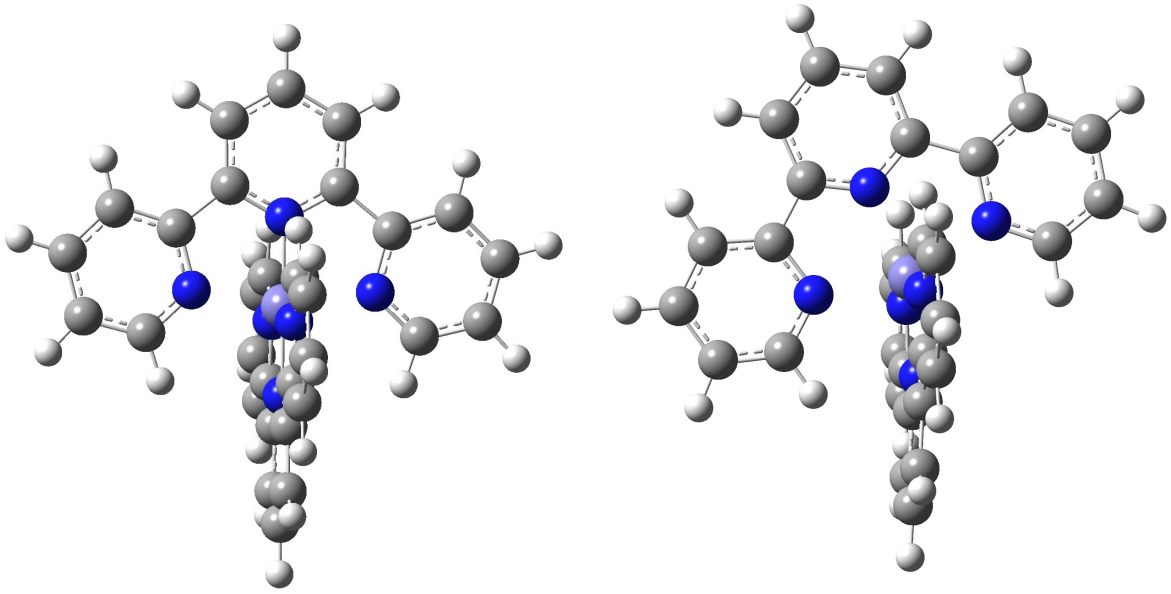


Figure 5.3: The iron-based complex $[Fe(tpy)_2]^{2+}$ with a rocking angle of 0° (left) and a rocking angle of 20° (right).

from the k=1 sparse grid, we only need to perform 18 new electronic structure calculations to construct the k=2 approximation.

We use the B3LYP functional [200, 119] with Grimme's D2 dispersion correction [91], which has been shown to be applicable to Fe(II)-polypyridine based complexes [28]. We utilize the SDD basis set for Fe and the 6-311G* basis for all other atoms for the ^{1}A, ^{3}MC, and ^{5}MC surfaces. The polarizable continuum model will be used to account for solvent effects (water). TD-DFT [178, 204, 41, 14] with the same functional and basis set are used to obtain the energies of ^{1}MLCT and ^{3}MLCT states, using the ^{1}A state as a reference.

For some sparse grid points the triplet and quintet electronic structure calculations did not converge to the desired electronic state. For example, several calculations for the ^{3}MC surface converged to a MLCT state rather than the correct metal-centered state. This is not entirely surprising as the ^{3}MC and ^{5}MC states are highly destabilized at reduced metal-ligand bond lengths. We developed a step-by-step process to attempt to converge to the correct state for these cases. The first step is to find "nearby" sparse grid point that converged to the correct state. Since it is known that the presence of a solvent stabilizes MLCT states, we then perform a single point calculation in vacuum using the wavefunction from the "nearby" sparse grid point as an initial guess. If the single point calculation successfully converges to the correct state, we then proceed with a geometry optimization in vacuum using the single point wavefunction as an initial guess. If the resulting electronic state is still MC, we repeat the single point/optimization steps in solvent using the wavefunction from the previous steps as an initial guess. To summarize, the process is:

1. Find a "nearby" sparse grid point that converged to the correct state.
2. Perform a single point calculation in vacuum using the "nearby" wavefunction as an initial guess.
3. If MC state, perform an optimization in vacuum using the wavefunction from step 2 as a guess.
4. If MC state, perform a single point calculation in solvent using the wavefunction from step 3 as an initial guess.
5. If MC state, perform an optimization in solvent using the wavefunction from step 4 as a guess.

If at any point these calculations do not converge to the correct state, one could either try using an initial guess from a different "nearby" sparse grid point, or simply keep the results from the lowest energy state. In choosing the latter, we simply refer to the PES as adiabatic instead of diabatic. We were unable to compute the MC state for 4 triplet sparse grid points and for 6 quintet sparse grid points, and these points are shown on the 3D sparse grid in Figures 5.4 and 5.5, respectively. These points all correspond to geometries with the shortest axial and

equatorial bond lengths, so we believe the MLCT state is lower in energy than the MC state for these geometries.

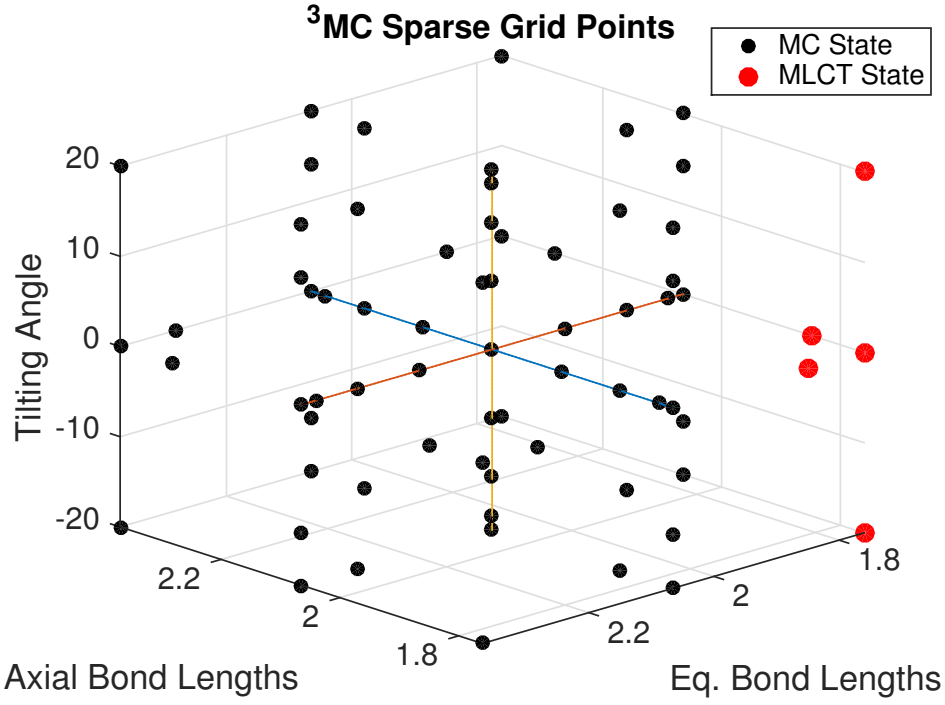


Figure 5.4: Electronic states for each sparse grid point for the ^{3}MC adiabatic PES. Metal-centered (MC) states are labeled in black, and metal-to-ligand charge transfer (MLCT) states are labeled in red.

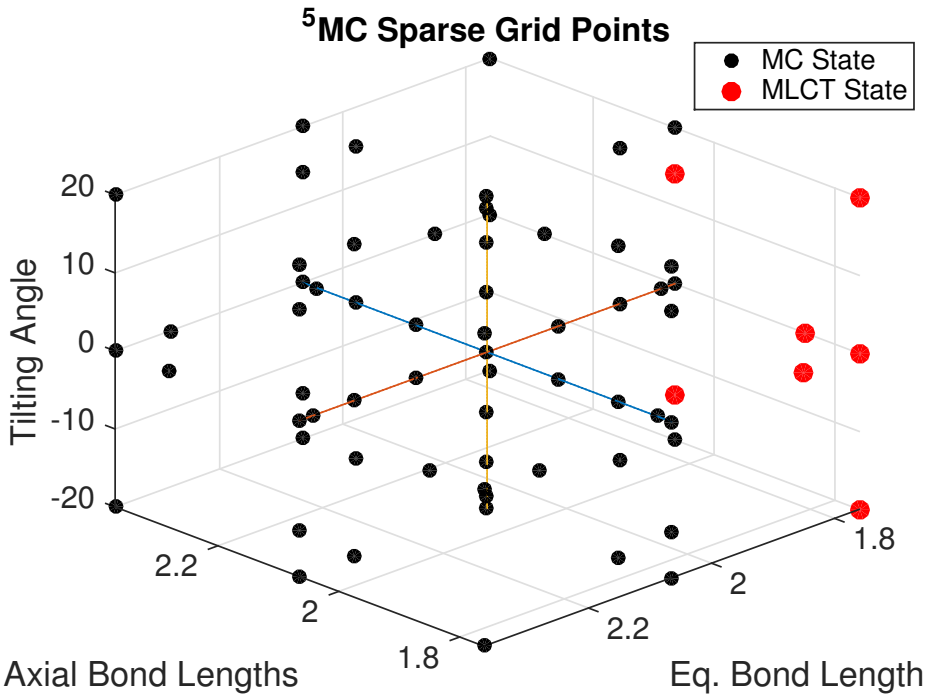


Figure 5.5: Electronic states for each sparse grid point for the ^{5}MC adiabatic PES. Metal-centered (MC) states are labeled in black, and metal-to-ligand charge transfer (MLCT) states are labeled in red.

Finally, a stability analysis was also performed on all singlet, triplet, and quintet electronic calculations. If the wavefunction was found to be unstable, we first stabilized the wavefunction (via the keyword and option Stable=Opt in Gaussian 09), and then performed a geometry optimization using the stable wavefunction as an initial guess.

5.2.1 Results

Table 5.1 shows the minimizers of each PES as well as their optimized values computed from a full optimization in Cartesian coordinates. Surfaces with the degree of polynomial exactness $k = 1$ and $k = 2$ were determined to have insufficient accuracy based on the reproduction of the energy minima of the metal-centered (MC) states. Interpolated minima for $k = 3$ surface are overall in a good agreement with the minima of the fully relaxed geometries. The errors in all metal-ligand bond lengths range from 0.00-0.02 Å, while the errors in Θ range from 0.0-1.0°.

Table 5.1: Optimized values for each degree of freedom for each state. ^{1}A is the singlet ground state, ^{3}MC is the lowest-lying triplet excited state, and ^{5}MC is the lowest-lying quintet excited state.

^{1}A				
Deegree of Freedom	Optimized Value	$k = 1$	$k = 2$	$k = 3$
R_{ax} (Å)	1.91	1.95	1.91	1.91
R_{eq} (Å)	2.00	2.09	2.01	2.01
Θ (°)	0.0	0.0	0.0	0.0
^{1}MLCT				
Deegree of Freedom	Optimized Value	$k = 1$	$k = 2$	$k = 3$
R_{ax} (Å)	-	1.94	1.93	1.90
R_{eq} (Å)	-	2.06	1.99	1.99
Θ (°)	-	0.0	0.0	0.0
^{3}MLCT				
Deegree of Freedom	Optimized Value	$k = 1$	$k = 2$	$k = 3$
R_{ax} (Å)	-	1.87	1.88	1.89
R_{eq} (Å)	-	2.04	1.98	1.94
Θ(°)	-	0.0	0.0	0.0
^{3}MC				
Deegree of Freedom	Optimized Value	$k = 1$	$k = 2$	$k = 3$
R_{ax} (Å)	1.94	2.01	1.92	1.93
R_{eq} (Å)	2.14	2.18	2.14	2.16
Θ (°)	±0.2	0.0	0.0	0.0
^{5}MC				
Deegree of Freedom	Optimized Value	$k = 1$	$k = 2$	$k = 3$
R_{ax} (Å)	2.16	2.18	2.17	2.16
R_{eq} (Å)	2.20	2.22	2.20	2.20
Θ (°)	±10.7	0.0	±6.1	±11.8

Table 5.2 shows the energies of each state's minimum relative to the minimum of the ^{1}A state for both fully optimized structures and the $k = 3$ sparse grid approximation. Again, we see that our sparse grid approximation is an agreement with results computed from the fully optimized structures.

Table 5.2: Energies (kcal/mol) of each state's minimum relative to the minimum of the ^{1}A state.

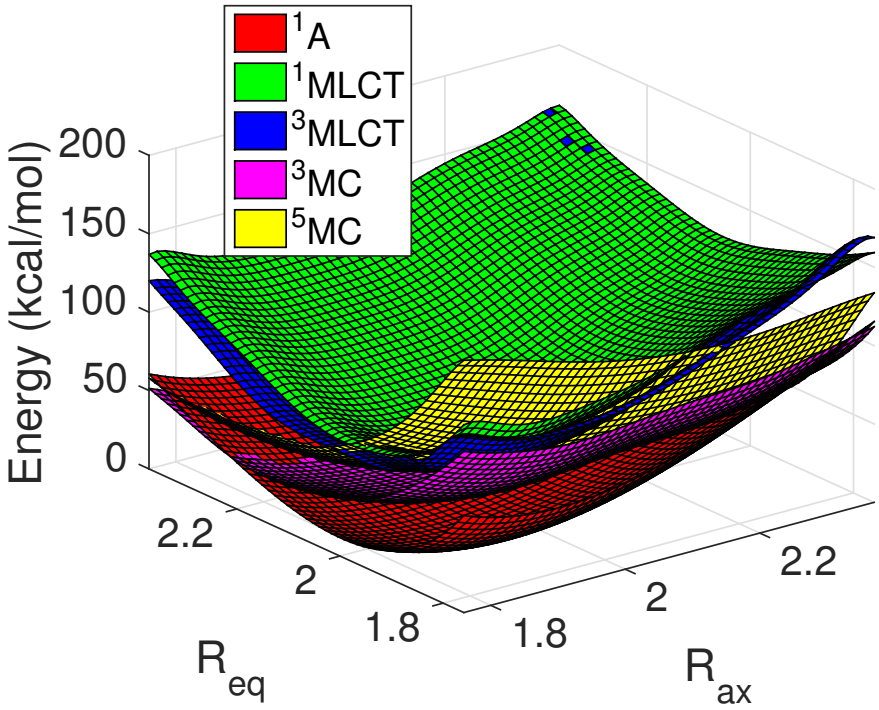
State	Optimized Energy	$k = 3$ Sparse Grid Energy
^{1}A	0.0	0.0
^{3}MC	15.27	14.17
^{5}MC	5.83	5.8
^{1}MLCT	-	61.91
^{3}MLCT	-	57.06

The use of the surrogate model enables us to visualize and explore the PES's in many different ways. Figures 5.6-5.8 show two-dimensional projections of the $k = 3$ PES's for each pair of degrees of freedom, fixing the third degree of freedom at its $k = 3$ optimized value for the ^{1}A state from Table 5.1. Figure 5.10 shows a one-dimensional projection of the $k = 3$ PES's where $R_{ax} = R_{eq}$ and $\Theta = 0°$. Finally, Figure **??** shows the stable (i.e. lowest energy) electronic state as a function of both bond lengths and a rocking angle of 0°.

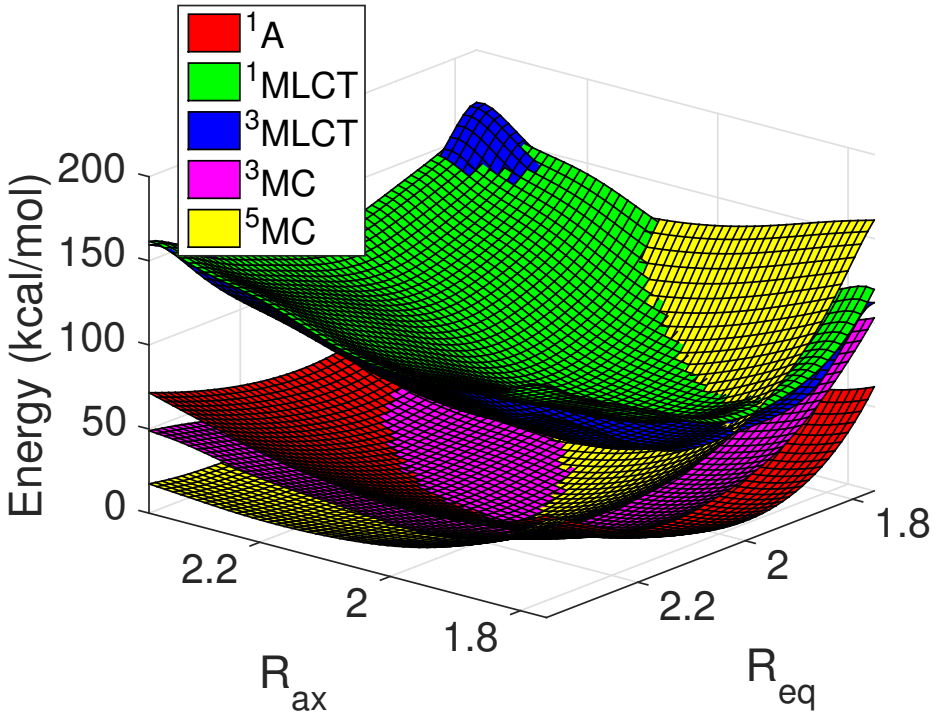
As we expect, the ^{1}MLCT and ^{3}MLCT states lie close together at all points in our domain, with the ^{1}MLCT state energy slightly higher than the ^{3}MLCT state energy. These are both higher in energy than the ^{3}MC and ^{5}MC states at all points in our domain except for a small region in the corner where axial and equatorial bond lengths are both less than ≈ 1.9 Å. Note also that the ^{3}MC and ^{5}MC states are both lower in energy than the ^{1}A state when both axial and equatorial bond lengths are elongated. There is a region of the domain where the ^{3}MLCT energy is higher than the ^{1}MLCT (see top corner of Figure 5.6A). This is not physical, but rather a result of Runge's phenomenon, a problem where interpolating polynomials oscillate near the boundary of the interpolation domain. These oscillations are reduced by our use of Chebyshev sparse grid points, but the phenomenon is still evident in our results because the two MLCT energies are very close together in that region. Regardless, we do not expect any dynamics to venture towards that area of the domain, so the small oscillation is of no concern.

It is evident from Figures 5.6-5.8 that the PESs of various electronic states are not energetically well separated, but there are a number of crossing seams between various pairs of PESs present. Most notably, ^{5}MC electronic surface crosses every single PES investigated. Therefore,

depending on the actual structure of the $[Fe(tpy)_2]^{2+}$, ^{5}MC electronic state can be either the lowest or the highest energy electronic state of this complex among those investigated. Interaction of the ^{5}MC surface with every PES investigated suggests that ^{5}MC electronic state plays a key role in the ISC cascade and the ability to control the overall shape of its PES via various structural modifications will translate into our ability to control the ISC cascade. Overall, it is apparent that the length of the metal-ligand bonds has a large impact on the ordering of the MC and MLCT states in Fe(II)-polypyridines: the short bond lengths tend to stabilize MLCT states, while the long bond lengths stabilize the MC states. This can be exploited in construction of the complexes with desired energetic ordering of electronic states.



(A)



(B)

Figure 5.6: Two views of the ^{1}A, ^{1,3}MLCT, ^{3}MC, and ^{5}MC electronic state PES's for $[Fe(tpy)_2]^{2+}$ as a function of axial and equatorial bond lengths.

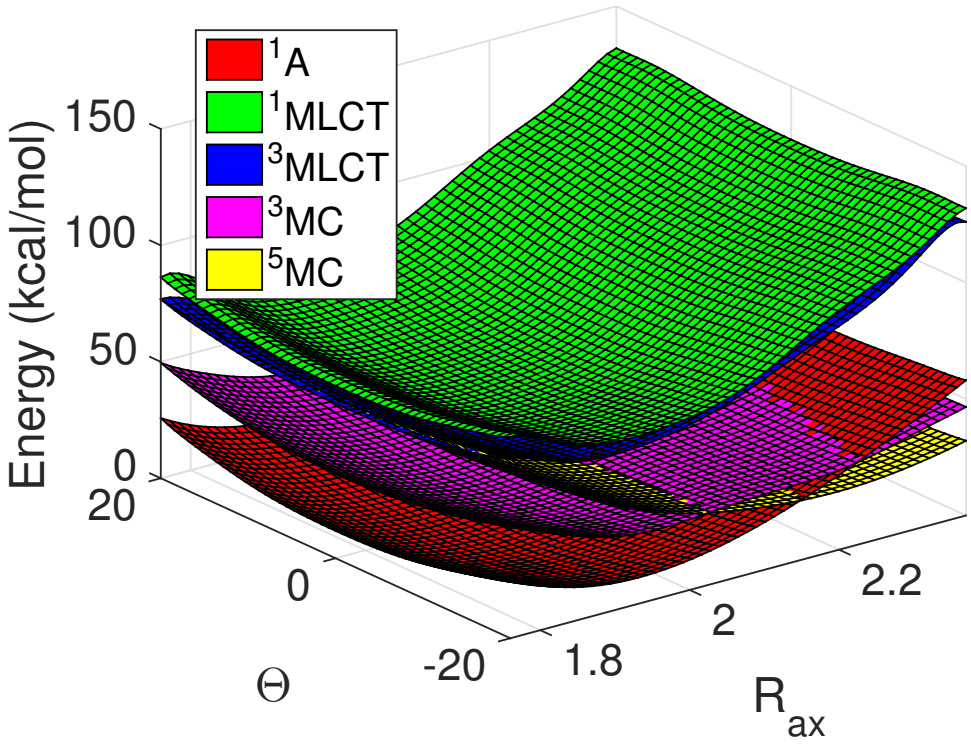


Figure 5.7: ^{1}A, ^{1,3}MLCT, ^{3}MC, and ^{5}MC electronic state PES's for $[Fe(tpy)_2]^{2+}$ as a function of axial bond lengths and rocking angle.

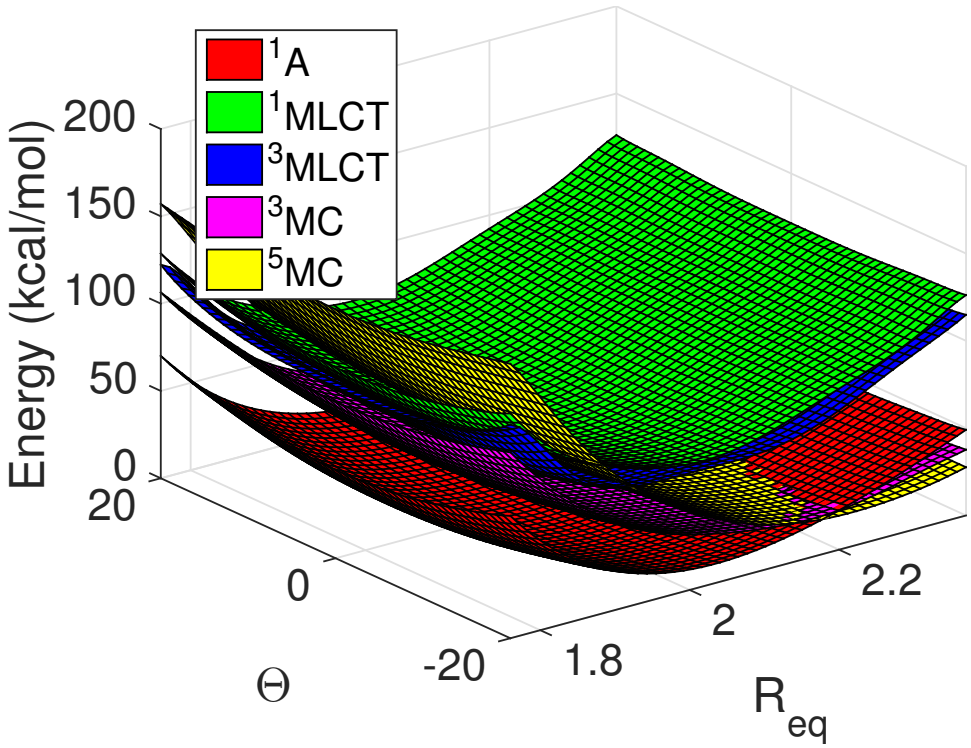


Figure 5.8: ^{1}A, ^{1,3}MLCT, ^{3}MC, and ^{5}MC electronic state PES's for $[Fe(tpy)_2]^{2+}$ as a function of equatorial bond lengths and rocking angle.

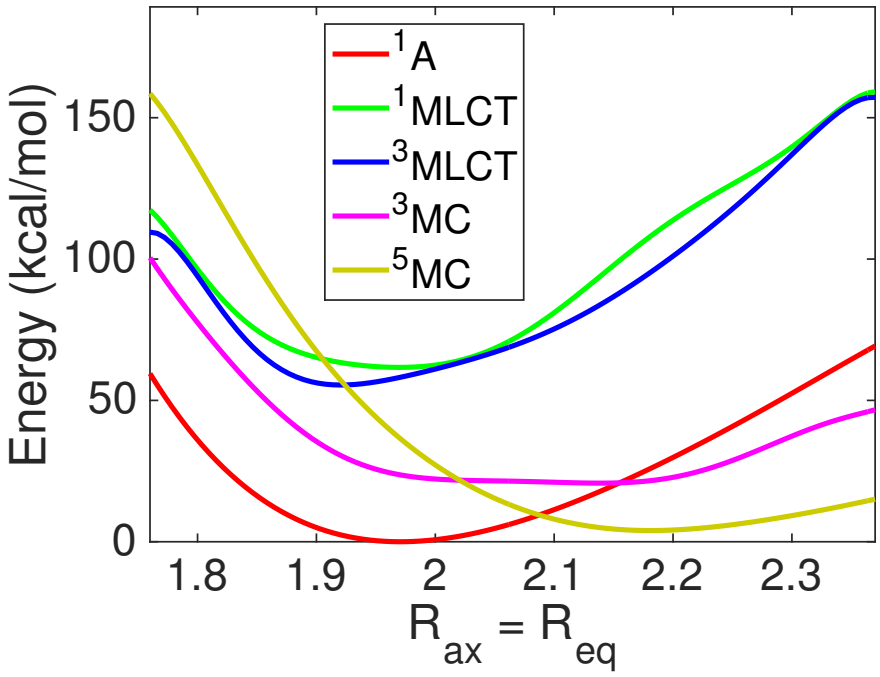


Figure 5.9: ^{1}A, ^{1,3}MLCT, ^{3}MC, and ^{5}MC electronic state PES's for $[Fe(tpy)_2]^{2+}$ as a function of equal axial and equatorial bond lengths.

