2013

Do We Need to Analyze Spectra by Hand?

Adam M. Terwilliger
Grand Valley State University

Richard L. Lord
Grand Valley State University

Do We Need to Analyze Spectra by Hand?

Adam M. Terwilliger and Richard L. Lord

Department of Chemistry, Grand Valley State University, Allendale, MI 49401

Abstract

Computational chemistry uses computer science to explore structures and energies of chemical species. A typical computational chemistry output file contains tens or hundreds of thousands of text lines. Automation makes the analysis of these large data sets increasingly more efficient. In turn, we constructed computer programs using Python that allow us to focus our time upon the chemical interpretation of these results. We used these efficient analyses to study a vanadium-oxo species synthesized by our collaborators. Our calculations answer many questions about the redox states in these compounds, though they predict that the experimental crystal structure may not reveal all species present. Subsequently, in an effort to distinguish which species are present, we simulated the absorption spectra of the lowest energy structures. These spectra motivated a spectral analysis program written in Mathematica, with which we gain greater insight into why these compounds absorb light differently.

Introduction

There are over seven billion people on this planet.[1] Each day the average human speaks approximately 7,000 words.[2] On Earth, the life expectancy rate is 69.91 years.[3] Through a simple extrapolation, that roughly equates to 125,119,674,750,000,000 total words spoken in the lifetime of every individual currently alive today. That's 1.25 quintillion words! To put this into perspective, on a single day IBM generates 2.5 quintillion bytes of data.[4] The magnitude of the digital world is almost unimaginable. Often it seems "big" data being described with such a short adjective does not do it justice. We see data in this world as colossal, prodigious, tremendous, capacious, expansive, and paramount. In other words, society has a problem to face. Many questions can be asked as we look for a solution. What data is important and what data is irrelevant? Where can we focus our time and effort to make the most meaningful impact? How do we organize this data? Where do we even begin?!

Thankfully with the greatest of minds coming together, we are on our way to handling the many dimensions that the ever expanding technological world has to offer in the way of "big" data. One such way is automation.[5] The beauty of automation is that it requires little to no control from the user. Any process that is overly repetitive can be automated. Traditionally, this automation has been focused on physical tasks (car assembly), but there is an increasing demand for automation of data organization.[6] With automation in action, the user can focus his or her attention on more meaningful analyses.

From understanding the human genome[7] to preserving national security[8] to the large hadron collider[9]; big data is everywhere. As research is vast in a multitude of fields, it is quite evident that owning control of data within the specific area is important. Computational chemistry is such a field where being able to manage data is crucial, and uses computer science to explore structures and energies of chemical species. Output files typically contain tens or hundreds of thousands of text lines. We hope to make more efficient and meaningful analyses of these structures and energies through the implementation of automation.

Theory

Equation 1.

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

$\hat{H} = Hamiltonian$

$\psi = wavefunction$

E = Energy

One of the major goals in computational chemistry is to gain insight into the structures and energetics of chemical species that cannot be isolated and studied in the laboratory. To do this, we need to be able to calculate the energy of a given species and to find its optimal geometry.[10] To understand geometric optimization, one must first understand the basis behind Schrödinger's equation.[11] In Equation 1, Schrödinger's equation describes how a physical system changes with respect to time involving the Hamiltonian operator, acting on the wavefunction, which can be broken into kinetic and potential energies. Furthermore, one can separate those terms into ones involving the nuclei and the electrons in a chemical system. These components, seen in Equation 2, include (in order) electron kinetic energy, nuclei kinetic energy, electron-electron potential energy, nuclei-electron potential energy, and nuclei-nuclei potential energy.