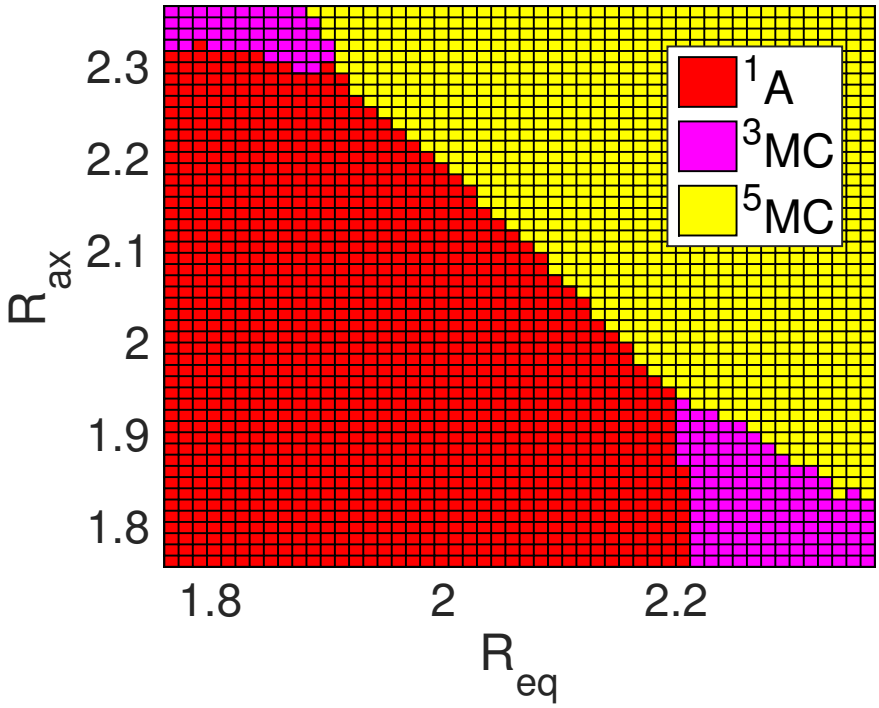


Figure 5.10: The stable electronic state for $[Fe(tpy)_2]^{2+}$ as a function of equal axial and equatorial bond lengths ($\Theta = 0°$).

Figures 5.11-5.13 show contour plot projections of the $k = 3$ PES's for each pair of degrees of freedom. All Figures show projections of the PES's for two degrees of freedom with the other degree of freedom at its $k = 3$ optimized value taken from Table 5.1. From these plots, we find large valleys of low energy when considering distortions to Θ near the energy minima. This is most evident for the ^{3}MLCT and ^{5}MC states, which have large regions of low energy around their minima. Also, the shallow symmetric double wells potential for the ^{3}MLCT and ^{5}MC states about Θ are clearly visible. These low energy regions are within the error of the DFT methodology, and so their energy cannot be considered significantly different than that of the minima. This is important when considering dynamics on the PESs, as there is very little barrier to prevent the molecular conformations from traversing these regions of the PESs. This could potentially lead to dynamics that differ quite significantly from other Fe(II)-polypyridines which lack the energetic capacity for these angular distortions. This is also important as we calculate the minimum-energy crossing points (MECPs) of the surfaces, where intersystem crossing is likely to occur. By considering as many design parameters as possible to explore the distortions between the relevant states, we insure that the surfaces and MECPs we calculate are reasonable approximations to the exact values (complete surface) at the same level of theory.

These results suggest that there is a very low energy barrier associated with the rocking motion of the ligand in these complexes. The distortion in Θ is one of the most pronounced structural differences between the ^{1}A and ^{5}MC minima, where the Θ changes from 0° in ^{1}A to 10.4° in ^{5}MC. Making this distortion less energetically favorable via structural modifications could kinetically slow down the ISC process.

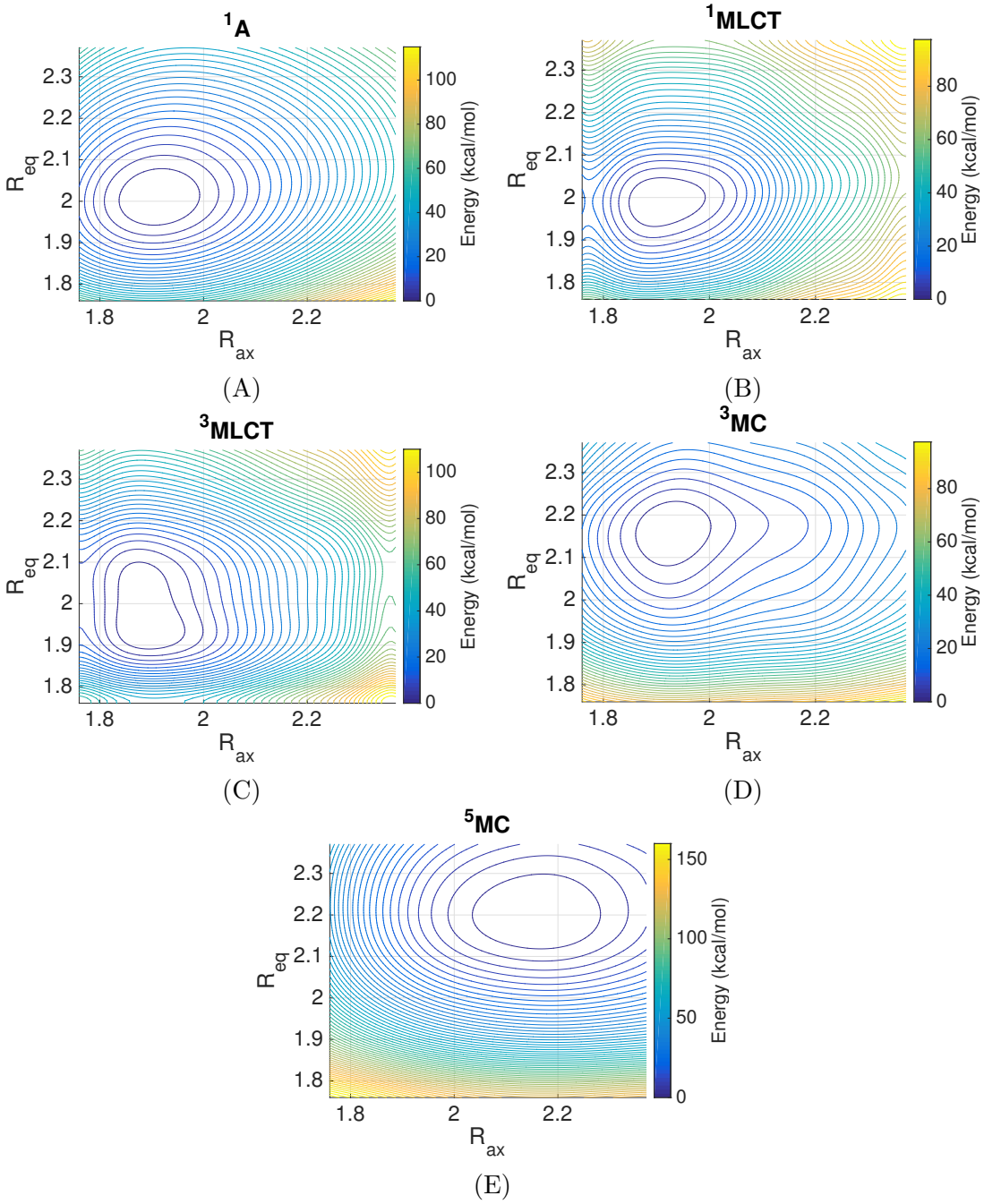


Figure 5.11: Contour plots of the (A) ^{1}A, (B) ^{1}MLCT, (C) ^{3}MLCT, (D) ^{3}MC, and (E) ^{5}MC electronic state PES's for $[Fe(tpy)_2]^{2+}$ as a function of axial and equatorial bond lengths.

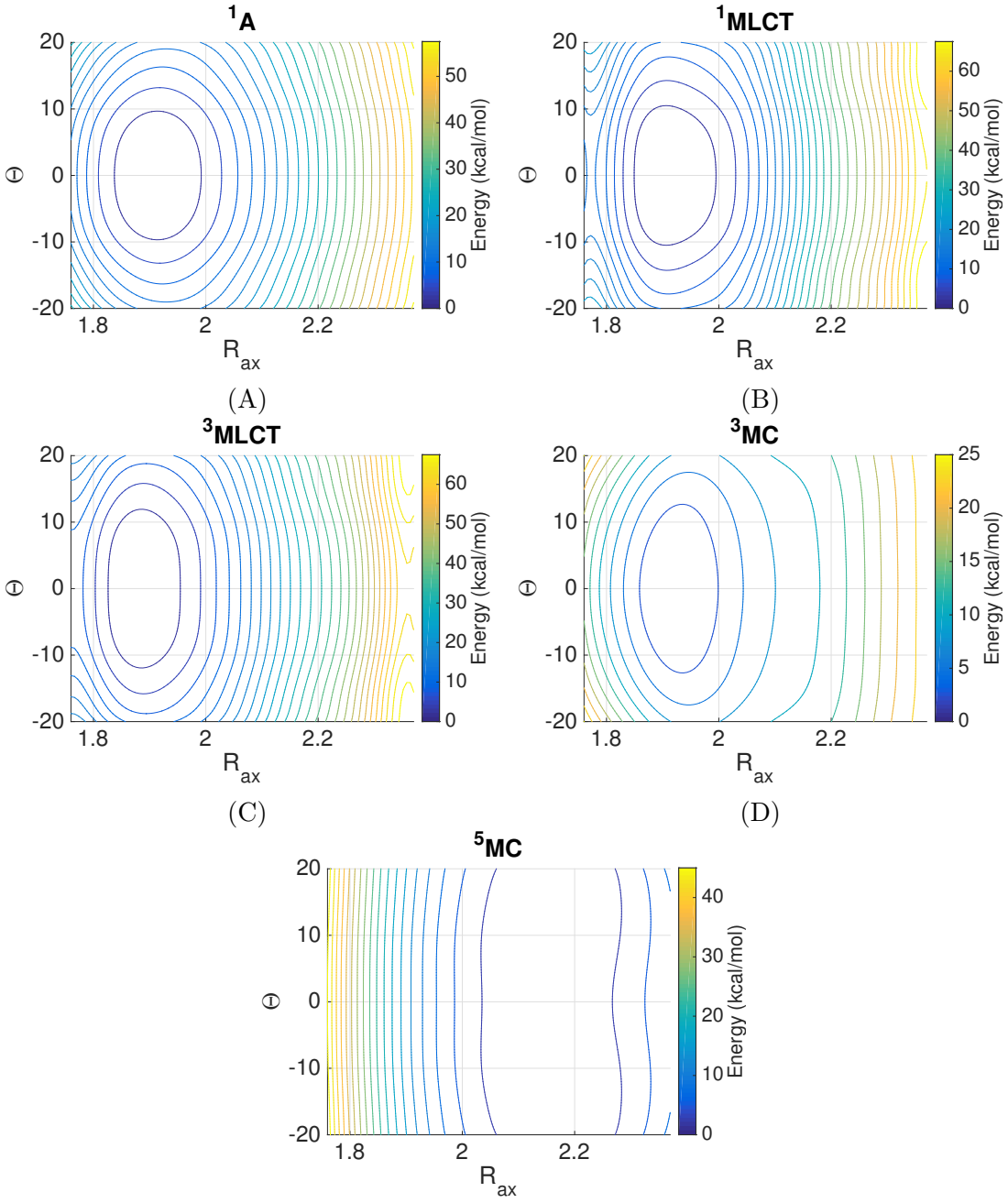


Figure 5.12: Contour plots of the (A) ^{1}A, (B) ^{1}MLCT, (C) ^{3}MLCT, (D) ^{3}MC, and (E) ^{5}MC electronic state PES's for $[Fe(tpy)_2]^{2+}$ as a function of axial bond length and rocking angle.

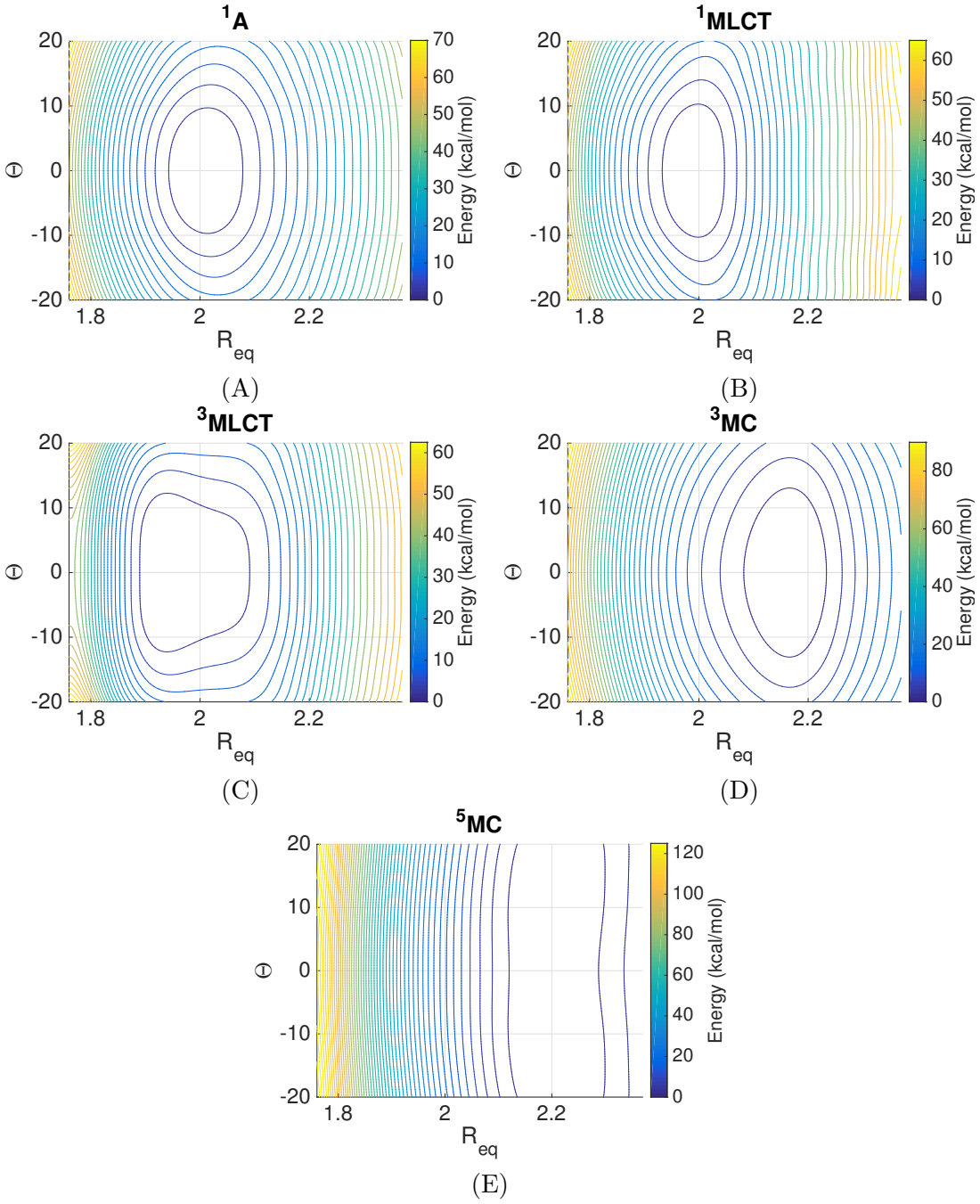


Figure 5.13: Contour plots of the (A) ^{1}A, (B) ^{1}MLCT, (C) ^{3}MLCT, (D) ^{3}MC, and (E) ^{5}MC electronic state PES's for $[Fe(tpy)_2]^{2+}$ as a function of equatorial bond length and rocking angle.

5.2.2 Minimum Energy Crossing Points and Seam of Intersection

In this section we calculate the minimum energy crossing points (MECPs) for each pair of PESs with different spin states. An MECP, the point where two PESs corresponding to different spin states intersect with minimum energy, is important because it serves as the most likely place where a transition between the two electronic states occurs [229, 230, 15]. It is necessary to compute MECPs in order to understand and quantify the physical interactions at the crossing point such as spin-orbit coupling [42].

Following the definition from [98], we begin by defining

$$\mathbf{y} := \nabla E_i(\mathbf{p}) - \nabla E_j(\mathbf{p}) \tag{5.1}$$

where $\mathbf{p}$ denotes geometric coordinates and $E_i(\mathbf{p})$ is the PES of state i. The MECP between two PESs E_i and E_j can be found by minimizing

$$\|\bar{\mathbf{g}}(\mathbf{p})\| = \|\mathbf{g}(\mathbf{p}) + \mathbf{f}(\mathbf{p})\| \tag{5.2}$$

where

$$\mathbf{f}(\mathbf{p}) = [E_i(\mathbf{p}) - E_j(\mathbf{p})]\,\mathbf{y} \tag{5.3}$$

$$\mathbf{g}(\mathbf{p}) = \nabla E_i(\mathbf{p}) - \frac{\mathbf{y}}{\|\mathbf{y}\|}\left(\nabla E_i(\mathbf{p}) \cdot \frac{\mathbf{y}}{\|\mathbf{y}\|}\right). \tag{5.4}$$

Transitions between the two PESs of different spin are most likely to occur at these points [229, 230, 15], and as such they play an important role in nonadiabatic transition state theory.

We also compute the intersection seam for each pair of PESs. The seam for two energy surfaces E_1 and E_2 is defined as the set

$$S_{1,2} = \{\mathbf{x} | E_1(\mathbf{x}) = E_2(\mathbf{x})\}. \tag{5.5}$$

Table 5.3 shows the MECP coordinates for each pair of the surfaces that intersect, along with the MECP energies relative to the minimum ground state energy. The structure and energies of the MECPs obtained from the interpolated surfaces are very similar to those of the fully optimized MECPs (see Table 5.1). The MECP for intersection of ^{1}A and ^{5}MC electronic states lies at the lowest energy, approximately 10 kcal/mol higher than the 1A minimum, with the MECPs for ^{1}A/^{3}MC and ^{3}MC/^{5}MC pairs lying only 4-4.4 kcal/mol above. These results are similar to those determined at the B3LYP* and CASPT2 levels of theory in the previous work by Papai and coworkers [159]

Figures 5.14-5.18 show plots of the entire intersection seam between these surfaces. The

Table 5.3: Minimum energy crossing points for each pair of PESs. ^{1}A is the singlet ground state, ^{3}MC is the lowest-lying triplet excited state, and ^{5}MC is the lowest-lying quintet excited state. The * indicates fully optimized MECPs that were calculated with Gaussian 09.

Surfaces	MECP (R_{ax}, R_{eq}, Θ)	MECP Energy (kcal/mol)
^{1}A/^{3}MC	(1.93, 2.20, 0.00)	14.76
^{1}A/^{5}MC	(2.03, 2.13, 0.00)	10.30
^{1}A/^{5}MC*	(2.03, 2.13, 0.00)	10.77
^{3}MC/^{5}MC	(1.95, 2.17, 0.00)	14.38
^{3}MC/^{5}MC*	(1.96, 2.15, 0.00)	15.31
^{1}MLCT/^{5}MC	(1.87, 1.92, 0.00)	65.53
^{3}MLCT/^{5}MC	(1.89, 1.94, 0.00)	57.06

MECP is shown in red, and the energy of the intersection seam is relative to the MECP energy. In each case, the intersection seam is a two-dimensional manifold that slices through our three-dimensional domain. Each intersection seam contains a large area around the MECP in which the energy is less than 5 kcal/mol higher than the MECP energy, with 2.5 and 5.0 kcal/mol contours plotted in white and red, respectively. With the exception of ^{1}MLCT/^{5}MC surface (Figure 5.16, this low-energy area around the MECP is situated primarily along the Θ coordinate, indicating that large areas of the intersection seams around the MECP are energetically accessible via the ligand rocking motion. Consequently, the state crossing may occur for a number of different conformations around the MECP, and not necessarily always at the MECP geometry.

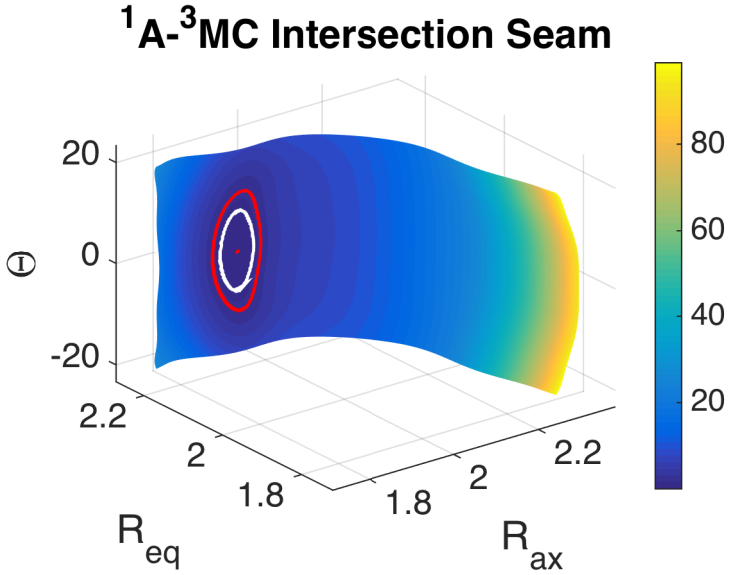


Figure 5.14: ^{1}A/^{3}MC intersection seam for $[Fe(tpy)_2]^{2+}$.

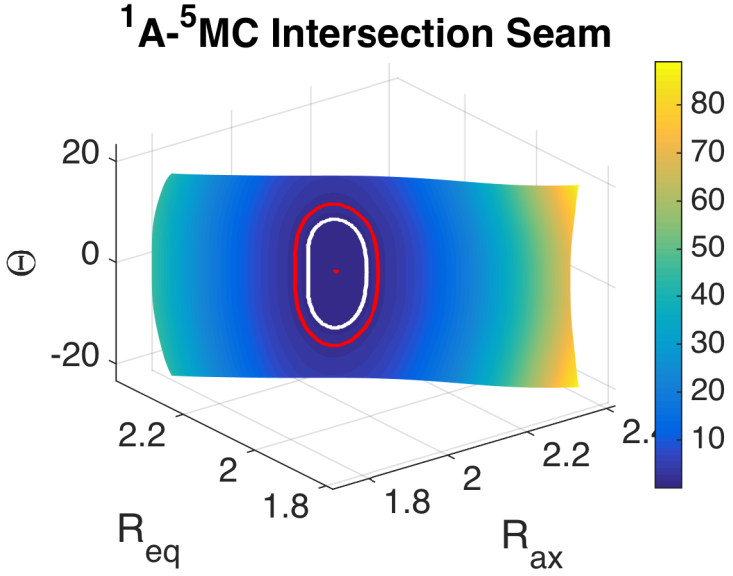


Figure 5.15: ^{1}A/^{5}MC intersection seams for $[Fe(tpy)_2]^{2+}$.

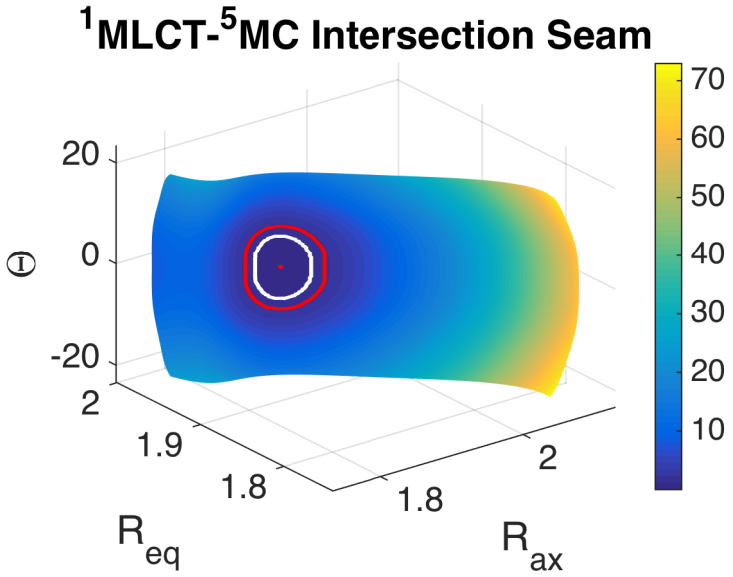


Figure 5.16: ^{1}MLCT/^{5}MC intersection seam for $[Fe(tpy)_2]^{2+}$.

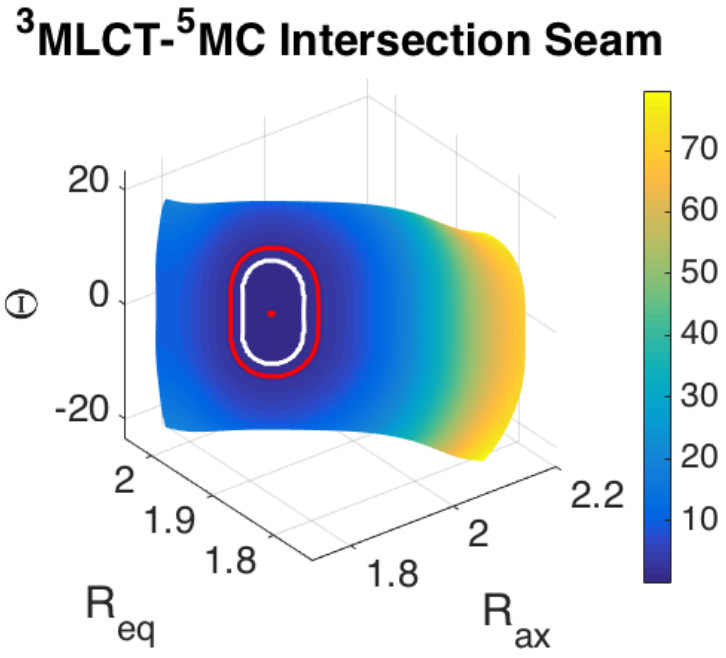


Figure 5.17: ^{3}MLCT/^{5}MC intersection seam for $[Fe(tpy)_2]^{2+}$.

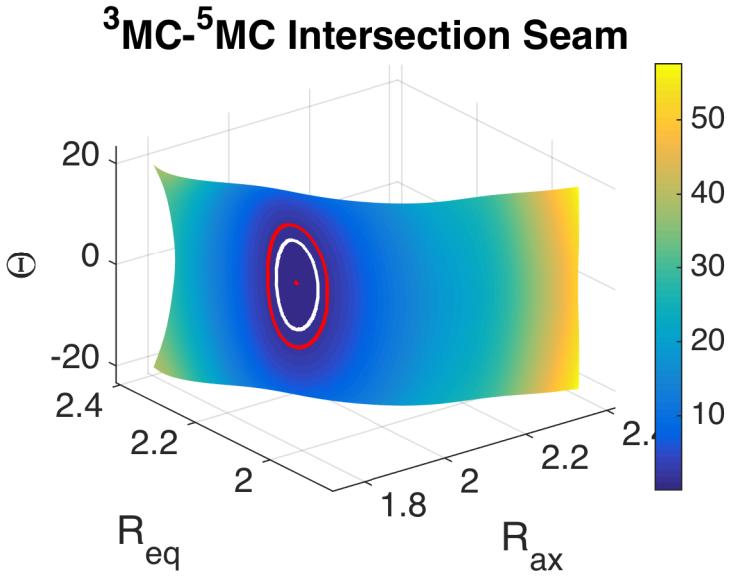


Figure 5.18: ^{3}MC/^{5}MC intersection seam for $[Fe(tpy)_2]^{2+}$.

Finally, we computed the minimum energy pathway (MEP) for each pair of surfaces that intersect. An MEP connects two separate surfaces' minimum via the MECP using the path with the smallest change in energy [190]. Figures 5.19-5.23 show the MEPs for each of the pairs of surfaces that intersect. We computed these MEPs by computing two steepest descent reaction paths: one from the MECP to the minimum of the first surface, and another from the MECP to the minimum of the second surface. With the exception of the the ^{3}MLCT/^{5}MC MEP (Figure 5.23), the rocking angle remained at $\Theta = 0°$ for the entire reaction path. Consequently, we were able to project these MEPs onto two-dimensional potential energy surfaces (see Figures 5.19-5.22).

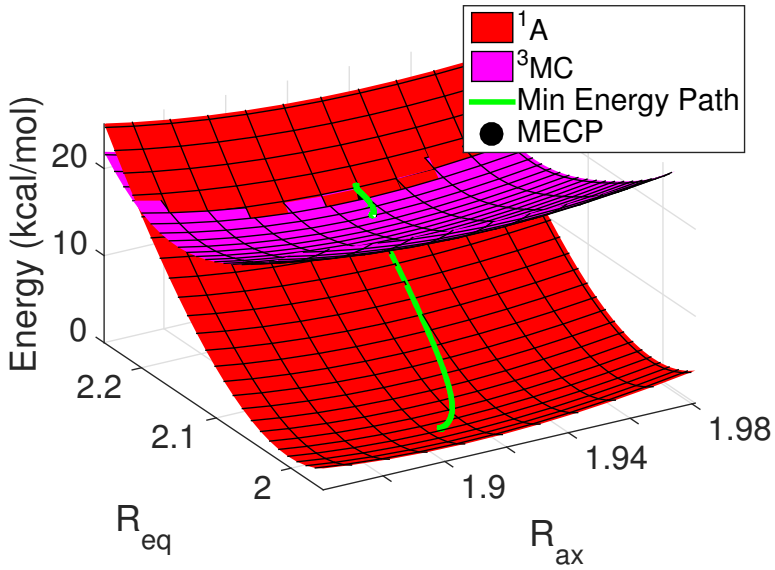


Figure 5.19: ^{1}A/^{3}MC minimum energy pathway for $[Fe(tpy)_2]^{2+}$.

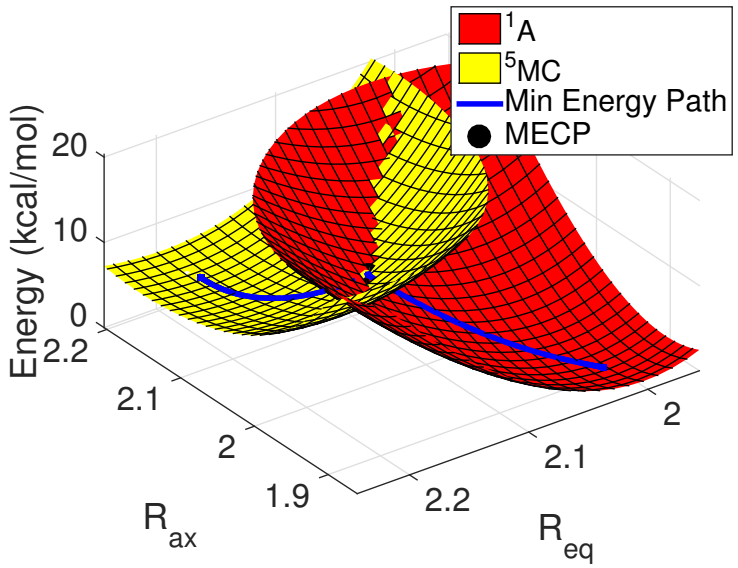


Figure 5.20: ^{1}A/^{5}MC minimum energy pathway for $[Fe(tpy)_2]^{2+}$.

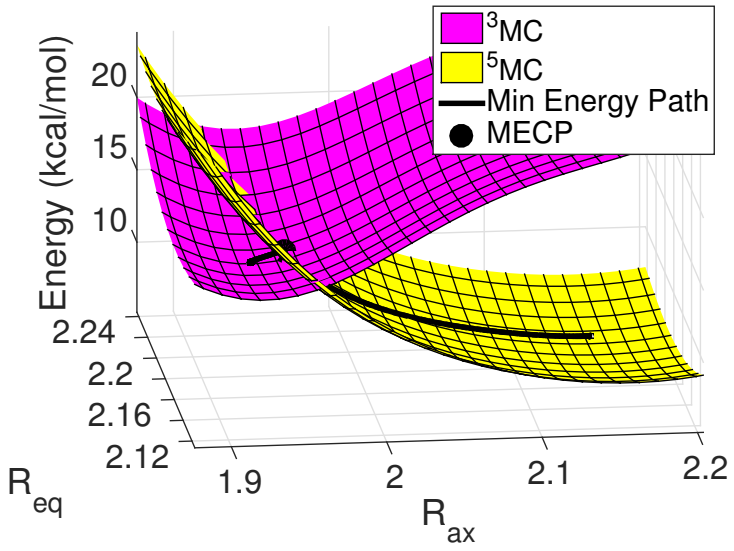


Figure 5.21: ^{3}MC/^{5}MC minimum energy pathway for $[Fe(tpy)_2]^{2+}$.

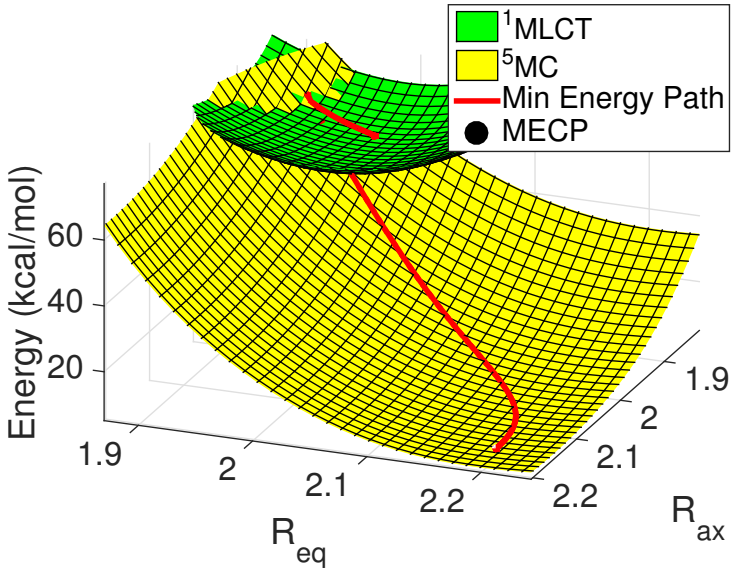


Figure 5.22: ^{1}MLCT/^{5}MC minimum energy pathway for $[Fe(tpy)_2]^{2+}$.

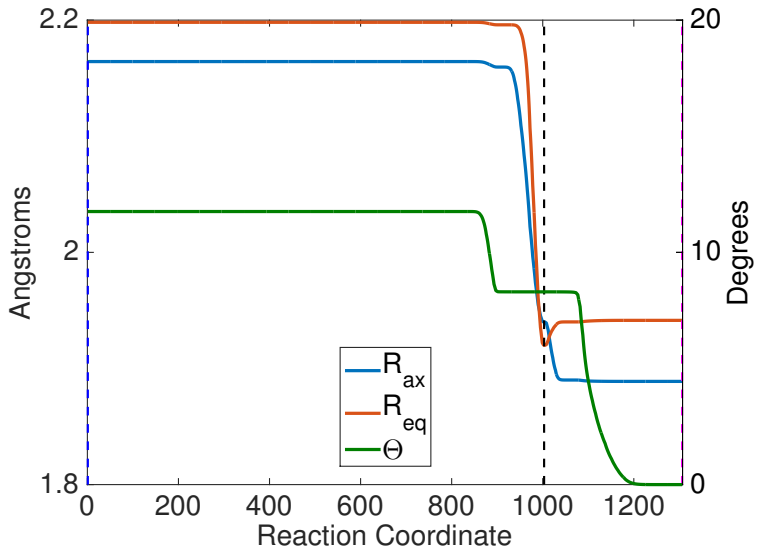


Figure 5.23: ^{3}MLCT/^{5}MC minimum energy pathway for $[Fe(tpy)_2]^{2+}$.

The construction of these reduced PESs is the first step towards performing dynamics of the photocycle for the complex, which could aid in understanding its unique strong-field LIESST behavior. Further investigations into the transitions between states and the deactivation pathways for the complex will hopefully lead to insight into the unique dynamics observed for the [Fe(tpy)2]2+ complex. This and future work will also aid in establishing design principles and structure-function relationships in Fe(II)-polypyridine complexes, to afford a wider range of tunability towards numerous photophysical applications.

5.3 Nonadiabatic Dynamics

In this section we discuss the methodologies needed to describe the complex dynamics of Fe(II)-polypyridine systems. These dynamics are called *nonadiabatic dynamics*, and require the inclusion of more quantum effects than the *adiabatic dynamics* described in Chapter 4.

There are several methods for describing nonadiabatic dynamics at or near intersections of two PESs intersect [212, 214]. There are two types of intersections: *conical intersections* take place between states of the same spin, and *intersystem crossings* take place between states of different spin [234]. The work presented in this Chapter focuses primarily on the latter. The approaches we will employ for nonadiabatic dynamics are nonadiabatic transition state theory (TST) [96, 97] and Tully's fewest switches trajectory surface hopping method [210]. TST provides a simple time-independent approach for computing nonadiabatic transition probabilities between two states of different spin. Nonadiabatic dynamics at an intersystem crossing between two PES's is described by the two surface's minimum energy crossing point (MECP), the lowest energy geometry on the seam of intersection between the two surfaces. In Tully's more complicated time-dependent method, nuclear dynamics are described by classical equations of motion of electronic PESs. At any given time, trajectories can "hop" from one surface to another. The probability of a hop from one state to another is calculated from either the spin-orbit or nonadiabatic coupling and the relative energy between the two states.

There are other methods for treating nonadiabatic dynamics. The ab initio multiple spawning (AIMS) method [18] interprets the nuclei as frozen Gaussian wave packets evolving on an electronic PES. As the wave packet propagates, it can separate, or spawn, depending on the nonadiabatic coupling between the electronic states. Another alternative is the the Ehrenfest method [141] where trajectories evolve on a optimally weighted average of PESs instead of hopping between them. Ref. [214] offers a great review of nonadiabatic dynamics and the methods for which to study them, including both the Ehrenfest and surface hopping methods. In the next section we describe Tully's fewest switches trajectory surface hopping method [210], as it will be our method of choice in future work.

5.3.1 Nonadiabatic Transition State Theory

Nonadiabatic transition state theory (TST) [96, 97] can account for the kinetics of reactions involving intersystem crossing. These reactions are nonadiabatic in the sense that they occur on more than one PES, with transformation from reactants to products requiring the system to "hop" from the PES corresponding to the initial spin state onto that corresponding to the product state for reaction to occur. TST is a time-independent method that yields reactivity rate constants on single PESs for each step of nuclear motion. The rate coefficient $k_{i\to j}$ of a

spin-forbidden reaction at a given internal energy E is

$$k_{i\to j} = \frac{k_g T}{h} \exp\left(\frac{\Delta E_{MECP}}{RT}\right) P_{hop} \tag{5.6}$$

where ΔE_{MECP} is the energy difference between the two surfaces at the MECP, k_g is the Boltzmann constant, R is the molar gas constant, T is the temperature, and P_{hop} is the probability of a transition between the two surfaces. This transition probability is calculated using the Lendau-Zener formula [235, 153, 225]

$$P_{hop} = 1 - \exp\left(\frac{-2\pi H_{ij}^{SO}}{\nu \Delta F}\right) \tag{5.7}$$

where ν is the velocity of the system as it passes through the MECP, H_{ij}^{SO} is the spin-orbit coupling-derived off-diagonal Hamiltonian matrix element between the two electronic states defined as

$$H_{ij}^{SO} = \left\langle \phi_i | \hat{H}_{SO} | \phi_j \right\rangle, \tag{5.8}$$

and ΔF is the relative slope of the two surfaces at the MECP [96, 97].

5.3.2 Tully's "Fewest Switches" Surface Hopping

Surface hopping [106] is a popular approach for modeling nonadiabatic dynamics. The method is a time-dependent mixed quantum mechanical/classical mechanical technique that is designed to more accurately treat quantum mechanical effects of dynamics in regions of PESs where the Born-Oppenheimer approximation breaks down (i.e. conical intersections and intersystem crossings). As opposed to the reaction path method of Chapter 4 where trajectories traveled on one PES, in the surface hopping method trajectories are allowed to "hop" between surfaces at any given time. Tully's "Fewest Switches" Surface Hopping method [210] is perhaps the most widely used surface hopping algorithm, and we describe the method in this section.

Let $\mathbf{r}$ and $\mathbf{R}$ denote the electronic and atomic coordinates, respectively. In [210], Tully writes the total Hamiltonian of the system as

$$\hat{H} = \hat{T}_R + \hat{H}_0(\mathbf{r}, \mathbf{R}), \tag{5.9}$$

where $\hat{H}_0(\mathbf{r}, \mathbf{R})$ is the electronic Hamiltonian for fixed atomic positions and $\hat{T}_R$ is the atomic motion kinetic energy operator.

We first choose an orthonormal set of M electronic basis functions $\phi_j(\mathbf{r}; \mathbf{R})$ that depend parametrically on the atomic positions. These may be, for example, wave functions derived from

the time-independent Schrödinger equation within Born-Oppenheimer approximation

$$\hat{H}_0\phi_i = \mathcal{E}_i\phi_i \tag{5.10}$$

for eigenstates (electronic states) $j = 1, 2, \ldots, M$. Next, define matrix elements of the electronic Hamiltonian

$$V_{ij}(\mathbf{R}) = \left\langle \phi_i(\mathbf{r};\mathbf{R}|\hat{H}_0(\mathbf{r},\mathbf{R})|\phi_j(\mathbf{r};\mathbf{R}) \right\rangle \tag{5.11}$$

where we use bra-ket notation introduced in Chapter 3. Note that

$$\mathcal{E}_i = V_{ii}. \tag{5.12}$$

Finally, we define the "nonadiabatic coupling vector"

$$\mathbf{d}_{ij}(\mathbf{R}) = \langle \phi_i(\mathbf{r};\mathbf{R})|\nabla_{\mathbf{R}}|\phi_j(\mathbf{r};\mathbf{R})\rangle \tag{5.13}$$

where the gradient is defined with respect to the atomic coordinates. This vector can also be computed from the off-diagonal Hellmann-Feynman forces [59]

$$\mathbf{d}_{ij}(\mathbf{R}) = \frac{\left\langle \phi_i(\mathbf{r};\mathbf{R}|\nabla_{\mathbf{R}}\hat{H}_0|\phi_j(\mathbf{r};\mathbf{R}) \right\rangle}{\mathcal{E}_i - \mathcal{E}_j}, \quad i \neq j. \tag{5.14}$$

Tully's surface hopping method assumes that there is a continuous function such that $\mathbf{R} = \mathbf{R}(t)$ where t is time. For this work we employ the continuous steepest descent equation from our reaction path method in Chapter 4

$$\dot{\mathbf{R}} = -\nabla_{\mathbf{R}}\mathcal{E}_j. \tag{5.15}$$

Consequently, the electronic Hamiltonian $\hat{H}_0(\mathbf{r};\mathbf{R})$ is now a time-dependent operator $\hat{H}_0(\mathbf{r};\mathbf{R}(t))$. Another popular option for the classical trajectories is $\mathbf{R}_m$ is

$$\ddot{\mathbf{R}} = -\frac{1}{M_m}\nabla_{\mathbf{R}}\mathcal{E}_j \tag{5.16}$$

where M_m is the associated mass. This is the equation of choice for the Newton-X software package [11, 12] and the solution is often approximated with the Velocity Verlet algorithm [205] (see Appendix A.5).

Consider the time-dependent electronic Schrödinger equation

$$i\hbar\frac{\partial}{\partial t}\psi(\mathbf{r},\mathbf{R},t) = \hat{H}_e\psi(\mathbf{r},\mathbf{R},t) \tag{5.17}$$

where $\psi(\mathbf{r}, \mathbf{R}, t)$ is the wave function that describes the electronic state at time t. Tully expands this wave function in terms of the electronic eigenfunctions

$$\psi(\mathbf{r}, \mathbf{R}, t) = \sum_{j}^{M} c_j(t)\phi_j(\mathbf{r}; \mathbf{R}) \tag{5.18}$$

where the time-dependent expansion coefficients $c_j(t) \in \mathbb{C}$ are called quantum amplitudes. This approximation is commonly known as the Born-Huang expansion [26]. Substituting this expansion into the time-dependent electronic Schrödinger equation, multiplying on the left by $\phi_k(\mathbf{r}, \mathbf{R})$, and integrating over $\mathbf{r}$ yields

$$i\hbar\dot{c}_k = \sum_{j}^{M} c_j \left(V_{kj} - i\hbar\dot{\mathbf{R}} \cdot \mathbf{d}_{kj} \right). \tag{5.19}$$

In this form two terms promote transitions between electronic states: the off-diagonal elements of V_{kj} and the nonadiabatic coupling term $\mathbf{R} \cdot \mathbf{d}_{kj}$.

It is often convenient to write Equation 5.19 in its equivalent density matrix form. Letting $a_{kj} = c_k c_j^*$ and using the properties of a set of orthonormal basis functions ϕ_i, Equation 5.19 becomes

$$i\hbar\dot{a}_{kj} = \sum_{i} \left[a_{ij} \left(V_{ki} - i\hbar\dot{\mathbf{R}} \cdot \mathbf{d}_{ki} \right) - a_{ki} \left(V_{ij} - i\hbar\dot{\mathbf{R}} \cdot \mathbf{d}_{ij} \right) \right]. \tag{5.20}$$

Here, a_{kk} are the occupation probabilities of the instantaneous adiabatic eigenstates and a_{kj}, $k \neq j$ define the quantum coherence [59]. For a swarm of N trajectories, the number of trajectories on a PES j at time t is $a_{jj}(t) \cdot N$. The electronic state populations satisfy

$$\dot{a}_{kk} = \sum_{l \neq k} b_{kl}$$

where

$$b_{kl} = 2\hbar^{-1}\mathrm{Im}(a_{kl}^* V_{kl}) - 2\mathrm{Re}(a_{kl}^* \mathbf{R} \cdot \mathbf{d}_{kl}).$$

Our task is to simultaneously integrate Equations 5.15 and 5.20. Even though the time-dependent wave function ψ is mixed state, the forces on the classical subsystem are determined by a single "occupied" state. Tully's fewest switches algorithm applies to the standard surface-hopping procedure [215, 191], where trajectories evolve on a single PES, not some weighted average, with the possibility of sudden jumps from one state to another that occur instantaneously. As the name suggests, Tully's fewest switches algorithm minimizes the number of state

switches. The method is outlined in three steps:

1. The system is assigned initial conditions, including positions and velocities of all the atoms and the initial electronic density matrix elements a_{kj}. The initial wave functions are usually the reference state wave function (e.g. ground state wave function) [170].

2. The classical mechanical equations of motion for the atoms (Equation 5.15) on the current PES V_{kk} are integrated for a sufficiently small time interval Δt.

3. For conical intersections, the switching probabilities g_{kj} from the current electronic state k to all other states j are computed from the density matrix elements via

$$g_{kj}(t) = \max\left(0, \frac{\Delta t b_{jk}}{a_{kk}}\right). \tag{5.21}$$

   For intersystem crossings, on the other hand, probabilities are computed using the Lendau-Zener formula [235, 153, 225] formula given in Equation 5.7. Then, a number ξ is drawn with uniform probability from the interval $(0, 1)$ to determine if a switch to any state j will occur.

4. If a switch to state k' occurs, the trajectory will now evolve on the PES $V_{k'k'}$. Note that any change in energy must be accounted for to conserve total energy. Tully suggests making this adjustment to the component of velocity in the direction of the nonadiabatic coupling vector $\mathbf{d}_{kk'}(\mathbf{R})$ at the position of the transition $\mathbf{R}$. If the velocity reduction required is greater than the component of velocity to be adjusted, then the switch is not invoked. Return to step 2.

Steps 2-4 are repeated until the trajectory reaches whatever stopping criteria specified by the application. We will apply this method to study nonadiabatic dynamics of PESs of $[\mathrm{Fe(tpy)_2}]^{2+}$.

Adiabatic vs. Diabatic Representation

For nonadiabatic dynamics there are two common choices of representation for the basis functions ϕ_i: *adiabatic* wave functions and *diabatic* wave functions. The most common is the adiabatic or Born-Oppenheimer wave functions, which are solutions to the time-independent electronic Schrödinger equation (Equation 5.10) for fixed nuclear coordinates $\mathbf{R}$. In this representation the electronic Hamiltonian (Equation 5.11) is a diagonal matrix where $V_{kk} = \mathcal{E}_k$ and $\mathbf{d}_{kj} \neq 0$. Diabatic wave functions, on the other hand, are designed such that $\mathbf{d}_{kj} = 0$ and $V_{kj} \neq 0$ [59]. A number of different procedures for defining diabatic wave functions have been proposed, but we describe the one presented in [214] here. First we split the electronic Hamiltonian

$$\hat{H}_0 = \hat{H}_D + \hat{V}_c$$

where $\hat{V}_c$ is a small coupling term which is responsible for the splittings between adiabatic states at avoided crossings or conical intersections (i.e., where adiabatic PES's nearly intersect). The diabatic wave functions are solutions to

$$\hat{H}_D \phi_i = \mathcal{E}_i \phi_i \tag{5.22}$$

and $\mathcal{E}_i$ are the associated diabetic energies. For example, this representation can be used to treat spin transitions by letting $\hat{H}_D$ be the non-relativistic electronic Hamiltonian and $\hat{V}_c$ characterize the spin-orbit interaction as in [84]. A diabatic basis can be defined by a transformation of the adiabatic electronic basis functions such that the vector of couplings $\mathbf{d}_{ij}$ is small enough to neglect [113]. We employ the adiabatic representation, as the diabatic representation is incompatible with surface hopping [213, 92].

Analogously, there are two representations of PESs: *adiabatic* and *diabatic*. An adiabatic PES is one in which the energy for a particular electronic state is followed. On the other hand, a diabatic PES is the lowest-energy electronic state available for each set of nuclear coordinates [231]. The Born-Oppenheimer approximation implies following adiabatic surfaces, but this approximation breaks down when the molecule transitions from one electronic state to another. Figure 5.24 shows two adiabatic PESs that intersect and the resulting diabatic PES.

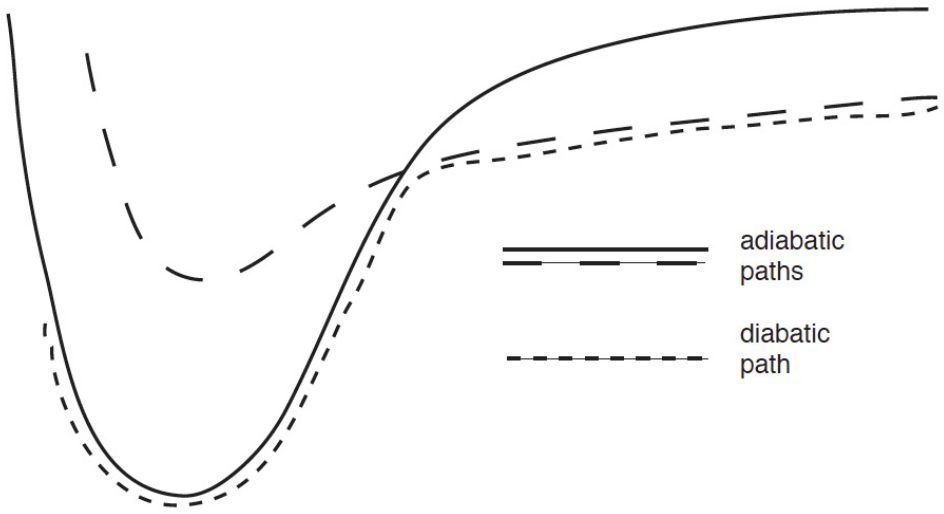


Figure 5.24: A simple one-dimensional example of two adiabatic paths that intersect and the resulting diabatic path (*image taken from [231], Chapter 20, pg. 174*).