Equation 2.

$$\hat{H} = -\frac{1}{2}\sum_{a=1}^{n}\nabla_a^2 - \frac{1}{2}\sum_{A=1}^{N}\nabla_A^2 + \frac{1}{2}\sum_a^n\sum_b^n\frac{q_a q_b}{r_{ab}} - \sum_a^n\sum_A^N\frac{Q_A}{r_{aA}} + \frac{1}{2}\sum_A^N\sum_B^N\frac{Q_A Q_B}{r_{AB}}$$

To help visualize these particles, refer to Figure 1, as it gives an out of scale picture of a "charge distribution" (electron cloud) and a "point charge" (nucleus). The mass of a nucleus is far greater than the mass of an electron; therefore, if we assume that the momentum of all particles is approximately equal, then the velocity of the nucleus is far smaller than the velocity of the electron. The assumption that nuclei are

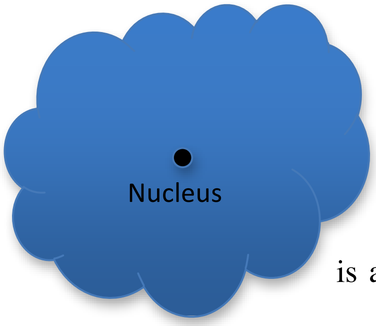

Figure 1. Electron Cloud

point charges (relative to the size of the electron clouds) also allows us to easily evaluate the nucleus-nucleus potential energy. Through this approximation, known as the Born-Oppenheimer Approximation,[12] inferences can be made that simplify the number of energies, and their interdependencies, necessary to solve for in the Hamiltonian operator. The kinetic energy of the nuclei and the nucleus-nucleus potential energy can be solved separate from the energy involving electrons due to the different timescale that these particles move on. The potential energy for an electron interacting with another electron is the most difficult remaining component in the Hamiltonian to solve for. In turn, this challenge provides the basis for the need of computation in solving quantum chemical problems.

At this point, we cannot solve for the wavefunction directly; we look to represent our wavefunction in terms of more tractable functions. There are two main ways to approach this electronic structure problem. In one approach, known as the *ab initio* method, we assume that the total wavefunction can be represented as a linear combination of simpler functions, namely atomic orbitals ($\psi = \sum_i c_i \phi_i$). The problem of solving the wavefunction then becomes one of determining the c_i, which are orbital coefficients.

Equation 3.

$$E_i = \int \psi_i^* \hat{H} \psi_j \, d\tau \rightarrow \int (\textstyle\sum_k c_k^* \phi_k^*) \hat{H} \, (\textstyle\sum_l c_l \phi_l)$$

$$E_i = \sum_k \sum_l c_k^* c_l \, (E_k \delta_{kl}) = \sum_k |c_k|^2 E_k$$

$$\sum_k |c_k|^2 Ei \, - \sum_k |c_k|^2 E_o \; = \; \alpha$$

ψ_i is some guess of our ground state wavefunction ψ_o. We choose ϕ such that they are eigenfunctions of H. As E_i-E_o=α, we can vary C_k such that α goes to zero, where $E_i - E_0 \geq 0$. δ_{kl} is the Kronecher delta. This systematic prescription for minimizing the ground state energy (and wavefunction) by varying the coefficients is known as the variational principle.[13] Instead of solving instantaneous interactions within the Hamiltonian operator between the electrons, we make a mean-field approximation.[14] That is, we will look at the ith electron interacting with an average density of the remaining (n-1) electrons.

$$\int \frac{\rho(r)q_i}{r} dr = V_i\{j\}$$

Where ρ is the density, r is the distance between charges, q is the charge, and V is the mean-field potential mentioned above. This casts each of the Hamiltonian components as one-electron problems. The problem is that we must know the where the electrons are to solve for their locations, so the problem is iterative in nature; we refer to this procedure and the resulting wavefunction and energies as a Self-Consistent Field (hence E (SCF)) method.