Propagation of Quantum Amplitudes

Propagation of the quantum amplitudes c_k can be carried out in a variety of different ways. We summarize three here that are available options in the Newton-X software package [11, 12]. First let

$$\sigma_{ij} = \mathbf{d}_{ij} \cdot \dot{\mathbf{R}}. \tag{5.23}$$

The three methods are:

1. Explicit evaluation of the nonadiabatic coupling vectors $\mathbf{d}_{ij}$ via Equation 5.13.

2. If the nonadiabatic coupling vectors are not available $\sigma_{ij}(t)$ can be approximated by

$$\sigma_{ij}(t) \approx \frac{1}{4\Delta t}(3S_{ij}(t) - 3S_{ji}(t) \tag{5.24}$$

$$- S_{ij}(t - \Delta t) + S_{ji}(t - \Delta t) \tag{5.25}$$

   where $S_{ij}(t) = \langle \phi_i(t - \Delta t) | \phi_j(t) \rangle$ are wavefunction overlaps between different time steps [95, 167].

3. Finally, the local diabatization approach [83, 168] allows one to obtain c_k without explicitly evaluating σ_{jk}. In this case,

$$\mathbf{c}(t + \Delta t) = T^{-1} \exp\left(-i \frac{V(t) + TV(t + \Delta t)T^{-1}}{2\hbar}\right) \tag{5.26}$$

   where again V is the diagonal matrix of adiabatic energies and T is an adiabatic-to-diabatic transformation constructed by a Löwdin orthogonalization of the $S(t + \Delta t)$ overlap matrix.

5.4 Nonadiabatic Dynamics with Surrogate Models

Given means to calculate nonadiabatic coupling vectors in Equation 5.13 and spin-orbit couplings in Equation 5.8, we can extend our reaction path following method from Chapter 4 to include nonadiabatic dynamics. To do so, we propose approximating these terms with Smolyak's sparse grid interpolation algorithm. Instead of explicitly evaluating the nonadiabatic coupling vectors and spin-orbit couplings during Tully's surface-hopping dynamics, we instead evaluate a surrogate.

As before, we begin by choosing design coordinates $\mathbf{x} \in \mathbb{R}^d$ such that $\mathbf{R} = (\mathbf{x}, \xi)$ and

approximating the adiabatic PESs for each electronic state of interest to obtain

$$\hat{E}_j(\mathbf{x}) = \mathcal{A}(q,d)[E_j](\mathbf{x}) \tag{5.27}$$

where

$$E_j(\mathbf{x}) = \min_{\xi} \mathcal{E}_j(\mathbf{x},\xi). \tag{5.28}$$

Next we use Smolyak's sparse grid interpolation algorithm to approximate the nonadiabatic coupling vectors

$$\hat{\mathbf{d}}_{ij}(\mathbf{x}) = \mathcal{A}(q,d)[\mathbf{d}_{ij}(\mathbf{R})] \tag{5.29}$$

and the spin orbit couplings

$$\hat{H}_{ij}^{SO}(\mathbf{x}) = \mathcal{A}(q,d)[H_{ij}^{SO}(\mathbf{R})]. \tag{5.30}$$

In the construction of these three surrogates the remainder variables ξ are taken to be

$$\xi = \operatorname*{argmin}_{\xi} \mathcal{E}_0(\mathbf{x},\xi).$$

Finally, Tully's surface hopping algorithm can be implemented by simultaneously integrating

$$\dot{\mathbf{x}} = -\nabla \hat{E}_j(\mathbf{x}) \tag{5.31}$$

and

$$i\hbar \dot{a}_{kj} = \sum_i \left[a_{ij} \left(V_{ki} - i\hbar \dot{\mathbf{x}} \cdot \hat{\mathbf{d}}_{ki} \right) - a_{ki} \left(V_{ij} - i\hbar \dot{\mathbf{x}} \cdot \hat{\mathbf{d}}_{ij} \right) \right] \tag{5.32}$$

where hopping probabilities are computed with

$$P_{hop} = 1 - \exp\left(\frac{-2\pi \hat{H}_{ij}^{SO}}{\nu \Delta F} \right). \tag{5.33}$$

Recall that since we are using an adiabatic representation of the PESs, V is a diagonal matrix with $V_{ii} = \hat{E}_i$.

Currently, we lack the tools required to calculate diabatic and spin-orbit couplings at the DFT level of theory. Nonadiabatic dynamics between states of different spin have been performed before, however. Zaari et al. studied intersystem crossing between triplet and singlet

spin-sates of SiH_2 in [234] and computed spin-orbit couplings with the complete active space configuration interaction method (CAS-CI) with singlet-triplet state averaged complete active space self-consistent field method (CASSCF) orbitals, and a partial 2-electron and full 1-electron spin-orbit Hamiltonian (HSO2P) [66, 65]. Several schemes for calculating these nonadiabatic coupling vectors within TDDFT are available, provided that one of the states is the ground state. In [207], Tavernelli et al. studied intersystem crossing of $[Ru(bpy)_3]^{2+}$ using the linear response TDDFT (LR-TDDFT) method from [208] to compute nonadiabatic coupling vectors with the CPMD software package (`http://www.cpmd.org/`), and the method of Wang et al. [221] to compute spin-orbit couplings with the ADF2009.01 software package (`http://www.scm.com/`). No methods for computing spin-orbit or diabatic couplings for DFT wave functions are available in Gaussian 09 [72].

However, our collaborators are working closely with Dr. Sergey Varganov of the University of Nevada, Reno, who is actively researching efficient methods to compute spin-orbit couplings at the DFT level of theory. Once these methods have been developed we can move forward in implementing our surrogate model for Tully's surface hopping algorithm and studying the nonadiabatic dynamics of Fe(II)-polypyridines. These are goals of future work related to this dissertation and will be carried out by another student.

Chapter 6

Conclusion

In this dissertation we have used Smolyak's sparse grid interpolation algorithm [198, 13, 118] to study reaction path dynamics and potential energy surfaces (PESs) of complex molecules. Sparse grids are an effective means by which to study these systems, as often of the computational cost of constructing entire PESs often limits the number of degrees of freedom and/or the size of the molecule one can study. Sparse grids optimize the ratio of invested storage and computation time to approximation accuracy [37]. In this sense, they give us the most accurate approximation using the least amount of grid points. In Chapter 2 we discussed the history of sparse grids, and detailed the specific interpolation algorithm we use as first proposed by Barthelmann, Novak, and Ritter [13]. We developed an implementation of this algorithm specifically for this application that allows us to compute several thousand reaction paths simultaneously and efficiently. To do so, we utilized a reformulation of Smolyak's algorithm by Judd and coworkers [118] that removes redundant calculations Smolyak's original formulation. We expanded upon this reformulation by including analytical gradients, recursively computing the basis polynomials, and vectorizing the computation of tensor products in MATLAB (see Appendix A.1.3 for details) [152]. We then demonstrated that our implementation outperforms MATLAB's Sparse Grid Interpolation Toolbox [121, 122, 124].

After giving an introduction to quantum and computational chemistry in Chapter 3, we presented our surrogate reaction path following method [151] in Chapter 4. There are several benefits to using our surrogate model instead of traditional reaction path methods. First, the use of sparse grids easily enables one to increase the number of degrees of freedom in reaction path dynamics. Second, the sparse grid points used by Smolyak's algorithm are uniquely determined a priori, meaning all required electronic structure calculations can be performed trivially in parallel. Third, the sparse grids we employ are nested, so one can reuse electronic structure calculations to construct a higher order approximation. Finally, our implementation of Smolyak's algorithm allows one to continuously follow several different reaction paths simul-

taneously. We demonstrated the power our surrogate model method by performing reaction path simulation studies for molecules including 2-butene, stilbene, and pentene, with as many as six degrees of freedom.

In Chapter 5 we use Smolyak's sparse grid interpolation algorithm to study the PESs of Fe(II)-polypyridines. These complexes are of great interest to the chemical community because of their potential application to solar cells [93, 51, 159, 27, 239]. The study of their PESs could lead to a better understanding of the photochemical processes that take place during the conversion of sunlight to electricity, and ultimately to the design of more efficient Fe-based complexes for solar energy applications. We used sparse grids to study the PESs of the Fe(II)-polypyridine $[Fe(tpy)_2]^{2+}$. Our surrogate model allowed us to not only visualize these surfaces in many different ways, but also compute entire intersection seams, locate minimum energy crossing points, and calculate minimum energy pathways. This work marks the first time the PESs of any Fe(II)-polypyridine has been studied so extensively with three degrees of freedom. Finally, we provided a theoretical framework for extending our reaction path following method from Chapter 4 to include nonadiabatic dynamics. To do so, we developed a surrogate model for Tully's surface hopping algorithm [211] that uses Smolyak's sparse grid interpolation algorithm to approximate PESs, nonadiabatic coupling vectors, and the spin-orbit coupling matrix. The work presented in this thesis will be continued to study different Fe(II)-polypyridines and implement our surrogate model for Tully's surface hopping algorithm. Once realized, our surrogate model could provide novel insights into the complicated photochemical processes involved in solar cells.

REFERENCES

[1] S. Achatz. Higher Order Sparse Grid Methods for Elliptic Partial Differential Equations with Variable Coefficients. *Computing*, 71(1):1–15, August 2003.

[2] Nitin Agarwal and N R Aluru. Weighted Smolyak algorithm for solution of stochastic differential equations on non-uniform probability measures. *International Journal for Numerical Methods in Engineering*, 85:1365–1389, 2010.

[3] Roland E. Allen, Traian Dumitrica, and Ben R. Torralva. Electronic and Structural Response of Materials to Fast, Intense Laser Pulses. In K. T. Tsen, editor, *Ultrafast Physical Processes in Semiconductors*, chapter 7, pages 315–388. Academic Press, San Diego, 2001.

[4] Donald G. Anderson. Iterative Procedures for Nonlinear Integral Equations. *J. Am. Chem. Soc.*, 12(4):547–560, October 1965.

[5] Joakim Andréasson, Uwe Pischel, Stephen D Straight, Thomas a Moore, Ana L Moore, and Devens Gust. All-photonic multifunctional molecular logic device. *J. Am. Chem. Soc.*, 133(30):11641–11648, August 2011.

[6] Philippe Y Ayala and H Bernhard Schlegel. A combined method for determining reaction paths, minima, and transition state geometries. *J. Chem. Phys.*, 107(2):375–384, 2001.

[7] George B. Bacskay. A quadratically convergent Hartree-Fock (QC-SCF) method. Application to closed shell systems. *Chem. Phys.*, 61:385–404, 1981.

[8] Jon Baker. An algorithm for the location of transition states. *J. Comput. Chem.*, 7(4):385–395, August 1986.

[9] Robert Balder and Christoph Zenger. The solution of Multidimensional Real Helmholtz Equations on Sparse Grids. *SIAM J. Sci. Comput.*, 17(3):631–646, 1996.

[10] Randolph E Bank, Todd F Dupont, and Harry Yserentant. The Hierarchical Basis Multigrid Method. *Numer. Math*, 52:427–458, 1987.

[11] Mario Barbatti, Giovanni Granucci, Maurizio Persico, Matthias Ruckenbauer, Mario Vazdar, Mirjana Eckert-Maksić, and Hans Lischka. The on-the-fly surface-hopping program system Newton-X: Application to ab initio simulation of the nonadiabatic photodynamics of benchmark systems. *J. Photochem. Photobio. A: Chem.*, 190(2-3):228–240, August 2007.

[12] Mario Barbatti, Matthias Ruckenbauer, Felix Plasser, Jiri Pittner, Giovanni Granucci, Maurizio Persico, and Hans Lischka. Newton-X: a surface-hopping program for nonadiabatic molecular dynamics. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 4(1):26–33, January 2014.

[13] Volker Barthelmann, Erich Novak, and Klaus Ritter. High dimensional polynomial interpolation on sparse grids. *Adv. in Comp. Math.*, 12:273–288, 2000.

[14] Riidiger Bauernschmitt and Reinhart Ahlrichs. Treatment of electronic excitations within the adiabatic approximation of time dependent density functional theory. *Chemical Physics Letters*, 256(4-5):454–464, 1996.

[15] Michael J Bearpark, Michael A Robb, and H Bernhard Schlegel. A direct method for the location of the lowest energy point on a potential surface crossing. *Chem. Phys. Lett.*, 223(June):269–274, 1994.

[16] A. D. Becke. Density-functional exchange-energy approximation with correct asymptotic behavior. *Phys. Rev. A*, 38(6):3098–3100, 1988.

[17] Stephen Bell and James S. Crighton. Locating transition states. *J. Chem. Phys.*, 80(6):2464–2475, 1984.

[18] Michal Ben-Nun, Jason Queenneville, and Todd J. Martinez. Ab Initio Multiple Spawning : Photochemistry from First Principles Quantum Molecular. *J. Phys. Chem.*, 104(22):5161–5175, 2000.

[19] T. Beringhelli, A. Gavezzotti, and M. Simonetta. Semi-empirical Calculations on the Thermal cis-trans Isomerization of 2-Butene, 2-Pentene, Beta-Methylstyrene and Stilbene. *J. Mol. Struct.*, 12:333–342, 1972.

[20] Adam B. Birkholz and H. Bernhard Schlegel. Coordinate reduction for exploring chemical reaction paths. *Theor. Chem. Acc.*, 131:1170, February 2012.

[21] J. B. Birks. The Photoisomerization of Stilbene. *Chem. Phys. Lett.*, 38(3):437–440, 1976.

[22] S. M. Blinder. Basic Concepts of Self-Consistent-Field Theory. *American Journal of Physics*, 33(6):431, 1965.

[23] Josep Maria Bofill. Updated Hessian Matrix and the Restricted Step Method for Locating Transition Structures. *J. Comput. Chem.*, 15(1):1–11, 1994.

[24] Josep Maria Bofill, Wolfgang Quapp, and Marc Caballero. Locating transition states on potential energy surfaces by the gentlest ascent dynamics. *Chem. Phys. Lett.*, 583:203–208, September 2013.

[25] M. Born and R. Oppenheimer. Zur Quantentheorie der Molekeln. *Annalen der Physik*, 389(20):457–484, 1927.

[26] Max Born and Kun Huang. *Dynamical Theory of Crystal Lattices*. Clarendon Press, Oxford, 1954.

[27] David N Bowman, James H Blew, Takashi Tsuchiya, and Elena Jakubikova. Elucidating band-selective sensitization in iron(II) polypyridine-TiO2 assemblies. *Inorganic Chemistry*, 52(15):8621–8, August 2013.

[28] David N Bowman and Elena Jakubikova. Low-spin versus high-spin ground state in pseudo-octahedral iron complexes. *Inorganic Chemistry*, 51(11):6011–9, June 2012.

[29] S. F. Boys. Electronic wave functions. I. A General Method of Calculation for the Stationary States of Any Molecular System. *Proc. of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 200(1063):542–554, 1950.

[30] G C Broyden. The convergence of a class of double rank minimization algorithms: 2. The new algorithm. *Journal of the Institute of Mathematics and Its Applications*, 6:76–231, 1970.

[31] A. L. Buchachenko. Modern Chemical Physics. Aims and Pathways to Progress. *Uspekhi Khimii*, 56(10):1593–1638, 1987.

[32] H.-J. Bungartz. Concepts for Higher Order Finite Elements on Sparse Grids. *International Conference on Spectral and High Order Methods*, pages 159–170, 1996.

[33] H.-J. Bungartz and S. Dirnstorfer. Multivariate Quadrature on Adaptive Sparse Grids. *Computing*, 71(1):89–114, August 2003.

[34] H.-J. Bungartz, Michael Griebel, and U Rude. Extrapolation, combination, and sparse grid techniques for elliptic boundary value problems. *Computer Methods in Applied Mechanics and Engineering*, 116:243–252, 1994.

[35] Hans-Joachim Bungartz. A Multigrid Algorithm for Higher Order Finite Elements on Sparse Grids. *Electronic Transactions on Numerical Analysis*, 6(December):63–77, 1997.

[36] Hans-Joachim Bungartz and Stefan Dirnstorfer. Higher Order Quadrature on Sparse Grids. In M. Bubak, editor, *ICCS 2004, LNCS 3039*, pages 394–401, Berlin Heidelberg, 2004. Springer-Verlag.

[37] Hans-Joachim Bungartz and Michael Griebel. Sparse grids. *Acta Numerica*, 13:1–123, June 2004.

[38] Hans-Joachim Bungartz, Alexander Heinecke, Dirk Pflüger, and Stefanie Schraufstetter. Option pricing with a direct adaptive sparse grid approach. *Journal of Computational and Applied Mathematics*, 236(15):3741–3750, September 2012.

[39] Gregery T. Buzzard. Global sensitivity analysis using sparse grid interpolation and polynomial chaos. *Reliability Engineering & System Safety*, 107:82–89, November 2012.

[40] S. E. Canton, X. Zhang, M. L. Lawson Daku, Y. Liu, J. Zhang, and S. Alvarez. Mapping the Ultrafast Changes of Continuous Shape Measures in Photoexcited Spin Crossover Complexes without Long-Range Order. *J. Phys. Chem. C*, page 150130145234001, 2015.

[41] Mark E Casida, Christine Jamorski, Kim C Casida, and Dennis R Salahub. Molecular excitation energies to high-lying bound states from time-dependent density-functional response theory: Characterization and correction of the time-dependent local density approximation ionization threshold. *Journal of Chemical Physics*, 108(11):4439–4449, 1998.

[42] Teepanis Chachiyo and Jorge H. Rodriguez. A direct method for locating minimum-energy crossing points (MECPs) in spin-forbidden transitions and nonadiabatic reactions. *J. Chem. Phys.*, 123:094711:1–9, 2005.

[43] Jun Chang, a. J. Fedro, and Michel van Veenendaal. Ultrafast cascading theory of intersystem crossings in transition-metal complexes. *Physical Review B*, 82(7):075124, August 2010.

[44] Hana Cho, Matthew L. Strader, Kiryong Hong, Lindsey Jamula, Eric M. Gullikson, Tae Kyu Kim, Frank M. F. de Groot, James K. McCusker, Robert W. Schoenlein, and Nils Huse. Ligand-field symmetry effects in Fe(ii) polypyridyl compounds probed by transient X-ray absorption spectroscopy. *Faraday Discussions*, 157:463, 2012.

[45] C W Clenshaw and A R Curtis. A method for numerical integration on an automatic computer. *Numerische Mathematik*, 2:197–205, 1960.

[46] Michael A. Collins. Molecular potential-energy surfaces for chemical reaction dynamics. *Theor. Chem. Acc.*, 108(6):313–324, December 2002.

[47] Michael A. Collins, Simon Petrie, Andrew J. Chalk, and Leo Radom. Proton-transport catalysis and proton-abstraction reactions: An ab initio dynamical study of X+HOC[sup +] and XH[sup +]+CO (X=Ne, Ar, and Kr). *J. Chem. Phys.*, 112(15):6625–6634, 2000.

[48] Edwin C Constable, Ana Hernandez Redondo, Catherine E Housecroft, Markus Neuburger, and Silvia Schaffner. Copper(I) complexes of 6,6'-disubstituted 2,2'-bipyridine dicarboxylic acids: new complexes for incorporation into copper-based dye sensitized solar cells (DSCs). *Dalton Transactions*, (33):6634–44, September 2009.

[49] R Courant. Variational Methods for the Solution of Problems of Equilibrium and Vibrations. *Bull. Amer. Math. Soc.*, 49:1–43, 1943.

[50] Thomas R. Cundari and Walter J. Stevens. Effective core potential methods for the lanthanides. *J. Chem. Phys.*, 98:5555–5565, 1993.

[51] Coen de Graaf and Carmen Sousa. Study of the light-induced spin crossover process of the [Fe(II)(bpy)3]2+ complex. *Chemistry (Weinheim an der Bergstrasse, Germany)*, 16(15):4550–6, April 2010.

[52] Coen de Graaf and Carmen Sousa. On the Role of the Metal-to-Ligand Charge Transfer States in the Light-Induced Spin Crossover in FeII(bpy)3. *International Journal of Quantum Chemistry*, 111:3385–3393, 2011.

[53] F.-J. Delvos. d-Variate Boolean Interpolation. *Journal of Approximation Theory*, 34:99–114, 1982.

[54] Gilles Deslauriers and Serge Dubuc. Symmetric Iterative Interpolation Processes. *Constructive Approximation*, 5:49–68, 1989.

[55] S. Dirnstorfer and Hans-Joachim Bungartz. Sparse Grids: Recent Developments for Elliptic Partial Differential Equations. In Wolfgang Hackbusch and Gabriel Wittum, editors, *Multigrid Methods V: Proceedings of the Fifth European Multigrid Conference held in Stuttgart, Germany, October 14, 1996*, pages 45–70. Springer-Verlag, New York, 1998.

[56] R. Ditchfield, W. J. Hehre, and J. A. Pople. Self-Consistent Molecular-Orbital Methods. IX. An Extended Gaussian-Type Basis for Molecular-Orbital Studies of Organic Molecules. *J. Chem. Phys.*, 54(2):724–728, 1971.

[57] Yusheng Dou and Roland E. Allen. Detailed Dynamics of a Complex Photochemical Reaction: CisTrans Photoisomerization of Stilbene. *J. Chem. Phys.*, 119(20):10658–10666, 2003.

[58] Yusheng Dou, Ben R. Torralva, and Roland E. Allen. Semiclassical electron-radiation-ion dynamics (SERID) and cis - trans photoisomerization of butadiene. *J. Mod. Opt.*, 50(15-17):2615–2643, January 2003.

[59] Karen Drukker. Basics of Surface Hopping in Mixed Quantum/Classical Simulations. *J. Comput. Phys.*, 153:225–272, 1999.

[60] Pingwu Du and Richard Eisenberg. Catalysts made of earth-abundant elements (Co, Ni, Fe) for water splitting: Recent progress and future challenges. *Energy & Environmental Science*, 5(3):6012, 2012.

[61] Darrell D. Ebbing and Steven D. Gammon. *General Chemistry*. Cengage Learning, Mason, Ohio, ninth edition, 2009.

[62] H. Ehlich and K. Zeller. Auswertung der Normen von Interpolationsoperatoren Ttibingen. *Math. Annalen*, 164:105–112, 1966.

[63] R. Elber and M. Karplus. A Method for Determining Reaction Paths in Large Molecules: Application to Myoglobin. *Chem. Phys. Lett.*, 139(5):375–380, 1987.

[64] Christian Evenhuis and Todd J Martínez. A scheme to interpolate potential energy surfaces and derivative coupling vectors without performing a global diabatization. *J. Chem. Phys.*, 135(22):224110, December 2011.

[65] Dg Fedorov and Ms Gordon. A study of the relative importance of one and two-electron contributions to spinorbit coupling. *The Journal of Chemical Physics*, 112(2000), 2000.

[66] Dmitri G. Fedorov, Shiro Koseki, Michael W. Schmidt, and Mark S. Gordon. Spin-orbit coupling in molecules: Chemistry beyond the adiabatic approximation, 2003.

[67] S. Ferrere. New photosensitizers based upon [Fe(L)2(CN)2] and [Fe(L)3] (L = substituted 2,2'-bipyridine): Yields for the photosensitization of TiO2 and effects on the band selectivity. *Chemistry of Materials*, 12:1083–1089, 2000.

[68] Suzanne Ferrere. New photosensitizers based upon [FeII(L)2(CN)2] and [FeIIL3], where L is substituted 2,2'-bipyridine. *Inorganica Chimica Acta*, 329:79–92, 2002.

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

[70] R. Fletcher. A new approach to variable metric algorithms. *The Computer Journal*, 13(3):317–322, 1970.

[71] Lisa a. Fredin, Mátyás Pápai, Emese Rozsályi, György Vankó, Kenneth Wärnmark, Villy Sundström, and Petter Persson. Exceptional Excited-State Lifetime of an Iron(II) N - Heterocyclic Carbene Complex Explained. *The Journal of Physical Chemistry Letters*, 5(12):2066–2071, June 2014.

[72] M J Frisch, G W Trucks, H B Schlegel, G E Scuseria, M A Robb, J R Cheeseman, G Scalmani, V Barone, B Mennucci, G A Petersson, H Nakatsuji, M Caricato, X Li, H P Hratchian, A F Izmaylov, J Bloino, G Zheng, J L Sonnenberg, M Hada, M Ehara, K Toyota, R Fukuda, J Hasegawa, M Ishida, T Nakajima, Y Honda, O Kitao, H Nakai, T Vreven, J A Montgomery Jr., J E Peralta, F Ogliaro, M Bearpark, J J Heyd, E Brothers, K N Kudin, V N Staroverov, R Kobayashi, J Normand, K Raghavachari, A Rendell, J C Burant, S S Iyengar, J Tomasi, M Cossi, N Rega, J M Millam, M Klene, J E Knox, J B Cross, V Bakken, C Adamo, J Jaramillo, R Gomperts, R E Stratmann, O Yazyev, A J Austin, R Cammi, C Pomelli, J W Ochterski, R L Martin, K Morokuma, V G Zakrzewski, G A Voth, P Salvador, J J Dannenberg, S Dapprich, A D Daniels, . Farkas, J B Foresman, J V Ortiz, J Cioslowski, and D J Fox. Gaussian 09, Revision A.1, 2009.

[73] Piotr Gajda. Smolyak's algorithm for weighted L1-approximation of multivariate functions with bounded rth mixed derivatives over d. *Numerical Algorithms*, 40(4):401–414, December 2005.

[74] Jochen Garcke and Michael Griebel. On the Computation of the Eigenproblems of Hydrogen and Helium in Strong Magnetic and Electric Fields with the Sparse Grid Combination Technique. *J. Comput. Phys.*, 165(2):694–716, December 2000.

[75] Jochen Garcke and Michael Griebel. Data mining with sparse grids using simplicial basis functions. *Proc. of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '01*, pages 87–96, 2001.

[76] T. Gerstner and Michael Griebel. Dimension-Adaptive Tensor-Product Quadrature. *Computing*, 71(1):65–87, August 2003.

[77] Thomas Gerstner and Michael Griebel. Numerical integration using sparse grids. *Numerical Algorithms*, 18:209–232, 1998.

[78] Joseph T Golab, Danny L Yeager, and Poul Jgrgensen. Proper Characterization of MC-SCF Stationary Points. *Int. J. Quantum Chem.*, 78:175–199, 1983.

[79] Donald Goldfarb. A Family of Variable-Metric Methods Derived by Variational Means. *Math. Comp.*, 24(109):23–26, 1970.

[80] William J. Gordon. Blending-Function Methods of Bivariate and Multivariate Interpolation and Approximation. *SIAM J. Numer. Anal.*, 8(1):158–177, 1971.

[81] V. Gradinaru. Fourier transform on sparse grids: Code design and the time dependent Schrödinger equation. *Computing*, 80(1):1–22, March 2007.

[82] V. Gradinaru and R. Hiptmair. Multigrid for Discrete Differential Forms on Sparse Grids. *Computing*, 71(1):471–495, August 2003.

[83] G. Granucci, M. Persico, and A. Toniolo. Direct semiclassical simulation of photochemical processes with semiempirical wave functions. *Journal of Chemical Physics*, 114:10608–10615, 2001.

[84] Giovanni Granucci, Maurizio Persico, and Gloria Spighi. Surface hopping trajectory simulations with spin-orbit and dynamical couplings. *Journal of Chemical Physics*, 137(2012), 2012.

[85] Michael Grätzel. Solar energy conversion by dye-sensitized photovoltaic cells. *Inorganic Chemistry*, 44(20):6841–51, October 2005.

[86] Michael Griebel. Adaptive sparse grid multilevel methods for elliptic PDEs based on finite differences. *Computing*, 61(2):151–179, June 1998.

[87] Michael Griebel and Helmut Harbrecht. On the Construction of Sparse Tensor Product Spaces. *Math. Comp.*, pages 1–20, 2012.

[88] Michael Griebel, P Oswald, and T Schiekofer. Sparse grids for boundary integral equations. *Numerische Mathematik*, 83:279–312, 1999.

[89] Michael Griebel, Michael Schneider, and Christoph Zenger. A combination technique for the solution of sparse grid problems. In R. Beauwens and P. de Groen, editors, *Iterative Methods in Linear Algebra*, pages 263–281. Elsevier & North-Holland, Amsterdam, 1992.

[90] Michael Griebel and Veronika Thurner. The efficient solution of fluid dynamics problems by the combination technique. *International Journal of Numerical Methods for Heat & Fluid Flow*, 5(3):251–269, 1995.

[91] Stefan Grimme. Semiempirical GGA-type density functional constructed with a long-range dispersion correction. *Journal of Computational Chemistry*, 27(15):1787–1799, 2006.

[92] Michael D Hack and Donald G Truhlar. Nonadiabatic Trajectories at an Exhibition. *The Journal of Physical Chemistry A*, 104(34):7917–7926, 2000.

[93] Anders Hagfeldt, Gerrit Boschloo, Licheng Sun, Lars Kloo, and Henrik Pettersson. Dye-sensitized solar cells. *Chem. Rev.*, 110(11):6595–663, November 2010.

[94] Rola H El Halabieh, Ozzy Mermut, and Christopher J Barrett. Using light to control physical properties of polymers and surfaces with azobenzene chromophores. *Pure Appl. Chem.*, 76(7-8):1445–1465, 2004.

[95] Sharon Hammes-Schiffer and John C. Tully. Proton transfer in solution: Molecular dynamics with quantum transitions. *The Journal of Chemical Physics*, 101:4657, 1994.

[96] Jeremy N Harvey. Understanding the kinetics of spin-forbidden chemical reactions. *Phys. Chem. Chem. Phys.*, 9(3):331–43, January 2007.

[97] Jeremy N. Harvey. Spin-forbidden reactions: computational insight into mechanisms and kinetics. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 4(1):1–14, January 2014.

[98] Jeremy N Harvey, Massimiliano Aschi, Helmut Schwarz, and Wolfram Koch. The singlet and triplet states of phenyl cation. A hybrid approach for locating minimum energy crossing points between non-interacting potential energy surfaces. *Theor. Chem. Acc.*, 99:95–99, 1998.

[99] Andreas Hauser, Peter Adler, Sonja Deisenroth, Philipp Gutlich, Christian Hennen, Hartmut Spiering, and Andreas Vef. Intersystem crossing in Fe (II) coordination compounds. *Hyperfine Interactions*, 90:77–87, 1994.

[100] Andreas Hauser, Cristian Enachescu, Max Lawson Daku, Alfredo Vargas, and Nahid Amstutz. Low-temperature lifetimes of metastable high-spin states in spin-crossover and in low-spin iron(II) compounds: The rule and exceptions to the rule. *Coordination Chemistry Reviews*, 250(13-14):1642–1652, July 2006.

[101] M Hegland. Adaptive Sparse Grids. *ANZIAM J.*, 44(E):C335–C353, 2003.

[102] Markus Hegland, Jochen Garcke, and Vivien Challis. The combination technique and some generalisations. *Linear Algebra and its Applications*, 420(2-3):249–275, January 2007.

[103] Markus Hegland, Andreas Hellander, and Per Lotstedt. Sparse grids and hybrid methods for the chemical master equation. *BIT Numerical Mathematics*, 48(2):265–283, July 2008.

[104] Dietmar Heidrich, editor. *The Reaction Path in Chemistry: Current Approaches and Perspectives*. Kluwer Academic Publishers, Dordrecht, Boston, London, 1995.

[105] H. Hellmann. *Einführung in die Quantenchemie*. Franz Deuticke, Leipzig, 1937.

[106] Michael F. Herman. Nonadiabatic semiclassical scattering. I. Analysis of generalized surface hopping procedures. *The Journal of Chemical Physics*, 81:754, 1984.

[107] P. Hohenberg and W. Kohn. Inhomogeneous Electron Gas. *Phys. Rev.*, 136(3B):864–871, 1964.

[108] Nils Huse, Hana Cho, Kiryong Hong, Lindsey Jamula, Frank M F De Groot, Tae Kyu Kim, James K Mccusker, and Robert W Schoenlein. Femtosecond Soft X-ray Spectroscopy of Solvated Transition-Metal Complexes: Deciphering the Interplay of Electronic and Structural Dynamics. *The Journal of Physical Chemistry Letters*, 2:880–884, 2011.

[109] Josef Ischtwan and Michael A. Collins. Molecular potential energy surfaces by interpolation. *J. Chem. Phys.*, 100(11):8080–8088, 1994.

[110] V. V. Ivanov and V. K. Dzjadyk. On asymptotics and estimates for the uniform norms of the Lagrange interpolation polynomials corresponding to the Chebyshev nodal points. *Analysis Mathematica*, 9(2):85–97, June 1983.

[111] J D Jakeman and S G Roberts. Local and Dimension Adaptive Sparse Grid Interpolation and Quadrature. Technical report, 2011.

[112] Lindsey L Jamula, Allison M Brown, Dong Guo, and James K McCusker. Synthesis and characterization of a high-symmetry ferrous polypyridyl complex: approaching the 5T2/3T1 crossing point for Fe(II.). *Inorganic Chemistry*, 53(1):15–7, January 2014.

[113] Ahren W Jasper, Shikha Nangia, Chaoyuan Zhu, and Donald G Truhlar. Non-Born-Oppenheimer molecular dynamics. *Accounts of chemical research*, 39(2):101–8, February 2006.

[114] Chenwei Jiang, Ruihua Xie, Fuli Li, and Roland E. Allen. Trans-to-cis isomerization of stilbene following an ultrafast laser pulse. *Chem. Phys. Lett.*, 474(4-6):263–267, June 2009.

[115] Sadegh Jokar, Volker Mehrmann, Marc E. Pfetsch, and Harry Yserentant. Sparse approximate solution of partial differential equations. *Applied Numerical Mathematics*, 60(4):452–472, April 2010.

[116] Hannes Jónsson, Greg Mills, and Karsten W. Jacobsen. Nudged elastic band method for finding minimum energy paths of transitions. In Bruce J. Berne, David F. Coker, and Giovanni Cicotti, editors, *Classical and Quantum Dynamics in Condensed Phase Simulations - Proceedings of the International School of Physics*, pages 385–404. World Scientific Publishing Company, Singapore, 1998.

[117] Eric a. Juban, Amanda L. Smeigh, Jeremy E. Monat, and James K. McCusker. Ultrafast dynamics of ligand-field excited states. *Coordination Chemistry Reviews*, 250(13-14):1783–1791, July 2006.

[118] Kenneth L. Judd, Lilia Maliar, Serguei Maliar, and Rafael Valero. Smolyak method for solving dynamic economic models: Lagrange interpolation, anisotropic grid and adaptive domain. *Journal of Economic Dynamics and Control*, 44:92–123, July 2014.

[119] K. Kim and K. D. Jordan. Comparison of Density Functional and MP2 Calculations on the Water Monomer and Dimer. *J. Phys. Chem.*, 98(40):10089–10094, October 1994.

[120] Andreas Klimke. Efficient Construction of Hierarchical Polynomial Sparse Grid Interpolants using the Fast Discrete Cosine Transform. Technical report, Berichte aus dem Institut fur Angewandte Analysis und Numerische Simulation, 2006.

[121] Andreas Klimke. Sparse Grid Interpolation Toolbox User's Guide. Technical report, Berichte aus dem Institut fur Angewandte Analysis und Numerische Simulation, 2008.

[122] Andreas Klimke and Barbara Wohlmuth. Algorithm 847: spinterp : Piecewise Multilinear Hierarchical Sparse Grid Interpolation in MATLAB. *ACM Transactions on Mathematical Software*, 31(4):561–579, 2005.

[123] Andreas Klimke and Barbara Wohlmuth. Computing expensive multivariate functions of fuzzy numbers using sparse grids. *Fuzzy Sets and Systems*, 154(3):432–453, September 2005.

[124] W Andreas Klimke. *Uncertainty Modeling using Fuzzy Arithmetic and Sparse Grids*. PhD thesis, Universitat Stuttgart, 2006.

[125] Wolfram Koch and Max C Holthausen. *A Chemist's Guide to Density Functional Theory*. Wiley-VCH, Weinheim, second edition, 2001.

[126] W. Kohn and L. J. Sham. Self-Consistent Equations Including Exchange and Correlation Effects. *Phys. Rev.*, 140(4A):A1133–A1138, 1965.

[127] Dirk Krueger and Felix Kubler. Computing equilibrium in OLG models with stochastic production. *Journal of Economic Dynamics and Control*, 28(7):1411–1436, April 2004.

[128] Konstantin N. Kudin, Gustavo E. Scuseria, and Eric Cancès. A black-box self-consistent field convergence algorithm: One step closer. *J. Chem. Phys.*, 116(2002):8255–8261, 2002.

[129] Boris Lastdrager, Barry Koren, and Jan Verwer. The sparse-grid combination technique applied to time-dependent advection problems. *Applied Numerical Mathematics*, 38(4):377–401, September 2001.

[130] Chengteh Lee, Weitao Yang, and Robert G. Parr. Development of the Colle-Salvetti correlation-energy formula into a functional of the electron density. *Phys. Rev. B*, 37(2):785–789, 1988.

[131] V. A. Legasov and A. L. Buchachenko. Problems in Modern Chemistry. *Uspekhi Khimii*, 55:1949–1978, 1986.

[132] Ira N. Levine. *Quantum Chemistry*. Pearson Prentice Hall, Upper Saddle River, NJ, 6th edition, 2009.

[133] Jinglai Li and Youssef M Marzouk. Adaptive Construction of Surrogates for the Bayesian Solution of Inverse Problems. *SIAM J. Sci. Comput.*, 36(3):1163–1186, 2014.

[134] Xiaoqing Lu, Shuxian Wei, Chi-man Lawrence Wu, Shaoren Li, and Wenyue Guo. Can Polypyridyl Cu(I)based Complexes Provide Promising Sensitizers for Dye-Sensitized Solar Cells? A Theoretical Insight into Cu(I) versus Ru(II) Sensitizers. *J. Phys. Chem. C*, 115:3753–3761, 2011.

[135] X.-L. Luo, C. T. Kelley, L.-Z. Liao, and H. W. Tam. Combining Trust-Region Techniques and Rosenbrock Methods to Compute Stationary Points. *J. Optim. Theory Appl.*, 140:265–286, 2009.

[136] Xiang Ma and Nicholas Zabaras. An adaptive hierarchical sparse grid collocation algorithm for the solution of stochastic differential equations. *J. Comput. Phys.*, 228(8):3084–3113, May 2009.

[137] Xiang Ma and Nicholas Zabaras. An efficient Bayesian inference approach to inverse problems based on an adaptive sparse grid collocation method. *Inverse Problems*, 25(3):035013, March 2009.

[138] Xiang Ma and Nicholas Zabaras. An adaptive high-dimensional stochastic model representation technique for the solution of stochastic partial differential equations. *J. Comput. Phys.*, 229(10):3884–3915, May 2010.

[139] Benjamin A. Malin, Dirk Krueger, and Felix Kubler. Solving the multi-country real business cycle model using a Smolyak-collocation method. *Journal of Economic Dynamics and Control*, 35(2):229–239, February 2011.

[140] MATLAB. *Version 8.3.0 (R2014a)*. The MathWorks Inc., Natick, MA, 2014.

[141] Hans-Dieter Meyer and William H Miller. A classical analog for electronic degrees of freedom in nonadiabatic collision processes. *The Journal of Chemical Physics*, 70(7):3214, 1979.

[142] R M Minyaev. Molecular Structure and Global Description of the Potential Energy Surface. 32(4):559–589, 1992.

[143] R M Minyaev. Reaction Path as a Gradient Line on a Potential Energy Surface. *Int. J. Quantum Chem.*, 49:105–127, 1994.

[144] D. S. Mokrauer and C. T. Kelley. Sparse Interpolatory Reduced-Order Models for Simulation of Light-Induced Molecular Transformations. *Optim. Methods Softw.*, 29(2):264–273, 2014.

[145] D. S. Mokrauer, C. T. Kelley, and A. Bykhovski. Efficient Parallel Computation of Molecular Potential Energy Surfaces for the Study of Light-Induced Transition Dynamics in Multiple Coordinates. *IEEE Trans. Nanotechnol.*, 10(1):70–74, January 2011.

[146] D. S. Mokrauer, C. T. Kelley, and A. Bykhovski. Simulations of Light-Induced Molecular Transformations in Multiple Dimensions with Incremental Sparse Surrogates. *J. Algorithms Comp. Tech.*, 6(4):577–592, December 2012.

[147] Jeremy E. Monat and James K. McCusker. Femtosecond Excited-State Dynamics of an Iron(II) Polypyridyl Solar Cell Sensitizer Model. *Journal of the American Chemical Society*, 122(17):4092–4097, May 2000.

[148] Oliver Morton. Solar Energy: A New Day Dawning? *Nature*, 443(September):19–22, 2006.

[149] Alin Murarasu, Gerrit Buse, Dirk Pfluger, Josef Weidendorfer, Bode Arndt, Alin Murarau, Dirk Püger, and Arndt Bode. fastsg : A Fast Routines Library for Sparse Grids. *Proc. Comp. Sci.*, 9:354–363, January 2012.

[150] Alin Murarasu, Dirk Pflüger, Josef Weidendorfer, Gerrit Buse, and Daniel Butnaru. Compact Data Structure and Scalable Algorithms for the Sparse Grid Technique. In *PPoPP*, pages 1–10. ACM, 2011.

[151] James Nance, Elena Jakubikova, and C. T. Kelley. Reaction Path Following with Sparse Interpolation. *Journal of Chemical Theory and Computation*, 10(8):2942–2949, August 2014.

[152] James Nance and C.T. Kelley. A sparse interpolation algorithm for dynamical simulations in computational chemistry. accepted to SIAM J. Sci. Comp.

[153] E. E. Nikitin. Nonadiabatic Transitions: What We Learned from Old Masters and How Much We Owe Them. *Annu. Rev. Phys. Chem.*, 50:1–21, 1999.

[154] F Nobile, R Tempone, and C G Webster. A Sparse Grid Stochastic Collocation Method for Partial Differential Equations with Random Input Data. *SIAM J. Numer. Anal.*, 46(5):2309–2345, 2008.

[155] Erich Novak and Klaus Ritter. High dimensional integration of smooth functions over cubes. *Numerische Mathematik*, 75:79–97, 1996.

[156] Erich Novak and Klaus Ritter. Simple Cubature Formulas with High Polynomial Exactness. *Constructive Approximation*, 15:499–522, 1999.

[157] Belen Ordejon, Coen de Graaf, and Carmen Sousa. Light-Induced Excited-State Spin Trapping in Tetrazole-Based Spin Crossover Systems. *J. Am. Chem. Soc.*, 130:13961–13968, 2008.

[158] Ian J Palmer, Ioannis N Ragazos, Fernando Bernardi, Massimo Olivucci, and Michael A Robb. An MC-SCF Study of the S1 and S2 Photochemical Reactions of Benzene. *J. Am. Chem. Soc.*, 115(2):673–682, 1993.

[159] Matyas Papai, Gyorgy Vanko, Coen De Graaf, and Tamas Rozgonyi. Theoretical Investigation of the Electronic Structure of Fe(II) Complexes at Spin-State Transitions. *J. Chem. Theory Comput.*, 9:509–519, 2012.

[160] Robert G. Parr and Weitao Yang. *Density-Functional Theory of Atoms and Molecules*. Oxford University Press, New York, 1989.

[161] M. Patrício, J. L. Santos, F. Patrício, and a. J. C. Varandas. Roadmap to spline-fitting potentials in high dimensions. *J. Math. Chem.*, 51(7):1729–1746, April 2013.

[162] Wolfgang Pauli. Nobel Lecture: Exclusion principle and quantum mechanics, 1946.

[163] Chunyang Peng, Philippe Y Ayala, H. Bernhard Schlegel, and Michael J Frisch. Using Redundant Internal Coordinates to Optimize Equilibrium Geometries and Transition States. *Journal of Computational*, 17(1):49–56, 1996.

[164] Knut Petras. Smolyak cubature of given polynomial degree with few nodes for increasing dimension. *Numerische Mathematik*, 93:729–753, 2003.

[165] Dirk Pflüger. *Spatially Adaptive Sparse Grids for High-Dimensional Problems*. PhD thesis, Technische Universität München, 2010.

[166] Dirk Pflüger, Benjamin Peherstorfer, and Hans-Joachim Bungartz. Spatially Adaptive Sparse Grids for High-Dimensional Data-Driven Problems. *J. of Complexity*, 26(5):508–522, October 2010.

[167] Jiri Pittner, Hans Lischka, and Mario Barbatti. Optimization of mixed quantum-classical dynamics: Time-derivative coupling terms and selected couplings. *Chemical Physics*, 356(1-3):147–152, February 2009.

[168] Felix Plasser, Giovanni Granucci, Jiri Pittner, Mario Barbatti, Maurizio Persico, and Hans Lischka. Surface hopping dynamics using a locally diabatic formalism: Charge transfer in the ethylene dimer cation and excited state dynamics in the 2-pyridone dimer. *Journal of Chemical Physics*, 137, 2012.

[169] P. M. Prenter. *Splines and Variational Methods*, volume 56. Wiley, New York, 1975.

[170] O V Prezhdo and P J Rossky. Mean-field molecular dynamics with surface hopping. *J. Chem. Phys.*, 107(February):825, 1997.

[171] Peter Pulay. Convergence Acceleration of Iterative Sequences. The Case of SCF Iteration. *Chem. Phys. Lett.*, 73(2):393–398, 1980.

[172] Peter Pulay and Geza Fogarasi. Geometry optimization in redundant internal coordinates. *J. Chem. Phys.*, 96(4):2856–2860, 1992.

[173] A. Quarteroni, R. Sacco, and F. Saleri. *Numerical Mathematics*. Springer, New York, 2nd edition, 2007.

[174] Jason Quenneville and Todd J. Martinez. Ab Initio Study of Cis-Trans Photoisomerization in Stilbene and Ethylene. *J. Phys. Chem.*, 107:829–837, 2003.

[175] Franz Renz, Hiroki Oshio, Vadim Ksenofontov, Markus Waldeck, Hartmut Spiering, and Philipp Gtlich. Strong field iron(II) complex converted by light into a long-lived high-spin state. *Angewandte Chemie - International Edition*, 39(20):3699–3700, 2000.

[176] Damien Riedel, Marion Cranney, Marta Martin, and Romain Guillory. Surface-Isomerization Dynamics of trans -Stilbene Molecules Adsorbed on Si(100)-21. *J. Am. Chem. Soc.*, 131:5414–5423, 2009.

[177] C. C. J. Roothaan. New Developments in Molecular Orbital Theory. *Reviews of Modern Physics*, 23(2):69–89, 1951.

[178] Erich Runge and E. K. U. Gross. Densitfy-Funcational Theory for Time-Dependent Systems. *Phys. Rev. Lett.*, 52(12):997–1000, 1984.

[179] Jack Saltiel, Srinivasan Ganapathy, and Constance Werking. The DeltaH for Thermal trans-Stilbene/cis-Stilbene Isomerlzatlon. Do S0 and T1 Potential Energy Curves Cross? *J. Phys. Chem.*, 1991(11):2755–2758, 1987.

[180] Sethuraman Sankaran, Charles Audet, and Alison L. Marsden. A method for stochastic constrained optimization using derivative-free surrogate pattern search and collocation. *J. Comput. Phys.*, 229(12):4664–4682, June 2010.

[181] N. Sathyamurthy. Quasiclassical trajectory studies using 3D spline interpolation of ab initio surfaces. *J. Chem. Phys.*, 63(1):464–473, 1975.

[182] H. Bernhard Schlegel. Optimization of Equilibrium Geometries and Transition Structures. *J. Comput. Chem.*, 3(2):214–218, 1982.

[183] H Bernhard Schlegel. Some Thoughts on Reaction Path Following. *J. Chem. Soc. Faraday Trans.*, 90(12):1569–1574, 1994.

[184] H. Bernhard Schlegel and J. J. W. McDouall. Do You Have SCF Stability and Convergence Problems? In C. Ogretir and I. G. Csizmadia, editors, *Computational Advances in Organic Chemistry: Molecular Structure and Reactivity*, pages 167–185. Kluwer Academic Publishers, The Netherlands, 1991.

[185] Ch. Schwab and R.a. A. Todor. Sparse Finite Elements for Stochastic Elliptic Problems - Higher Order Moments. *Computing*, 71(1):43–63, August 2003.

[186] Roseanne J. Sension, Stephen T. Repinec, Arpad Z. Szarka, and Robin M. Hochstrasser. Femtosecond laser studies of the cis-stilbene photoisomerization reactions. *J. Chem. Phys.*, 98(8):6291–6315, 1993.

[187] Lawrence F. Shampine and Mark W. Reichelt. The MATLAB ODE Suite. *SIAM J. Sci. Comput.*, 18:1–22, 1997.

[188] D. F. Shanno. Conditioning of Quasi-Newton Methods for Function Minimization. *Math. Comp.*, 24(111):647–656, 1970.

[189] Junfeng Shao, Yibo Lei, Zhenyi Wen, Yusheng Dou, and Zhisong Wang. Nonadiabatic simulation study of photoisomerization of azobenzene: detailed mechanism and load-resisting capacity. *J. Chem. Phys.*, 129(16):164111, October 2008.

[190] Daniel Sheppard, Rye Terrell, and Graeme Henkelman. Optimization methods for finding minimum energy paths. *J. Chem. Phys.*, 128(134106):1–10, April 2008.

[191] David S. Sholl and John C. Tully. A generalized surface hopping method. *J. Chem. Phys.*, 109(18):7702–7710, 1998.

[192] Winfried Sickel and Tino Ullrich. Smolyak's Algorithm, Sampling on Sparse Grids and Function Spaces of Dominating Mixed Smoothness. *East J. Approx.*, 13:387–425, 2007.

[193] Anton Simeonov, Masayuki Matsushita, Eric A Juban, Elizabeth H Z Thompson, Timothy Z Hoffman, Albert E Beuscher Iv, Matthew J Taylor, Peter Wirsching, James K Mccusker, Raymond C Stevens, David P Millar, Peter G Schultz, A Richard, Kim D Janda, Wolfgang Rettig, and Richard A Lerner. Blue-Fluorescent Antibodies. *Science*, 290(5490):307–313, October 2000.

[194] Jack Simons, Poul Jorgensen, Hugh Taylor, and Judy Ozment. Walking on Potential Energy Surfaces. *J. Phys. Chem.*, 87(15):2745–2753, 1983.

[195] J. C. Slater. The Theory of Complex Spectra. *Phys. Rev.*, 34(10):1293–1322, 1929.

[196] J. C. Slater. Atomic Shielding Constants. *Phys. Rev.*, 36:57–64, 1930.

[197] Amanda L Smeigh, Mark Creelman, Richard a Mathies, and James K McCusker. Femtosecond time-resolved optical and Raman spectroscopy of photoinduced spin crossover: temporal resolution of low-to-high spin optical switching. *Journal of the American Chemical Society*, 130(43):14105–7, October 2008.

[198] S Smolyak. Quadrature and interpolation formulas for tensor products of certain classes of functions. *Soviet Math. Dokl.*, 4:240–243, 1963.

[199] Frauke Sprengel. Interpolation of Functions from Besov-type Spaces on Gauss-Chebyshev Grids, 2000.

[200] P. J. Stephens, F. J. Devlin, C. F. Chabalowski, and M. J. Frisch. Ab Initio Calculation of Vibrational Absorption and Circular Dichroism Spectra Using Densitfy Functional Force Fields. *J. Phys. Chem.*, 98(45):11623–11627, 1994.

[201] Walter J Stevens, Harold Basch, and Morris Krauss. Compact effective potentials and efficient sharedexponent basis sets for the first and secondrow atoms. *J. Chem. Phys.*, 81:6026–6033, 1984.

[202] Walter J. Stevens, Morris Krauss, Harold Basch, and Paul G. Jasien. Relativistic compact effective potentials and efficient, shared-exponent basis sets for the third-, fourth-, and fifth-row atoms. *Can. J. Chem.*, 70:612–630, 1992.

[203] T. Stortkuhl. *Ein numerisches adaptives Verfahren zur Losung der biharmonischen Gleichung auf dunnen Gittern.* PhD thesis, TU Munchen, 1995.