We can deconstruct this hierarchy of approximations, known as the Hartree Fock method,[15] into seven main steps. Ultimately, while very complicated, this approach can be summarized by a relative simple flowchart (Figure 2) which has been automated in quantum chemical programs like Gaussian.[16]

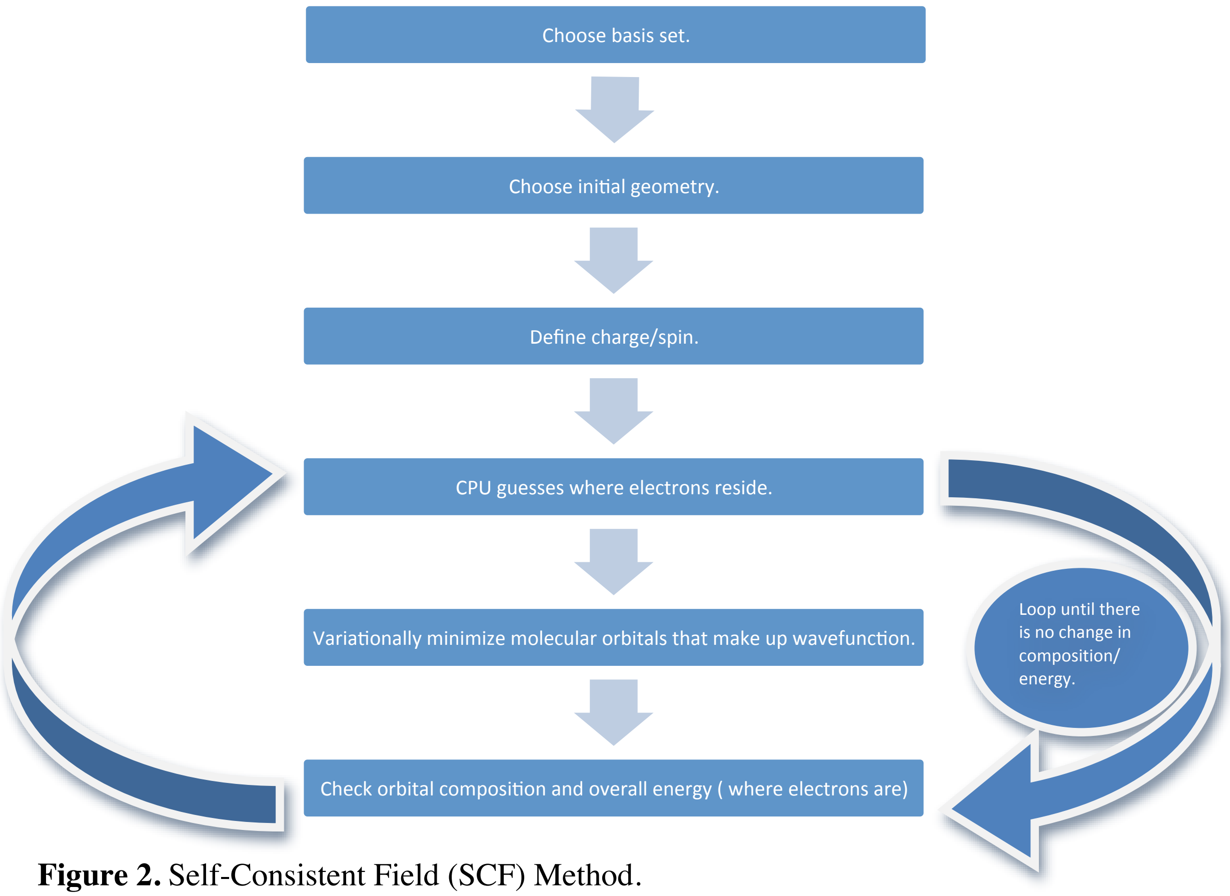
Choose basis set.
Choose initial geometry.
Define charge/spin.
CPU guesses where electrons reside.
Variationally minimize molecular orbitals that make up wavefunction.
Check orbital composition and overall energy (where electrons are)
Loop until there is no change in composition/ energy.

Figure 2. Self-Consistent Field (SCF) Method.