[204] R. Eric Stratmann, Gustavo E. Scuseria, and Michael J. Frisch. An efficient implementation of time-dependent density-functional theory for the calculation of excitation energies of large molecules. *The Journal of Chemical Physics*, 109(19):8218–8224, 1998.

[205] William C. Swope. A computer simulation method for the calculation of equilibrium constants for the formation of physical clusters of molecules: Application to small water clusters. *The Journal of Chemical Physics*, 76(1):637, 1982.

[206] Attila Szabo and Neil S Ostlund. *Modern Quantum Chemistry: Introduction to Advanced Electronic Structure Theory.* McGraw-Hill, New York, 1st edition, 1989.

[207] Ivano Tavernelli, Basile F.E. Curchod, and Ursula Rothlisberger. Nonadiabatic molecular dynamics with solvent effects: A LR-TDDFT QM/MM study of ruthenium (II) tris (bipyridine) in water. *Chem. Phys.*, 391:101–109, 2011.

[208] Ivano Tavernelli, Enrico Tapavicza, and Ursula Rothlisberger. Non-adiabatic dynamics using time-dependent density functional theory: Assessing the coupling strengths. *Journal of Molecular Structure: THEOCHEM*, 914(1-3):22–29, 2009.

[209] Joseph a. Treadway, John a. Moss, and Thomas J. Meyer. Visible Region Photooxidation on TiO(2) with a Chromophore-Catalyst Molecular Assembly. *Inorganic Chemistry*, 38(20):4386–4387, October 1999.

[210] John C. Tully. Molecular dynamics with electronic transitions. *J. Chem. Phys.*, 93(2):1061–1071, 1990.

[211] John C. Tully. Nonadiabatic molecular dynamics. *Int. J. Quantum Chem.*, 40(S25):299–309, 1991.

[212] John C Tully. Mixed quantum-classical dynamics. *Faraday Discuss.*, 110:407–419, 1998.

[213] John C. Tully. Nonadiabatic Dynamics. In Donald L. Thompson, editor, *Modern Methods for Multidimensional Dynamics Computations in Chemistry*, chapter Nonadiabat, pages 34–72. World Scientific Publishing Company, Singapore, 1998.

[214] John C Tully. Perspective: Nonadiabatic dynamics theory. *J. Chem. Phys.*, 137(22):22A301, December 2012.

[215] John C. Tully and Richard K. Preston. Trajectory Surface Hopping Approach to Nonadiabatic Molecular Collisions: The Reaction of H+ with D2. *J. Chem. Phys.*, 55(2):562–572, 1971.

[216] Valentin D Vachev, John H Frederick, Boris A Grishaninj, Victor N Zadkov, and Nikolai I Koroteev. Quasiclassical Molecular Dynamics Simulation of the Photoisomerization of Stilbene. *J. Phys. Chem.*, 99:5247–5263, 1995.

[217] Michel van Veenendaal, Jun Chang, and a. J. Fedro. Model of Ultrafast Intersystem Crossing in Photoexcited Transition-Metal Organic Compounds. *Physical Review Letters*, 104(6):067401, February 2010.

[218] Loup Verlet. Computer "Experiments" on Classical Fluids. I. Thermodynamical Properties of Lennard-Jones Molecules. *Phys. Rev.*, 159(1):98–103, 1967.

[219] D. H. Waldeck. Photoisomerization Dynamics of Stilbenes. *Chem. Rev.*, 91:415–436, 1991.

[220] David J. Wales. *Energy Landscapes*. Cambridge University Press, Cambridge, UK; New York, 2003.

[221] Fan Wang and Tom Ziegler. A simplified relativistic time-dependent density-functional theory formalism for the calculations of excitation energies including spin-orbit coupling effect. *Journal of Chemical Physics*, 123(2005), 2005.

[222] Grzegorz W. Wasilkowski and Henryk Wozniakowski. Explicit Cost Bounds of Algorithms for Multivariate Tensor Product Problems. *J. of Complexity*, 11:1–56, 1995.

[223] E Weinan, Ren Weiqing, and Eric Vanden-Eijnden. Finite temperature string method for the study of rare events. *J. Phys. Chem. B*, 109(14):6688–6693, April 2005.

[224] E. Bright Wilson, Jr, J.C. Decius, and Paul C. Cross. *Molecular Vibrations: The Theory of Infrared and Raman Vibrational Spectra*. McGraw-Hill, New York, 1955.

[225] Curt Wittig. The Landau-Zener Formula. *J. Phys. Chem. B*, 109(17):8428–8430, May 2005.

[226] Dwight L. Woolard, Elliott R Brown, Michael Pepper, Michael Kemp, and R. Brown. Terahertz Frequency Sensing and Imaging: A Time of Reckoning Future Applications? *Proc. IEEE*, 93(10):1722–1743, October 2005.

[227] Dongbin Xiu. Efficient Collocational Approach for Parametric Uncertainty Analysis. *Communications in Computational Physics*, 2(2):293–309, 2007.

[228] Dongbin Xiu and Jan S. Hesthaven. High-Order Collocation Methods for Differential Equations with Random Inputs. *SIAM J. Sci. Comput.*, 27(3):1118–1139, 2005.

[229] David R. Yarkony. Theoretical studies of spin-forbidden radiationless decay in polyatomic systems: insights from recently developed computational methods. *Journal of the American Chemical Society*, 114(13):5406–5411, June 1992.

[230] David R Yarkony. Systematic Determination of Intersections of Potential Energy Surfaces Using a Lagrange Multiplier Constrained Procedure. *J. Phys. Chem.*, 97:4407–4412, 1993.

[231] David C Young. *Computational Chemistry: A Practical Guide for Applying Techniques to Real-World Problems*. John Wiley & Sons, Inc., New York, 2001.

[232] H. Yserentant. On the Multi-level Splitting of Finite Element Spaces for Indefinite Elliptic Boundary Value Problems. *SIAM J. Numer. Anal.*, 23(3):581–595, 1986.

[233] H. Yserentant. Sparse Grids, Adaptivity, and Symmetry. *Computing*, 78(3):195–209, November 2006.

[234] Ryan R. Zaari and Sergey a. Varganov. Nonadiabatic Transition State Theory and Trajectory Surface Hopping Dynamics: Intersystem Crossing Between 3B1 and 1A1 States of SiH2. *The Journal of Physical Chemistry A*, page 150213093044006, 2015.

[235] C. Zener. Non-Adiabatic Crossing of Energy Levels. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 137(833):696–702, September 1932.

[236] Lingzao Zeng, Liangsheng Shi, Dongxiao Zhang, and Laosheng Wu. A sparse grid based Bayesian method for contaminant source identification. *Advances in Water Resources*, 37:1–9, March 2012.

[237] Christoph Zenger. Sparse Grids. *Notes on Numerical Fluid Mechanics*, 31:241–251, 1991.

[238] Guannan Zhang, Dan Lu, Ming Ye, Max Gunzburger, and Clayton Webster. An efficient surrogate modeling approach in Bayesian uncertainty analysis. In *11th International Conference of Numerical Analysis and Applied Mathematics*, volume 1558, pages 898–901. AIP Publishing LLC, 2013.

[239] Wenkai Zhang, Roberto Alonso-Mori, Uwe Bergmann, Christian Bressler, Matthieu Chollet, Andreas Galler, Wojciech Gawelda, Ryan G Hadt, Robert W Hartsock, Thomas Kroll, Kasper S Kjæ r, Katharina Kubiček, Henrik T Lemke, Huiyang W Liang, Drew a Meyer, Martin M Nielsen, Carola Purser, Joseph S Robinson, Edward I Solomon, Zheng Sun, Dimosthenis Sokaras, Tim B van Driel, György Vankó, Tsu-Chien Weng, Diling Zhu, and

Kelly J Gaffney. Tracking excited-state charge and spin dynamics in iron coordination complexes. *Nature*, 509(7500):345–8, May 2014.

[240] X. Zhang, M. L. Lawson Daku, J. Zhang, K. Suarez-Alcantara, G. Jennings, C. a. Kurtz, and S. E. Canton. Dynamic JahnTeller Effect in the Metastable High-Spin State of Solvated [Fe(terpy)2]2+. *J. Phys. Chem. C*, 119:3312–3321, 2015.

APPENDICES

Appendix A

Appendix-A

A.1 Smolyak Interpolation in MATLAB

In this section we will describe our implementation of Smolyak's algorithm in MATLAB and provide some user documentation for the codes.

A.1.1 m-files

Your MATLAB path should contain the following m-files:

- `smolyak_step1_T.m`: Evaluates user's function at Smolyak grid points
- `smolyak_approx_T.m`: Evaluates Smolyak interpolation at user-supplied points

A.1.2 Implementation

The user will only have to call `smolyak_step1_T.m` and `smolyak_approx_T.m`. For `smolyak_step1_T.m`, the syntax is as follows:

```
>> S = smolyak_step1_T(d, k, bounds, fun);
```

The inputs are:

- `d`: The dimension of user's function
- `k`: The degree of polynomial exactness for the interpolation
- `bounds`: The boundary of user's domain. Each dimension's bounds are a row of this matrix

- `fun`: The name of user's function

The outputs are:

- `S`: A data structure of information to be used in smolyak_approx_T.m.

After this function runs successfully, the user must call `smolyak_approx_T.m`. The syntax is:

```
>> [fhat, ghat] = smolyak_approx_T(t, S);
```

The inputs are:

- `t`: $N \times d$ matrix of points at which to evaluate Smolyak's interpolant
- `S`: Data structure from smolyak_step1_T.m

The output are:

- `fhat`: $N \times 1$ vector of values of Smolyak interpolating function at `t`
- `ghat`: $N \times d$ matrix of values of the gradient of Smolyak interpolating function at `t`

Example

We will interpolate the function

$$f(x, y, z) = xy + e^y + z$$

on $[-1, ,\ 1]^3$ with $k = 5$. The user must write the following function:

```
function [y, g]=test_function(x)
  y=x(:,1).*x(:,2) + exp(x(:,2));
  y=y+x(:,3);
  g=ones(size(x));
  g(:,1)=x(:,2);
  g(:,2)=x(:,1)+exp(x(:,2));
end
```

We will approximate this function at $t_1 = (-0.1, -0.2, 0.1)$ and $t_2 = (0.9, 0.25, -.7)$. This can be done as follows:

```
>> d=3; k=5; bounds=[-1 1; -1 1; -1 1];
>> S = smolyak_step1_T(d,k,bounds,@test_function);
>> t=[-.1 -.2 .1; .9 .25 -.7];
>> [fhat, ghat] = smolyak_approx_T(k,all_f,all_points,nodesMaxK,nnodes,t);
>> fhat

fhat =

   0.9387
   0.8090

>> ghat

ghat =

   -0.2000    0.7187    1.
    0.2500    2.1840    1.

>> [f, g]= test_function(t)
>> f

f =

   0.9387
   0.8090

>> g

g =

   -0.2000    0.7187    1.
    0.2500    2.1840    1.
```

A.1.3 Function Details

`S` Structure Data Fields

The output of `smolyak_step1` is a structure containing information that is be used in `smolyak_approx` and other subfunctions of `smolyak_step1`. The data fields of `S` are listed in Table A.1.

Table A.1: Data fields of `S`. See Section 2.3.1 for notational details.

Field Name	Description
`A_indices`	Cell whose i^{th} entry contains the indices for interpolation level i so that `nodesMaxLevel(d,Al_indices{i})` returns a vector of all the nodes in $A_i = \chi_i \setminus \chi_{i-1}$ in dimension d.
`A_nnodes`	Vector whose i^{th} entry is the number of unidimensional nodes in the set A_i.
`A_nodes`	Cell whose i^{th} entry contains the unidimensional nodes in A_i on $[-1,,\ 1]$.
`Q`	Matrix containing multi-indices for Smolyak's algorithm.
`b`	Coefficient vector for Smolyak's algorithm (solution to Equation 2.47).
`bounds`	Interpolation domain, given by `bounds = [a1, b1; ... ad, bd]`.
`d`	Dimension.
`ell_list`	Matrix whose rows contain each basis number combination `j` in the order they appear in Smolyak's algorithm in Equation 2.48.
`k`	Degree of exactness.
`level_indices`	Cell whose i^{th} entry contains the indices for interpolation level i so that `nodesMaxLevel(d,level_indices{i})` returns a vector of all level i nodes in dimension d.
`maxLevel`	Highest level of sparse grid.
`nodesMaxLevel`	All nodes in each dimension for highest interpolation level.
`numNodes`	Vector whose i^{th} entry is the number of unidimensional nodes in χ_i.
`q`	$d + k$.
`unidimNodes`	Cell whose i^{th} entry contains the unidimensional nodes in χ_i on $[-1,,\ 1]$.

Tensor Product Calculation in `smolyak_approx_T.m`

By far the most expensive part of Smolyak's algorithm is the calculation of tensor product basis functions in Equation 2.46. Here we take advantage of the MATLAB function `bsxfun` to compute these products.

Consider evaluating Smolyak's algorithm at a $N \times d$ matrix of points `t`

$$\hat{f} = \mathcal{A}(q,d)[f](t).$$

Before computing any tensor products we first construct `A_bases`, a $(k+1) \times d$ cell whose (i^{th}, j^{th}) entry contains a $N \times M$ all of the M disjoint basis functions for dimension j and level i. For example, if we let ϕ be the first basis function for level $i = 3$ in dimension $j = 2$, `A_bases{3,2}(:,1)` is the vector of ϕ evaluated at `t(:,2)`. If gradients are requested we construct the analogous cell `A_bases_deriv` for the derivatives of the basis functions.

After we evaluate the unidimensional basis functions we loop through the set of allowable multi indices $Q(q,d)$. Since the first element of $Q(q,d)$ is $\mathbf{i} = \mathbf{1}$, the first term of Smolyak's algorithm is simply the constant term b_1 computed from Equation 2.47. As such, we initialize the approximation with $\hat{f} = b_1$ and start our loop through $Q(q,d)$ with the second multi index. For each $\mathbf{i} \in Q(q,d)$ we initialize

```
L=A_bases{i(1),1};
```

and loop through the remaining dimensions to compute

```
A=reshape(L,ts,1,[]);
B=bsxfun(@times,A,A_bases{i(di),di});
L=reshape(B,ts,[]);.
```

The `reshape`/`bsxfun`/`reshape` sequence element-wise multiplies all possible combinations of columns of `L` and columns of `A_bases{i(di),di}`. For example, consider $\mathbf{i} = (2,2)$ and let ϕ_i^j be the i^{th} basis function in dimension j. Interpolation level $i = 2$ corresponds to the two disjoint basis functions ϕ_2 and ϕ_3. Initializing `L` in the first dimension with

$$L = [\phi_2^1, \ \ \phi_3^1],$$

the `reshape`/`bsxfun`/`reshape` step process yields

$$L = [\phi_2^1\phi_2^2, \ \ \phi_2^1\phi_3^2, \ \ \phi_3^1\phi_2^2, \ \ \phi_3^1\phi_3^2].$$

In this way we compute all tensor products of basis function functions for the multi index $\mathbf{i}$. We then add the current piece of the linear combination to our approximation of $\hat{f}$ via

```
fhat =  fhat + L*b_i
```

where `b_i`$= b_{\mathbf{i}}$ is the slice of the coefficient vector b that corresponds to the multi index $\mathbf{i}$. The Vandermonde matrix in Equation 2.47 is constructed so that we can initialize a counter `c=2` and perform

```
b_i = b(c:c+size(L,2)-1);
fhat =  fhat + L*b_i;
c=c+size(L,2));.
```

Finally, the pseudocode for the tensor product evaluation is given below:

```
% Evaluate unidimensional basis functions
A_bases = eval_unidim_bases(t,S);

ts = size(t,1);

% Initialize fhat with constant term
fhat=b(1);
c=2;

for each i in Q
    % Initialize in the first dimension
    L=A_sets_bases{i(1),1};

    % Loop through other dimensions
    for di = 2:d
        A=reshape(L,ts,1,[]);
        B=bsxfun(@times,A,A_bases{i(di),di});
        L=reshape(B,ts,[]);
    end

    % Add current piece to Smolyak approximation
    b_i = b(c:c+size(L,2)-1);
    fhat=fhat+L*b_i;

    % Update b index
    b_i=b_i+size(L,2);
end
```

A.2 Documentation for MATLAB Molecular Dynamics Codes

The purpose of this section is to introduce readers to the MATLAB code I have written for studying molecular potential energy surfaces and reaction paths. The code is capable of writing Gaussian 09 input files corresponding to sparse grid points in a given molecular coordinate interpolation domain, reading Gaussian 09 .log files to read molecular energies, plotting and finding minima of interpolated potential energy surfaces, and simulating reaction path dynamics. Section A.2.1 contains documentation for each MATLAB function, Section A.2.2 describes the structure of problem-specific MATLAB scripts that must be written to generate Gaussian 09 input files, and Section A.2.3 gives a step-by-step example.

NOTE: The following assumes that `smolyak_step1_T.m` and `smolyak_approx_T.m` are both in your MATLAB search path. Documentation for these codes can be found in Appendix [reference to be added later].

A.2.1 Function Documentation

- `addStates(molecule,nAddStates)`: creates Gaussian input files according to the data found in the `molecule` data structure to add `nAddStates` excited states to existing TD calculations. The input files are stored in the directory ./*name*/*method*/d*d*/k*k*.

Input	Description
`molecule`	a structure containing information about the jobs to be performed. See section A.2.2 for information on structure fields and A.2.3 for an example.
`nAddStates`	number of excited states to add

- `create_input_files(molecule)`: creates Gaussian input files according to the data found in the `molecule` data structure. The input files are stored in the directory ./*name*/*method*/d*d*/k*k*.

Input	Description
`molecule`	a structure containing information about the jobs to be performed. See section A.2.2 for information on structure fields and A.2.3 for an example.

- `find_MECP(path_to_energy,state_indices,x0)`: computes the minimum energy crossing point between the two surfaces given by `state_indices` using `x0` as an initial guess.

Input	**Description**
`path_to_energy`	string of path to the directory that contains the /tt energy.mat file
`state_indices`	2x1 array of indices of energies matrix (found in `path_to_energy/energy.mat`) corresponding to states to compute MECP between.
Output	**Description**
`xstar`	MECP

- `PES_energy(x,S)`: evaluates the potential energy surface stored in the Smolyak data structure `S` at a point `x`.

Input	**Description**
`x`	point at which to evaluate the PES
`S`	Smolyak data structure for PES (see documentation for `PES_S.m`).
Output	**Description**
`E`	energy of PES at `x`.
`f`	forces (gradient) of PES at `x`.

- `PES_min(path_to_energy,state_index,x0)`: finds a local minimum of the potential energy surface using the energies found in the column corresponding to `state_index` found in energies matrix stored at `path_to_energy/energies.mat`. The function computes the minimum with MATLAB's `fmincon` function using the initial iterate specified by `x0`.

Input	**Description**
`path_to_energy`	string of path to the directory that contains the /tt energy.mat file
`state_index`	column index of energies matrix (found in `energy.mat`) that corresponds to state of interest.
`x0`	initial guess for minimization algorithm
Output	**Description**
`xmin`	local minimum of PES

- `PES_S(path_to_energy,state_index)`: generates the Smolyak data structure for the potential energy surface using the energies stored in the column corresponding to `state_index` found in the energies matrix stored at `path_to_energy/energies.mat`.

Input	Description
`path_to_energy`	string of path to the directory that contains the /tt energy.mat file
`state_index`	column index of energies matrix (found in `energy.mat`) that corresponds to state of interest.
Output	**Description**
`S`	Smolyak data structure

- `plot_intersection_seam(path_to_energy,state_indices,fixed_dims,x0,var_names)`: finds the minimum energy crossing point and plots 1D, 2D, or 3D visualization of the intersection seam between the two PES's. The dimensions specified in the `fixed_dims` input are held constant and the PES is projected onto the remaining dimensions. This input is only necessary for cases where $d > 3$. The input `x0` is used as an initial guess for the minimum energy crossing point (see documentation for `find_MECP`). If unspecified, `x0` is the center of the interpolation domain.

Input	Description
`path_to_energy`	string of path to the directory that contains the /tt energy.mat file
`state_indices`	2x1 array of indices of energies matrix (found in `path_to_energy/energy.mat`) corresponding to states plot intersection seam between.
`fixed_dims`	(required for $d > 3$) fixed dimensions and values for projection given by the matrix `[dim1 val1; dim2 val2; ...]`
`var_names`	(optional) labels for x- and y-axes given by {`'name1', 'name2'`}

- `plot_PES(path_to_energy,state_indices,fixed_dims,var_names)`: plots 3D visualization of PES projected onto two dimensions. The dimensions specified in the `fixed_dims` input are held constant and the PES is projected onto the two remaining dimensions.

Input	Description
`path_to_energy`	string of path to the directory that contains the /tt energy.mat file
`state_indices`	array of indices of energies matrix (found in `path_to_energy/energy.mat`) corresponding to states to plot.
`fixed_dims`	fixed dimensions and values for 2D projection plot given by the matrix `[dim1 val1; dim2 val2; ...]`
`var_names`	(optional) labels for x- and y-axes given by {`'name1', 'name2'`}

- `plot_contour(path_to_energy,state_index,fixed_dims,var_names)`: plots contour plot visualization of PES projected onto two dimensions. The dimensions specified in the `fixed_dims` input are held constant and the PES is projected onto the two remaining dimensions.

Input	Description
`path_to_energy`	string of path to the directory that contains the `energy.mat` file
`state_index`	column index of energies matrix (found in `energy.mat`) that corresponds to state of interest.
`fixed_dims`	fixed dimensions and values for 2D projection plot given by the matrix `[dim1 val1; dim2 val2; ...]`
`var_name`	(optional) labels for x- and y-axes given by {`'name1', 'name2'`}

- `read_logs(path_to_files,path_to_prev_k_files,NaddStates)`: reads Gaussian .log files and stores energies (in eVs) in `path_to_files/energies.mat`. For each .log file there are five possible outcomes.
  - The calculation successfully converged. No further actions are needed.
  - The calculation terminated via Link 9999, meaning the optimization failed to converge and needs to be restarted. In this case `read_logs` will write a new .com file that will restart the optimization from the checkpoint file. All jobs that fail in this way can be resubmitted by running `bash resubmit_failed_opts.sh` from the command line.
  - The calculation terminated because of a failed SCF calculation. The user must make appropriate changes to the .com file and resubmit the job manually.
  - The calculation terminated because of a hardware issue. On the -gto queue I have experienced two additional failure modes which I call "NaN" and "ITU=***," both of which are the result of a hardware issue on the blade. In this case `read_logs` will write a new .com file that will restart the optimization from a step in the checkpoint file before either the "NaN" or "ITU=**" occurred. All jobs that fail in these ways can be resubmitted by running `bash resubmit_thrown_nans.sh` or `bash resubmit_null_itu.sh` from the command line.
  - The calculation terminated for some unforeseen reason (e.g. a file was accidentally deleted during the calculation). All jobs that fail in this way can be resubmitted by running `bash resubmit_failed_others.sh` from the command line, but it is recommended that the user inspect each of these jobs to discover the failure reason before doing so.

Input	**Description**
`path_to_files`	string of path to the directory that contains the .log files to be read.
`path_to_prev_k_files`	(optional) string of path to the directory that contains the /tt energy.mat file from the k-1 sparse grid
`NaddStates`	(optional) the number of states added to TD calculations (required to read jobs created by `addStates.m`)
Output	**Description**
`energies`	A $M \times N$ matrix of energies where M is the number of sparse grid points and N is the number of electronic states.

- `reaction_path(path_to_energy,x0,stateSequence,gammaRange,nSims)`: simulates reaction path dynamics.

Input	**Description**
`path_to_energy`	directory where the /tt energy.mat file is stored.
`x0`	initial point for simulations.
`stateSequence`	electronic state sequence for simulations.
`gammaRange`	vector specifying radius for each degree of freedom's approximated molecular vibration/thermal fluctuation effects.
`nSims`	number of reaction paths to follow.
Output	**Description**
`yPath`	$N \times (d \cdot nSims)$ matrix of reaction paths where N is the number of time steps. The first d columns are the paths the d degrees of freedom take for the first reaction path, the next d columns are the paths the d degrees of freedom take for the second reaction path, and so forth.
`ePath`	$N \times nSims$ matrix of reaction path energies.
`yFinal`	returns the result of `Ypath(end, :)`

A.2.2 Molecule Input Scripts

Input scripts are used to create a data structure called `molecule` that stores information about the molecule and electronic structure calculations. This data structure is then fed to `create_input_files.m` to create all required Gaussian 09 input files.

The first thing to do in an input script is initialize the `molecule` structure by reading in the z-matrix from a .zmat file. This can be accomplished by calling `molecule = read_zmat('your_molecule.zmat');`. Next, add fields to the `molecule` structure that provide details about the electronic structure calculations. Table A.2 describes the data fields that need to be user-specified. After all data fields have been set, create the Gaussian .com files by calling `create_input_files(molecule);`. Alternatively, to add excited states to existing calculations, call `addStates(molecule, N);` where `N` is the number of states to add. For each .com file, `create_input_files` or `addStates` create an associated hpc job submission script outlined as follows for the first job for butene:

```
-------BEGIN FILE--------------------
#!/bin/csh

#BSUB -o butene1.out
#BSUB -e butene1.err
#BSUB -n 8
#BSUB -R "span[ptile=8]"
#BSUB -q gto
source /home/gwhowell/scripts/mvapich-intel/int101_mvapich.csh
#BSUB -W 400:00
#BSUB -J butene

mkdir /scratch/unity_id
setenv GAUSS_SCRDIR /scratch/unity_id
g09 butene1.com
rm -rf /scratch/unity_id
------------END FILE--------------------
```

Finally, both of these functions create a bash script called `submit_files.sh` that needs to be executed on the henry2 to submit all the jobs. This can be done on the Terminal command line with `$ bash submit_files.sh`.

Table A.2: User-specified data fields for the `molecule` data structure.

`DOF`	A cell containing strings of code specifying which degree of freedom corresponds to which molecular coordinate. For example, `'D(5)=DOF(i,1);'` would set the 5th dihedral angle to the 1st degree of freedom.
`atom`	Cell of atomic symbols listed in order as they appear in the molecule's z-matrix.
`bases`	A cell containing two basis sets: the first is for the ground state geometry optimizations and the second is for ground state point calculations to be performed after the optimization. If these bases are the same, no point calculation is performed. If `gen` is used for the basis set, the data field `basis_options` must be specified.
`basis_options`	A cell comprised of lines of the `gen` basis set input section of Gaussian input files, e.g. `{'Fe 0','sdd','****','N C H 0','6-311g*'}`.
`bounds`	Matrix containing the sparse grid interpolation domain, given in the form `[A1 B1; A2 B2; ... Ad Bd];`.
`charge`	Charge of molecule.
`frozen`	A cell containing the coordinates to be frozen during Opt=Z-Matrix calculations. For example, `{'D1','D2'};` would freeze the first two dihedral angles.
`k`	Degree of polynomial exactness for Smolyak's sparse grid interpolation algorithm.
`modredun`	Cell of strings where each string is a line of an Opt=ModRedundant modify redundant coordinates section. For example, `{'D 1 2 3 4 F','B 2 3 F'};` would freeze the dihedral angel created by atoms 1-2-3-4 and the bond distance between atoms 2-3.
`multiplicity`	Spin multiplicity of molecule.
`name`	Name of molecule.
`pop_options`	Population analysis options, e.g. `Full`. If not specified, no population analysis is performed.
`opt_coord`	Coordinates for geometry optimization. Use either `'z-matrix'` or `'ModRedundant'`.
`opt_method`	Method for geometry optimizations, e.g. `b3lyp`.
`other_options`	Other options ground state calculations to be included in the route section of the .com file, e.g. `'NoSymm'` or `Int=UltraFine`.
`point_method`	Method for point calculations, e.g. `b3lyp`.
`prev_k`	Set to `1` if the k-1 sparse grid has already been calculated. Set to `0` otherwise.
`scf_options`	Options for SCF procedure, e.g. `'VTL, XQC'`.
`scrf_options`	Options for SCRF procedure, e.g. `'PCM, Solvent=Water, Read'`. If not specified, SCRF is not used in the calculations.
`td_options`	Options for TD calculation, e.g. `'Root=1, NStates=10'`.
`td_other_options`	Other options for excited state calculations to be included in the route section of the Gaussian input file, e.g. `'NoSymm Int=UltraFine'`.
`unity_id`	NCSU unity ID for use on the henry2 cluster.

A.2.3 2-Butene Example

Here we will show an example of a singlet 2-butene input script. We will perform a geometry optimization using the 3-21G* basis set, followed by a point calculation using the CEP-31G* basis set. We will also use TD to compute the first three singlet excited states. We will use the B3LYP DFT functional for all calculations. The 2-butene z-matrix, as written in `butene.zmat`, is shown in Figure A.1. Note the use of 0's in place of blanks for the first few lines. This is necessary for how `read_zmat` reads the .zmat file.

```
C 0 0   0 0   0 0    0
C 1 1.4 0 0   0 0    0
C 2 1.4 1 125 0 0    0
C 3 1.4 2 125 1 0.0  0
H 4 1.1 3 114 2 0.0  0
H 4 1.1 3 114 5 120  0
H 4 1.1 3 114 5 -120 0
H 1 1.1 2 114 3 0    0
H 1 1.1 2 114 8 120  0
H 1 1.1 2 114 8 -120 0
H 2 1.1 3 115 1 180  0
H 3 1.1 2 115 4 180  0
```

Figure A.1: `butene.zmat`: z-matrix file for butene.

We choose three degrees of freedom for this example: `D1`, `D2`, and `D5`. These correspond to the dihedral angle formed by the 4 C atoms, and one C-C-C-H dihedral angle for each methyl group. We will use a degree of exactness of $k = 5$ for Smolyak's sparse grid interpolation algorithm. We will also perform a full population analysis after the point calculation and excited states calculation. The entire MATLAB input script is shown in Figure A.2 and can be run from the MATLAB command line.

```
% Name .zmat file
zmat_file = 'butene.zmat';

% Populate fields of molecule data structure
molecule = read_zmat(zmat_file);
molecule.name='butene';
molecule.charge=0;
molecule.multiplicity=1;
molecule.opt_method='b3lyp';
molecule.point_method='b3lyp';
opt_basis = '3-21G*';
point_basis = 'CEP-31G*';
molecule.bases={opt_basis, point_basis};
molecule.k=5;
molecule.prev_k=0;
molecule.bounds=[-20 200; -100 100; -100 100];
molecule.opt_options = 'VeryTight,CalcFC';
molecule.opt_coord = 'Z-Matrix';
molecule.scf_options='VTL,xqc,maxConventionalCycles=1000';
molecule.other_options='Symm=None Int=UltraFine';
molecule.pop_options = 'Full';
molecule.td_options='Root=1,NStates=3';
molecule.td_other_options='Symm=None Int=UltraFine';
molecule.DOF={'D(1)=DOF(i,1); '...
              'D(2)=DOF(i,2);'...
              'D(5)=DOF(i,3);'};
molecule.frozen={'D1','D2','D5'};
molecule.unity_id='jdnance2';

% Call create_input_files
create_input_files(molecule);
```

Figure A.2: `butene_input.m`: an example of an input script for butene.

After executing `submit_files.sh` and all Gaussian jobs have finished, we can read the .log files and extract energies by executing

```
>> [energies, failures, successes] = read_logs('butene/b3lyp/d3/k5');
```

in the MATLAB command line. If some optimizations failed to converge, `read_logs` will create a bash script called `resubmit_failed_opts.sh` that can be executed to resubmit failed optimizations by reading the initial geometry from the job's checkpoint file. Similarly, if some jobs fail for other reasons where a restart would be appropriate (I've seen invalid read/write issues on henry2 which can be fixed by simply restarting the calculation), `read_logs` will create a bash script called `resubmit_failed_others.sh`. Finally, if any jobs fail during an SCF procedure, `read_logs` will list these jobs so appropriate modifications can be made by the user to the .com files (e.g. using less strict convergence criteria or changing basis sets). If all jobs are successful then `read_logs` will store the energies as a matrix in `energy.mat` in the directory `butene/b3lyp/d3/k5/`.

Now that we have performed all the necessary electronic structure calculations, we can visualize the PES's and run dynamics simulations. As an example of how to visualize the PES, we will project the 3-dimensional PES for the ground and first excited singlet state onto the first two degrees of freedom and hold the 3rd constant at its equilibrium value of $x_3 = 0$. This can be performed by executing

```
>> names = {'C=C Rotation', 'Methyl Rotation 1'};
>> plot_PES('butene/b3lyp/d3/k5',[1 2],[3 0],names);
```

Alternatively, we can view a contour plot of the ground state PES by executing

```
>> plot_contour('butene/b3lyp/d3/k5',1,[3 0],names);
```

Figures A.3 and A.4 show the results of these two commands.

Finally, we will compute a reaction path for the *cis-trans* photoisomerization of 2-butene. Our electronic state sequence is {0,1,0}, the initial point for our simulations is $\mathbf{x} = (1.0, 0.0, 0.0)$, each degree of freedom's molecular vibration/thermal fluctuation radius is 20, and we compute 50 reaction paths. The simulation can be started with

```
>> N = 50;
>> x0 = [1 0 0];
>> seq = [0 1 0];
>> gamma = [20 20 20];
>> [yPath, ePath, yFinal] = reaction_path('butene/b3lyp/d3/k5',x0,seq,gamma,N);
```

Figures A.5-A.8 shows the results of the reaction path simulation for each degree of freedom and the energies of all 50 reaction paths.

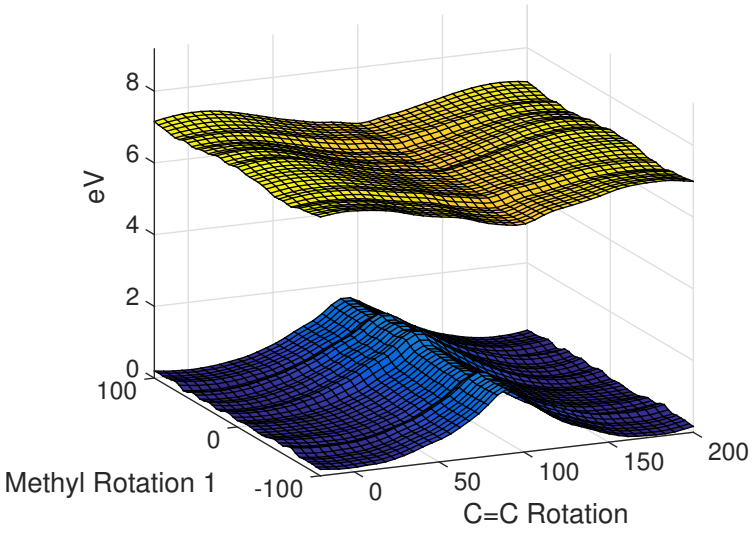


Figure A.3: Ground and lowest-lying excited singlet state PES's for 2-butene projected onto the first two degrees of freedom.

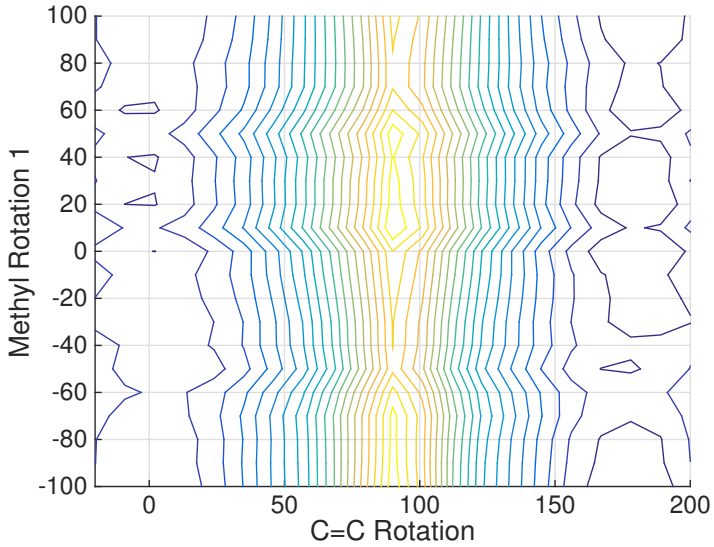


Figure A.4: Contour plot of ground state PES for 2-butene projected onto the first two degrees of freedom.

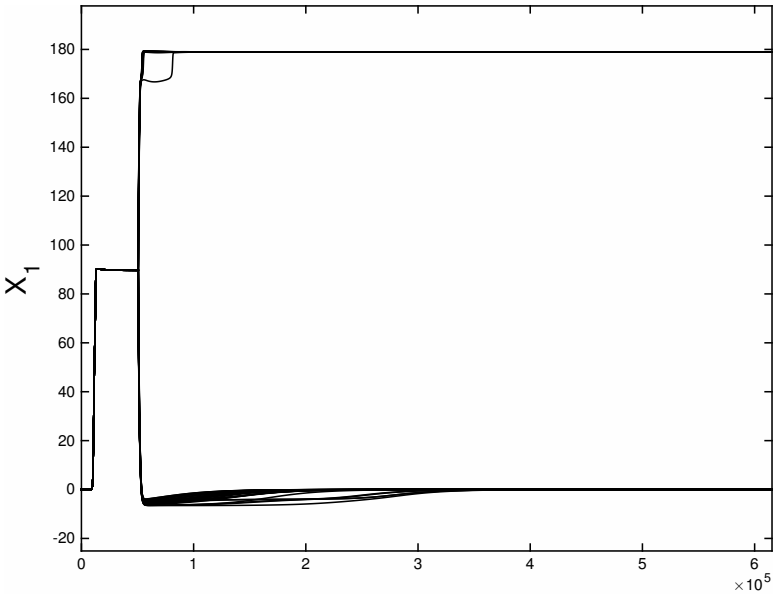


Figure A.5: Reaction paths for x_1.

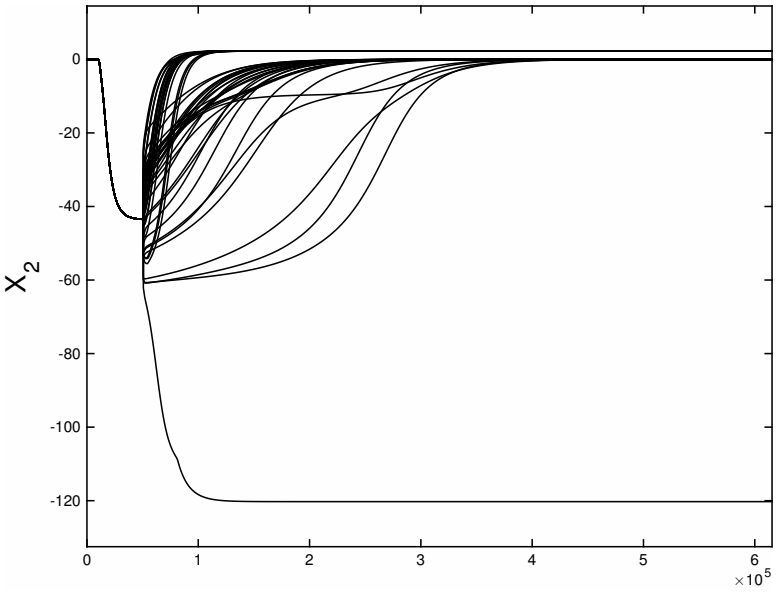


Figure A.6: Reaction paths for x_2.

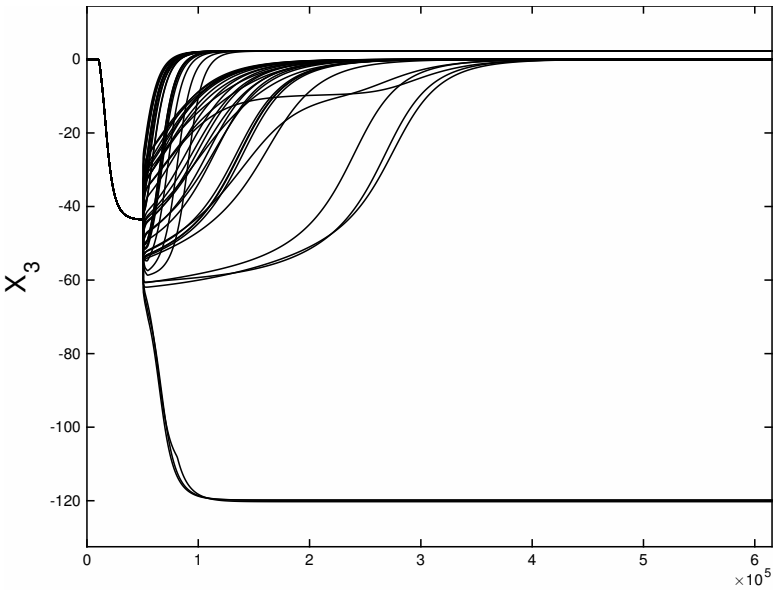


Figure A.7: Reaction paths for x_3.

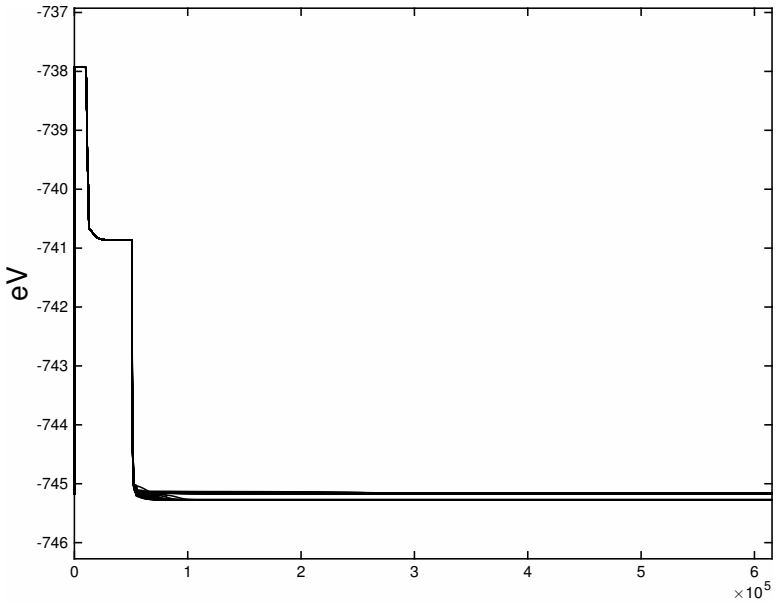


Figure A.8: Energies of reaction paths.

A.2.4 Troubleshooting Gaussian Optimizations

Computational chemistry algorithms, especially Gaussian's optimization algorithm, are finicky and very sensitive to input parameters. In this section we survey the most common issues and offer some possible solutions.

Error Messages and Solutions

- "Error: Old/New curvilinear step failed..." means the SCF algorithm has failed.
  - If you are using the default algorithm (a variation of DIIS from [128]), try using alternative algorithms for the SCF method. The Gaussian 09 options SCF=XQC or SCF=QC take more computer time but converge more frequently than DIIS. Try SCF=XQC first. It performs an iterate of the QC algorithm in the event of a failed DIIS iterate. QC stands for "quadratically convergent" and the algorithm can be found in [7]. A great reference for tips on SCF convergence can be found in [184].
  - Use higher accuracy integrals, a finer integration grid, or both. The Gaussian 09 option to set the integral tolerance to 10^{-N} is Int=(Acc2E=N), and the option to use a finer grid is Int=VeryFine or Int=UltraFine.
- "Maximum number of iterations exceeded" means the optimization algorithm failed to converge in the allowed number of optimization iterations. Unlike SCF, we cannot increase the maximum number of iterations, but there are other solutions. In addition to the solutions mentioned above, also consider the following:
  - Try a slightly different input geometry by shortening or lengthening one or two of the bond distances. Also, Gaussian's optimization algorithm seems to not like dihedral angles close to 180 degrees, so this may require reconstructing the z-matrix to reduce the number of angles close to 180.
  - Allow more SCF iterations. This is done using the SCF=(MaxCycle=N) option, where N is the maximum number of iterations.
  - Consider using a different (smaller) basis set, level of theory, or both.
  - Request that the algorithm compute analytic Hessians at the first step. This is done with the Opt=CalcFC option.
- "Small interatomic distances encountered . . . Atoms too close."
  - Include the Geom=NoCrowd option to disable distance checking.

Remarks

- If you are requesting higher accuracy optimizations via the Opt=VeryTight option, make sure to request higher accuracy integrals with Int=VeryFine, too.

A.3 Semiclassical electron-radiation-ion dynamics (SERID)

In this section we outline a method called semiclassical electron-radiation-ion dynamics (SERID), which is a technique for simulating the coupled dynamics of valence electrons and ion cores (nuclei) in a molecule. The method was introduced by Allen, Dumitrica and Torralva in [3], where it is called tight-binding electron-ion dynamics (TED), and used in [57] and [114] to study the isomerization of stilbene. The method treats valence electrons quantum-mechanically but treats the radiation field and the motion of the ion cores classically. According to [3], this treatment is valid since the masses of the ions are $10^4 - 10^5$ times larger than electron mass. SERID is presented as an alternative to the popular Born-Oppenheimer approximation and allows one to study processes such as multiple electronic and vibrational excitations, intramolecular vibrational energy redistribution, and interdependence of the various electronic and vibrational degrees of freedom [57]. SERID is an $O(N)$ method where N is the number of atoms in the system [3].

SERID calculates the forces on the ion cores and updates the wave function at every time step. The wavefunctions are found via the time-dependent Schrödinger equation

$$i\hbar\frac{\partial\Psi_j}{\partial t} = S^{-1}H\Psi_j \tag{A.1}$$

where S is the overlap matrix for the atomic orbitals and j is an index over the electrons [3]. Note that the product $S^{-1}H$ is a Hermitian operator since: first, the overlap matrix S is Hermitian and symmetric so S^{-1} is also Hermitian and symmetric; second, the Hamiltonian H is Hermitian; and third, $S^{-1}H = HS^{-1}$ since S and thus S^{-1} are symmetric, so the product $S^{-1}H$ is also Hermitian. A laser pulse is introduced into the system by coupling a time-dependent vector potential $A(t)$ to the electronic Hamiltonian via the Peierls substitution

$$H_{ab}(X - X') = H^0_{ab}(X - X')\exp\left(\frac{iq}{\hbar c}A(t)(X - X')\right) \tag{A.2}$$

with

$$H^0 = -\frac{\hbar\nabla^2}{2m} + V(x). \tag{A.3}$$

Here, X and X' are nuclear coordinates, a and b label atomic orbitals, q is the charge of an electron, and c is the speed of light. Finally, the motion of the ion cores is described by Ehrenfest's theorem

$$M_\ell\ddot{X}_{\ell\alpha} = -\frac{1}{2}\sum_j \Psi_j^*\left(\frac{\partial H}{\partial X_{\ell\alpha}} - i\hbar\frac{\partial S}{\partial X_{\ell\alpha}}\frac{\partial}{\partial t}\right)\Psi_j + \texttt{h.c.} - \frac{\partial U_{\texttt{rep}}}{\partial X_{\ell\alpha}} \tag{A.4}$$

where the parameters determining the Hamiltonian matrix H and ion-ion interaction $U_{\texttt{rep}}$ are

fitted to first-principles densitfy-functional calculations, and "`h.c.`" represents the Hermitian conjugate of the first term [57]. Derivations and justifications of these formulae, along with further analysis, can be found in [3].

The solution to Equation A.4 is approximated via the velocity Verlet algorithm [218]. The algorithm integrates the general equation of motion

$$m\ddot{r}_i = \sum_{j \neq i} f\left(r_{ij}\right) \tag{A.5}$$

via

$$r_i(t+h) = -r_i(t-h) + 2r_i(t) + \sum_{j \neq i} f\left(r_{ij}(t)\right) h^2 \tag{A.6}$$

where h is the time step. At each step the algorithm must compute $\frac{1}{2}N(N-1)$ terms, but Verlet introduces a bookkeeping device that reduces reduces the computing time by a factor of the order of 10 [218].

The solution to Equation A.1 requires a different approach. Following [3], the time-evolution equation can be written in the form

$$\exp\left(\frac{iH\Delta t}{2\hbar}\right)\Psi_j(t+\Delta t) = \exp\left(\frac{-iH\Delta t}{2\hbar}\right)\Psi_j(t). \tag{A.7}$$

Approximating the exponential by its first two terms in its Taylor series yields the Cayley algorithm

$$\Psi_j(t+\Delta t) = \left(1+\frac{iH\Delta t}{2\hbar}\right)^{-1}\left(1-\frac{iH\Delta t}{2\hbar}\right)\Psi_j(t). \tag{A.8}$$

This algorithm is ideal because it preserves the probability and the orthogonality of the system [3]. Torralva introduced an improvement to this method in [3] where the first-order term in a Dyson-like series for the time evolution operator $U(t+\Delta t)$ is written as

$$U(t+\Delta t, t) = \left(\frac{1+i\bar{H}}{2\hbar}\right)^{-1}\left(\frac{1-i\bar{H}}{2\hbar}\right) \tag{A.9}$$

with

$$\bar{H} = \frac{1}{\Delta t}\int_t^{t+\Delta t} dsH(s). \tag{A.10}$$

After approximating the elements of $\bar{H}$ with a sufficient quadrature rule, the electron states are obtained from

$$\Psi_j(t+\Delta t) = U(t+\Delta t, t)\Psi_j(t). \tag{A.11}$$

This algorithm preserves orthonormality of Ψ_j to the machine accuracy of better than 10^{-12}

[3].

SERID was originally developed in the context of semiconductors and is the recommended method for simulations of the interaction of light and matter. DFT methods are not suitable for such problems because the excited states are too low for semiconductors and nonadiabatic processes require very small time steps [3]. SERID has been used to study elements and molecules including gallium arsenide, silicon and H_2^+ [3], butadiene [58], and stilbene [57, 114].

A.4 Modified Shepard Interpolation

In this section we outline a potential energy surface interpolation scheme originally proposed by Ischtwan and Collins in [109]. The scheme is a modified Shepard interpolation and is related to a moving least squares interpolation procedure. There are several variations of and improvements to this method in the literature (see [64], for example), but the core algorithm remains the same so we present the scheme in it's original form.

Consider a molecule with N atoms and let $R \in \mathbb{R}^{3N-6}$ be a column vector of interatomic distances. We will let $\{R(i),\ i = 1, \ldots, N_d\}$ denote a set of N_d configurations of the molecule at which the potential energy V is known. Using a Taylor series expansion for the potential energy, we have

$$U(R;i) = VR(i) + [R - R(i)]^T G(i) + \frac{1}{2}[R - R(i)]^T F(i)[R - R(i)] + \ldots \tag{A.12}$$

where $G(i)$ and $F(i)$ are the gradient vector and the Hessian at $R(i)$, respectively. Ischtwan and Collins employ inverse length coordinates $\rho_n = \frac{1}{R_n}$ in the Taylor series, yielding

$$V(\rho(R);i) = VR(i) + [\rho - \rho(i)]^T G_\rho(i) + \frac{1}{2}[\rho - \rho(i)]^T F_\rho(i)[\rho - \rho(i)] + \ldots \tag{A.13}$$

where $G_\rho(i)$ and $F_\rho(i)$ are the analogous gradient vector and the Hessian in inverse coordinates.

One of the goals of reaction-path based potentials is to choose an optimal configuration $R(i)$ on the reaction path about which to center the Taylor expansion. The potential energy of any configuration can be given by a modified Shepard interpolation, a scheme that gives the potential energy surface as a weighted average of the Taylor expansions of the PES about each point in the configuration $R(i)$. More explicitly, the potential energy becomes

$$V(R) = \sum_{i=1}^{N_d} w_i(R)[V\rho(R);i] \tag{A.14}$$

where the weight function $w_i(R)$ has the form

$$w_i(R) = \frac{v_i(R)}{\sum_{k=1}^{N_d} v_k(R)}. \tag{A.15}$$

The functions $v_i(R)$ are unnormalized weight functions such that $v_i(R) \to 0$ as $|R - R_i| \to \infty$ and $v_i(R) \to \infty$ as $|R - R_i| \to 0$. Ischtwan and Collins adopt the form

$$v_i(R) = \frac{1}{|R - R(i)|^p} \tag{A.16}$$

for a parameter p. Assuming distinct configurations $R(i)$, as $|R - R_i| \to 0$, $w_i(R) \to 1$ and $w_j(R) \to 0$ for $j \neq i$. Furthermore, the N_d weights $w_i(R)$ sum to 1 and thus

$$\sum_{k=1}^{N_d} \frac{\partial w_k(R)}{\partial R} = 0. \tag{A.17}$$

With these properties and with $p > 2$, $V(R)$ will exactly interpolate the energy, gradient, and Hessian at $R = R(i)$ for $i = 1, \ldots, N_d$. In general, if the Taylor series in Equation A.13 is truncated after the nth order, the function, gradient, and Hessian will be interpolated correctly if $p > n$ [109].

A.5 Velocity Verlet Algorithm [205]

:

$$\ddot{\mathbf{R}} = f(\mathbf{R}) \tag{A.18}$$

$$\frac{d^2\mathbf{R}}{dt^2} = -\frac{1}{M}\frac{\partial V_k}{\partial \mathbf{R}}. \tag{A.19}$$

Letting h be the time step for numerical integration, define $t_n = nh$. The Verlet algorithm uses the following approximations for first and second derivatives:

$$\dot{\mathbf{R}}_n = \frac{\mathbf{R}_{n+1} - \mathbf{R}_{n-1}}{2h} \tag{A.20}$$

$$\ddot{\mathbf{R}}_n = \frac{\mathbf{R}_{n+1} - 2\mathbf{R}_n + \mathbf{R}_{n-1}}{h^2}. \tag{A.21}$$

Treating Equation A.21 as an equality yields the Verlet algorithm:

$$\mathbf{R}_{n+1} = 2\mathbf{R}_n - \mathbf{R}_{n-1} + h^2 f(\mathbf{R}_n). \tag{A.22}$$

Equation A.20 is used to compute the velocity, if it is needed.

The velocity form of the Verlet algorithm is

$$\mathbf{R}_{n+1} = \mathbf{R}_n + h\mathbf{V}_n + \frac{h^2}{2} f(\mathbf{R}_n), \tag{A.23}$$

$$\mathbf{V}_{n+1} = \mathbf{V}_n + \frac{h}{2} \left(f(\mathbf{R}_{n+1}) + f(\mathbf{R}_n) \right). \tag{A.24}$$

This formulation is mathematically equivalent to the Verlet algorithm, but is more numerically stable.

A.6 Z-Matrices

A.6.1 2-Butene

The 2-butene z-matrix with $x_1 = -20.0$ and $x_2 = -100.0$ is printed below. Note that x_1 corresponds to D1 and x_2 corresponds to D2.

```
0 1
C
C 1 B1
C 2 B2 1 A1
C 3 B3 2 A2 1 D1 0
H 4 B4 3 A3 2 D2 0
H 4 B5 3 A4 5 D3 0
H 4 B6 3 A5 5 D4 0
H 1 B7 2 A6 3 D5 0
H 1 B8 2 A7 8 D6 0
H 1 B9 2 A8 8 D7 0
H 2 B10 3 A9 1 D8 0
H 3 B11 2 A10 4 D9 0

B1=1.450000
B2=1.450000
B3=1.450000
B4=1.100000
B5=1.100000
B6=1.100000
B7=1.100000
B8=1.100000
```

```
B9=1.100000
B10=1.100000
B11=1.100000
A1=125.000000
A2=125.000000
A3=114.000000
A4=114.000000
A5=114.000000
A6=114.000000
A7=114.000000
A8=114.000000
A9=115.000000
A10=115.000000
D1=-20.000000
D2=-100.000000
D3=120.000000
D4=-120.000000
D5=0.000000
D6=120.000000
D7=-120.000000
D8=180.000000
D9=180.000000
```

A.6.2 Stilbene

The z-matrix for stilbene with $x_1 = -200$, $x_2 = -100$ and $x_3 = -150$ is

```
C
C 1 B1
C 2 B2 1 A1
C 3 B3 2 A2 1 D1 0
C 4 B4 3 A3 2 D2 0
C 5 B5 4 A4 3 D3 0
H 2 B6 1 A5 6 D4 0
H 3 B7 2 A6 7 D5 0
H 4 B8 3 A7 8 D6 0
H 5 B9 4 A8 9 D7 0
H 6 B10 5 A9 10 D8 0
```

```
C 1 B11 2 A10 7 D9 0
C 12 B12 1 A11 2 D10 0
C 13 B13 12 A12 1 D11 0
C 14 B14 13 A13 12 D12 0
C 15 B15 14 A14 13 D13 0
C 16 B16 15 A15 14 D14 0
C 17 B17 16 A16 15 D15 0
C 18 B18 17 A17 16 D16 0
H 19 B19 14 A18 13 D17 0
H 18 B20 19 A19 20 D18 0
H 17 B21 18 A20 21 D19 0
H 16 B22 17 A21 22 D20 0
H 15 B23 16 A22 23 D21 0
H 13 B24 14 A23 19 D22 0
H 12 B25 1 A24 6 D23 0

B1=1.4
B2=1.4
B3=1.4
B4=1.4
B5=1.4
B6=1.09
B7=1.09
B8=1.09
B9=1.09
B10=1.09
B11=1.4
B12=1.4
B13=1.4
B14=1.4
B15=1.4
B16=1.4
B17=1.4
B18=1.4
B19=1.09
B20=1.09
B21=1.09
```

B22=1.09
B23=1.09
B24=1.09
B25=1.09
A1=120.0
A2=120.0
A3=120.0
A4=120.0
A5=120.0
A6=120.0
A7=120.0
A8=120.0
A9=120.0
A10=120.0
A11=120.0
A12=120.0
A13=120.0
A14=120.0
A15=120.0
A16=120.0
A17=120.0
A18=120.0
A19=120.0
A20=120.0
A21=120.0
A22=120.0
A23=110.0
A24=110.0
D1=0.0
D2=0.0
D3=0.0
D4=180.0
D5=0.0
D6=0.0
D7=0.0
D8=0.0
D9=0.0

```
D10=-150.0
D11=-200.0
D12=-100.0
D13=180.0
D14=0.0
D15=0.0
D16=0.0
D17=0.0
D18=0.0
D19=0.0
D20=0.0
D21=0.0
D22=-100.0
D23=-150.0
```

where x_1 corresponds to D11, x_2 corresponds to D12 with preprocessed angle D22 and x_3 corresponds to D10 with preprocessed angle D23.

A.6.3 2-Pentene

The z-matrix for 2-pentene with $x_1 = 0$ and $x_2 = 0$ is

```
C
C 1 B1
C 2 B2 1 A1
C 3 B3 2 A2 1 D1
C 4 B4 3 A3 2 D2
H 1 B5 2 A4 3 D3
H 1 B6 2 A5 3 D4
H 1 B7 2 A6 3 D5
H 2 B8 1 A7 7 D6
H 3 B9 2 A8 9 D7
H 5 B10 4 A9 3 D8
H 5 B11 4 A10 3 D9
H 5 B12 4 A11 3 D10
H 4 B13 5 A12 11 D11
H 4 B14 5 A13 11 D12

B1=1.4
```

B2=1.4
B3=1.4
B4=1.4
B5=1.1
B6=1.1
B7=1.1
B8=1.1
B9=1.1
B10=1.1
B11=1.1
B12=1.1
B13=1.1
B14=1.1
A1=120.0
A2=120.0
A3=120.0
A4=120.0
A5=120.0
A6=120.0
A7=120.0
A8=120.0
A9=120.0
A10=120.0
A11=120.0
A12=120.0
A13=120.0
D1=0.0
D2=180.0
D3=-120.0
D4=0.0
D5=120.0
D6=180.0
D7=0.0
D8=180.0
D9=300.0
D10=60.0
D11=60.0

```
D12=-60.0
```

A.6.4 $[Fe(tpy)_2]^{2+}$

The z-matrix for $[Fe(tpy)_2]^{2+}$ with axial bond lengths and equatorial bond lengths equal to 2.065 Åand a tilting angle of 0° is given below.

```
Fe
X 1 1.200000
X 1 1.200000 2 90.000000
N 1 2.065000 2 90.000000 3 D1 0
N 1 2.065000 4 A3 3 D2 0
N 1 2.065000 4 A4 5 D3 0
N 1 2.065000 3 90.000000 2 90.000000 0
N 1 2.065000 7 A6 2 D5 0
N 1 2.065000 7 A7 8 D6 0
C 4 B9 1 A8 5 D7 0
C 10 B10 4 A9 1 D8 0
C 11 B11 10 A10 4 D9 0
C 12 B12 11 A11 10 D10 0
C 4 B13 10 A12 11 D11 0
C 5 B14 1 A13 4 D12 0
C 15 B15 5 A14 1 D13 0
C 16 B16 15 A15 5 D14 0
C 17 B17 16 A16 15 D15 0
C 18 B18 17 A17 16 D16 0
C 6 B19 1 A18 4 D17 0
C 20 B20 6 A19 1 D18 0
C 21 B21 20 A20 6 D19 0
C 22 B22 21 A21 20 D20 0
C 23 B23 22 A22 21 D21 0
H 15 B24 16 A23 17 D22 0
H 16 B25 15 A24 5 D23 0
H 17 B26 16 A25 15 D24 0
H 18 B27 17 A26 16 D25 0
H 13 B28 12 A27 11 D26 0
H 12 B29 13 A28 14 D27 0
H 11 B30 12 A29 13 D28 0
```

```
H 23 B31 24 A30 6 D29 0
H 22 B32 23 A31 24 D30 0
H 21 B33 22 A32 23 D31 0
H 20 B34 21 A33 22 D32 0
C 7 B35 1 A34 8 D33 0
C 36 B36 7 A35 1 D34 0
C 37 B37 36 A36 7 D35 0
C 38 B38 37 A37 36 D36 0
C 7 B39 36 A38 37 D37 0
C 8 B40 1 A39 7 D38 0
C 41 B41 8 A40 1 D39 0
C 42 B42 41 A41 8 D40 0
C 43 B43 42 A42 41 D41 0
C 8 B44 41 A43 42 D42 0
C 9 B45 1 A44 7 D43 0
C 46 B46 9 A45 1 D44 0
C 47 B47 46 A46 9 D45 0
C 48 B48 47 A47 46 D46 0
C 9 B49 46 A48 47 D47 0
H 41 B50 42 A49 43 D48 0
H 42 B51 43 A50 44 D49 0
H 43 B52 44 A51 45 D50 0
H 44 B53 45 A52 8 D51 0
H 39 B54 38 A53 37 D52 0
H 38 B55 37 A54 36 D53 0
H 37 B56 36 A55 7 D54 0
H 49 B57 50 A56 9 D55 0
H 48 B58 49 A57 50 D56 0
H 47 B59 48 A58 49 D57 0
H 46 B60 47 A59 48 D58 0

B9=1.350000
B10=1.380000
B11=1.380000
B12=1.380000
B13=1.350000
B14=1.350000
```

B15=1.380000
B16=1.380000
B17=1.380000
B18=1.380000
B19=1.350000
B20=1.380000
B21=1.380000
B22=1.380000
B23=1.380000
B24=1.080000
B25=1.080000
B26=1.080000
B27=1.080000
B28=1.080000
B29=1.080000
B30=1.080000
B31=1.080000
B32=1.080000
B33=1.080000
B34=1.080000
B35=1.350000
B36=1.380000
B37=1.380000
B38=1.380000
B39=1.350000
B40=1.350000
B41=1.380000
B42=1.380000
B43=1.380000
B44=1.350000
B45=1.350000
B46=1.380000
B47=1.380000
B48=1.380000
B49=1.350000
B50=1.080000
B51=1.080000

B52=1.080000
B53=1.080000
B54=1.080000
B55=1.080000
B56=1.080000
B57=1.080000
B58=1.080000
B59=1.080000
B60=1.080000
A3=76.000000
A4=76.000000
A6=76.000000
A7=76.000000
A8=120.000000
A9=120.000000
A10=120.000000
A11=120.000000
A12=120.000000
A13=120.000000
A14=120.000000
A15=120.000000
A16=120.000000
A17=120.000000
A18=120.000000
A19=120.000000
A20=120.000000
A21=120.000000
A22=120.000000
A23=120.000000
A24=120.000000
A25=120.000000
A26=120.000000
A27=120.000000
A28=120.000000
A29=120.000000
A30=120.000000
A31=120.000000

A32=120.000000
A33=120.000000
A34=120.000000
A35=120.000000
A36=120.000000
A37=120.000000
A38=120.000000
A39=120.000000
A40=120.000000
A41=120.000000
A42=120.000000
A43=120.000000
A44=120.000000
A45=120.000000
A46=120.000000
A47=120.000000
A48=120.000000
A49=120.000000
A50=120.000000
A51=120.000000
A52=120.000000
A53=120.000000
A54=120.000000
A55=120.000000
A56=120.000000
A57=120.000000
A58=120.000000
A59=120.000000
D1=90.000000
D2=0.000000
D3=180.000000
D5=0.000000
D6=180.000000
D7=180.000000
D8=180.000000
D9=0.000000
D10=0.000000

D11=0.000000
D12=180.000000
D13=180.000000
D14=0.000000
D15=0.000000
D16=0.000000
D17=180.000000
D18=180.000000
D19=0.000000
D20=0.000000
D21=0.000000
D22=180.000000
D23=180.000000
D24=180.000000
D25=180.000000
D26=180.000000
D27=180.000000
D28=180.000000
D29=180.000000
D30=180.000000
D31=180.000000
D32=180.000000
D33=180.000000
D34=180.000000
D35=0.000000
D36=0.000000
D37=0.000000
D38=180.000000
D39=180.000000
D40=0.000000
D41=0.000000
D42=0.000000
D43=180.000000
D44=180.000000
D45=0.000000
D46=0.000000
D47=0.000000

D48=180.000000
D49=180.000000
D50=180.000000
D51=180.000000
D52=180.000000
D53=180.000000
D54=180.000000
D55=180.000000
D56=180.000000
D57=180.000000
D58=180.000000

Appendix B

Appendix-B

B.1 Sparse Grids Applied to Bayesian Inference

Consider the model for heat transport in the shallow subsurface given by

$$u(t, z; \theta) = Ce^{-\alpha z} \sin\left(\frac{\pi}{12}t - \alpha z + \omega\right) + \hat{C} \tag{B.1}$$

with parameters $\theta = [C, \alpha, \omega, \hat{C}]$, depth z, and time t. C is the amplitude, α is the damping parameter, ω is the phase shift, and $\hat{C}$ is the average temperature, assuming the temperature u approaches $\hat{C}$ as $z \to \infty$.

Data, which we will denote by Y, was collected at three locations in North America with different Köpen climate classifications: tropical (Dataset A), desert (Dataset B), and undifferentiated highlands (Dataset H). Temperatures were collected over 5-minute intervals for a period of 40 days at depths of $z = 1, 5, 10, 15, 20, 25$, and 30 cm. With this data we calibrate our model parameters θ using the Bayesian framework.

Bayes' Theorem is given by

$$\pi(\theta|Y) = \frac{f(Y|\theta)\pi_0(\theta)}{\int_\Gamma f(Y|\theta)\pi_0(\theta)d\theta} \tag{B.2}$$

where $p(\theta|Y)$ is the posterior probability density of θ given our data Y, $f(Y|\theta)$ is the sampling density of the data Y given our parameters θ, and $\pi_0(\theta)$ is the prior density. The prior density can be used to incorporate prior knowledge about the parameters into analysis. For this work we assume an uninformative (constant) prior density.

Assuming identically and independently (iid) normally distributed errors with mean 0 and

weighted variance $\frac{\sigma^2}{2_i^2}$, the sampling density is given by

$$f(Y|\theta) = \frac{1}{(\sqrt{2\pi\sigma^2})^{7n_t}} \prod_{i=1}^{7} \left[w_i^{n_t} \exp\left(\frac{-1}{2\sigma^2} SS\right)\right] \tag{B.3}$$

where n_t is the number of data points in time and the weights w_i depend on depth z. The sum of squares SS is given by

$$SS = \sum_{j=1}^{n_t} [w_i(y_{ij} - f(t_i, z_j, \theta))]^2. \tag{B.4}$$

While the variance σ^2 could be considered as a fifth parameter in Bayesian analysis, we fix the variances $\sigma_A^2 = 0.01$, $\sigma_B^2 = 0.0567$, and $\sigma_H^2 = 0.3299$ for the respective datasets.

To evaluate the posterior density $\pi(\theta|Y)$ we approximate the likelihood function and its integral with sparse grids using the MATLAB Sparse Grid Interpolation Toolbox [121, 124, 122]. The bounds for the interpolation domain Γ_i for θ_i was taken from the minimum and maximum parameter values from Markov Chain Monte-Carlo (MCMC) chains computed with the Delayed Rejection Adaptive Metropolis (DRAM) algorithm. The chains were run for 7500 iterations. For Dataset A, for example, the interpolation domain is

$$\begin{aligned} \Gamma &= \Gamma_C \times \Gamma_\alpha \times \Gamma_\omega \times \Gamma_{\hat{C}} \\ &= [2.368, 2.666] \times [0.08514, 0.09126] \times [5.845, 5.960] \times [23.11, 23.14]. \end{aligned} \tag{B.5}$$

We use the `NoBoundary` sparse grid option that employs the maximum-norm-based sparse grid without points on the boundary with a grid depth of 10 to obtain the approximation

$$\hat{\pi}(\theta|Y) \approx \pi(\theta|Y) \tag{B.6}$$

that requires 471,041 evaluations of the likelihood $f(Y|\theta)$. The sharp peak of the likelihood function necessitates the high sparse grid depth, as sparse grids with lower depth lead to significant error due to magnitude of the likelihood's derivatives in the neighborhood of the peak.

Using our approximation of the likelihood, the marginal density for θ_i is

$$p_i(\theta_i) = \int_{\Gamma'} \hat{\pi}(\theta|Y) d\theta' \tag{B.7}$$

where Γ' and $d\theta'$ include all Γ_j and $d\theta_j$ for $j \neq i$. For a given value of θ_i, the marginal distribution is found by calculating the integral in Equation B.7 with sparse grids using the MATLAB Sparse Grid Interpolation Toolbox [121]. Here we also use the `NoBoundary` sparse grid option but with a depth of 9. We note that this integral must computed each time we evaluate the marginal $p_i(\theta_i)$, and the high depth of the sparse grids makes this an expensive process.

In Figure B.1, we compare the marginal distributions for each parameter for Dataset A computed from Equation B.7 with the marginal distributions computed from DRAM results. The same comparisons for Datasets B and H are shown in Figures B.2 and B.3. We expect that if we increase the depth of our sparse grids the DRAM and sparse grid marginal distributions would agree even more, but the computational expense of higher-level sparse grids is inhibitive.

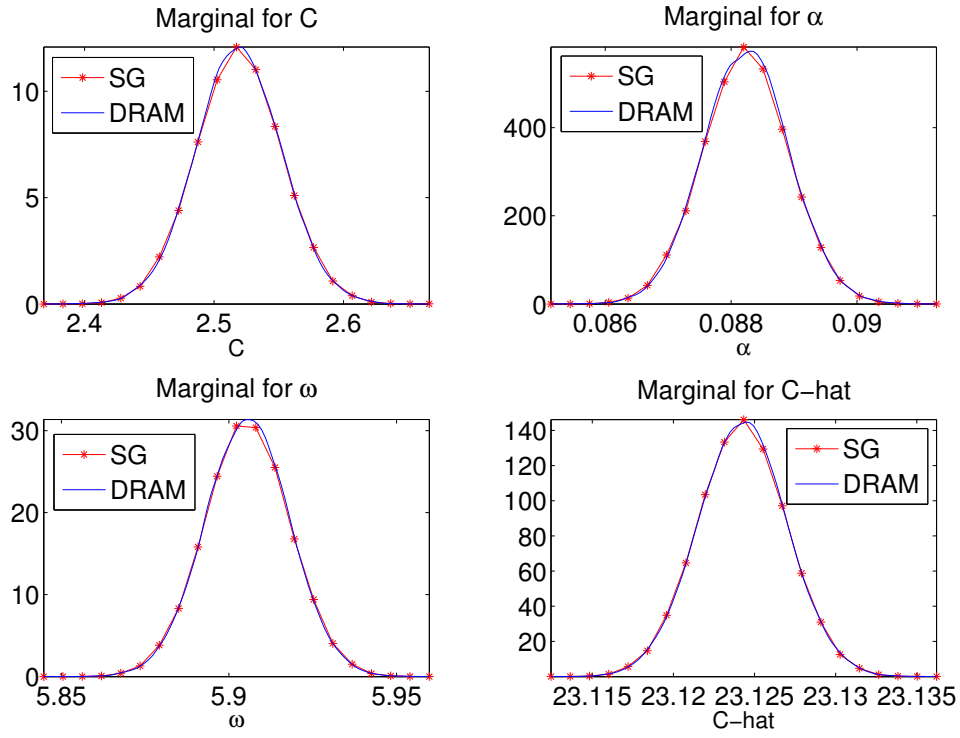


Figure B.1: Comparison for Dataset A: each marginal distribution calculated with the MATLAB Sparse Grid Interpolation Toolbox (SG) and the marginal distribution calculated from DRAM results.

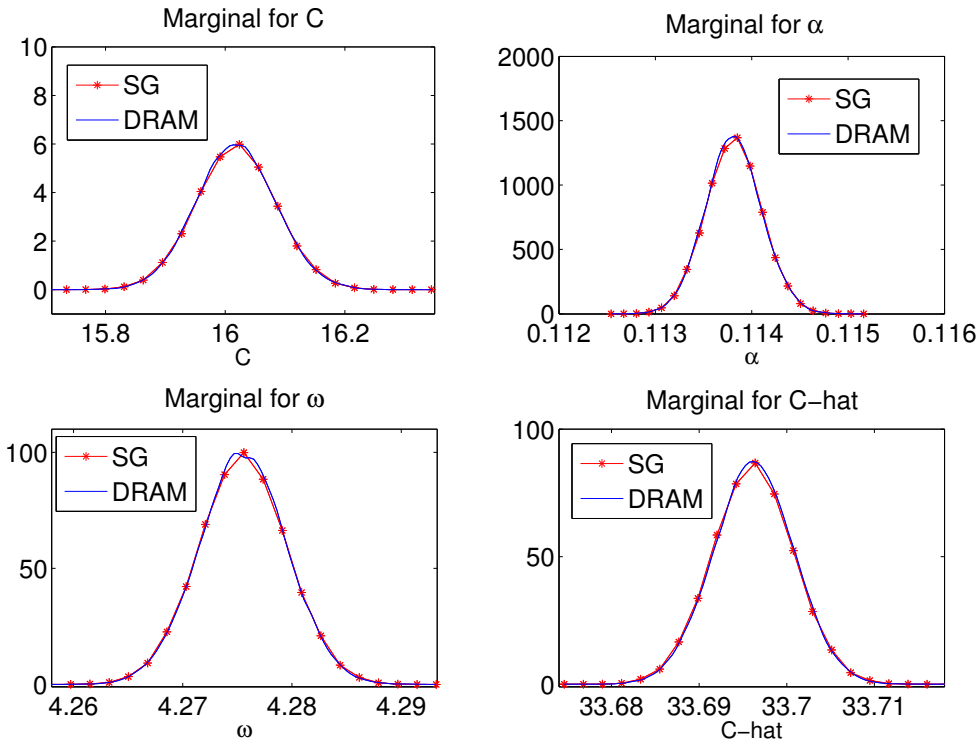


Figure B.2: Comparison for Dataset B: each marginal distribution calculated with the MATLAB Sparse Grid Interpolation Toolbox (SG) and the marginal distribution calculated from DRAM results.

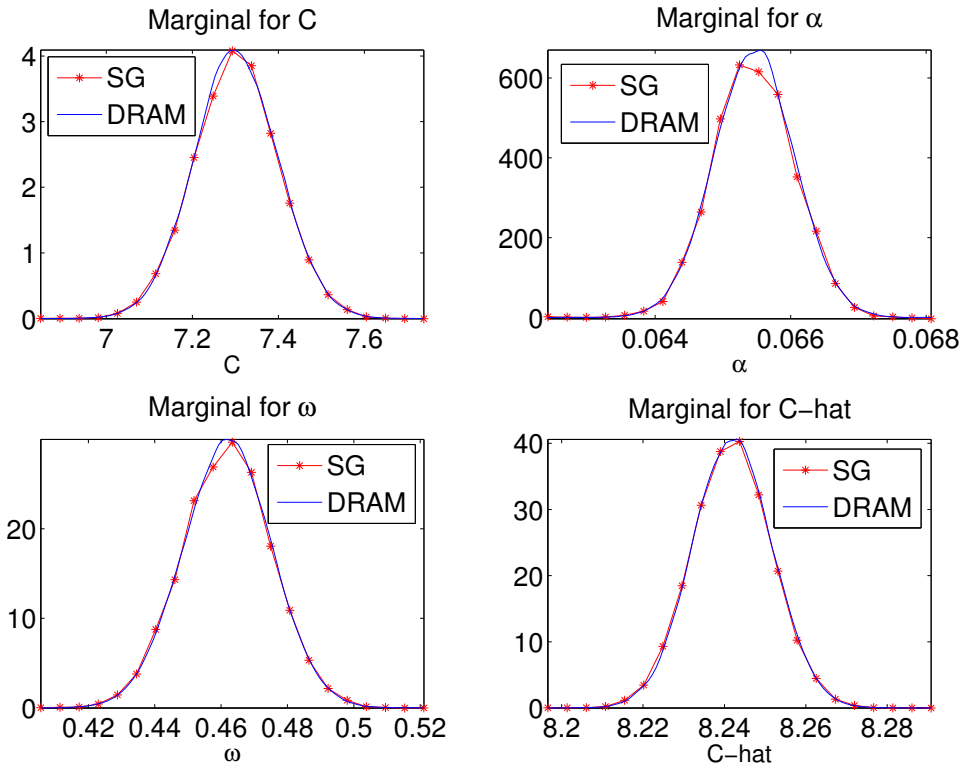


Figure B.3: Comparison for Dataset H: each marginal distribution calculated with the MATLAB Sparse Grid Interpolation Toolbox (SG) and the marginal distribution calculated from DRAM results.