The second approach to the electronic structure problem is density functional theory (DFT),[17] with $F[\rho] = E * \rho$, where the energy is determined for the density and we do not attempt to solve for the complicated wavefunction. With *ab initio* methods, we know H exactly but we do not know Ψ; while, with DFT we know ϱ exactly but we do not know F[]. From a philosophical perspective, with *ab initio* methods, we know what the question to ask for the E is, but we have no idea what the wavefunction looks like; however, with DFT, we know the density to get an E from exactly and can even measure it experimentally, but we do not know the question to that E from the density. The main advantage to using DFT is the relative cost compared to the expensive ab initio approaches. Unfortunately, a big disadvantage to DFT is the inability to systematically improve the energy.

Equation 4.

$$\widehat{H}_i = \frac{-1}{2}\nabla_i^2 - \sum_A \frac{Q_A}{r_{iA}} + \sum_i \sum_j \frac{1}{r_{ij}}$$

Our goal is to represent each part of the Hamilitonian in terms of density. Fortunately, we are able to solve two terms very easily.

Equation 5.

$$\sum_A \frac{Q_A}{r_{iA}} \rightarrow \int \frac{\rho_i(r_1)Q_A}{|r_A - r_1|} dr_1$$

$$\sum_i \sum_j \frac{1}{r_{ij}}(electrostatic) \rightarrow \iint \frac{\rho(r_1)\rho(r_2)}{|r_1 - r_2|} dr_1 dr_2$$

We can use orbitals to approximate the kinetic energy term. The final term we are left to solve for is the exchange and correlation term. This term collects various approximation errors and purely quantum effects and is usually fit to a small data set. In our calculations, we use the specific prescription called B3LYP[18-22] because it has been shown to perform well for a diverse set of chemical problems.

Returning to the idea of the Born Oppenheimer approximation, because nuclei move much slower than electrons, we consider the nuclear coordinates to be parameters in the electronic structure problem. Potential energy surfaces (PES) define the potential energy of a collection of atoms over all possible atomic arrangements. As energy is plotted versus nuclear coordinates we are particularly interested in maxima, minima, and saddle points that correspond to chemical species like reactants, products, intermediates, and transition states. To locate and evaluate these critical points, calculus can be utilized. To optimize the system partial first derivatives with respect to each coordinate are evaluated. A critical point is a point on the PES for which all partial first derivatives are equal to zero (Equation 6). In turn, the second derivative with respect to each of the nuclear coordinates is taken. A positive second derivative (positive curvature) for all dimensions (or normal modes) in the system represents the minimum for which we set out to find. A negative second derivative gives rise to a maximum in that dimension, which is reported as an imaginary frequency by our quantum chemical program. We seek to rid our system of imaginary frequencies; in turn, we will see later one of our developed programs looks to solve this issue.

Equation 6.

$$\frac{\delta E}{\delta x_A} = \frac{\delta E}{\delta y_A} = \frac{\delta E}{\delta z_A} = \frac{\delta E}{\delta x_B} = \frac{\delta E}{\delta y_B} = \frac{\delta E}{\delta z_B} = 0$$

Programming

How can we make human analyses of our data sets most efficient? As noted in the Theory section, we look to find the minimized energies of our system. Additionally, linked to these energies as previously seen, are coordinates (geometries) and frequencies. Although this may seem fairly straightforward at first glance, with all cycles the total geometries and frequencies can run from 50-100 for each optimization. With each output file, the revelant data for analysis is found in only a handful of lines. However, the human must parse through thousands of lines in these log files to reach the very few revelant lines. We chose to utilize the language of Python[23] as it allowed for quick understanding due to its relatively simple syntax. In turn, we save time and effort that can be focused on more meaninful analyses. Automation serves as the key with our programs.

Thermo.py: Natures tends to thermodynamic equilibrium, and a large part of chemistry is understanding these equilibria. Thermo.py extracts and organizes thermodynamic data from a quantum chemical simulation.

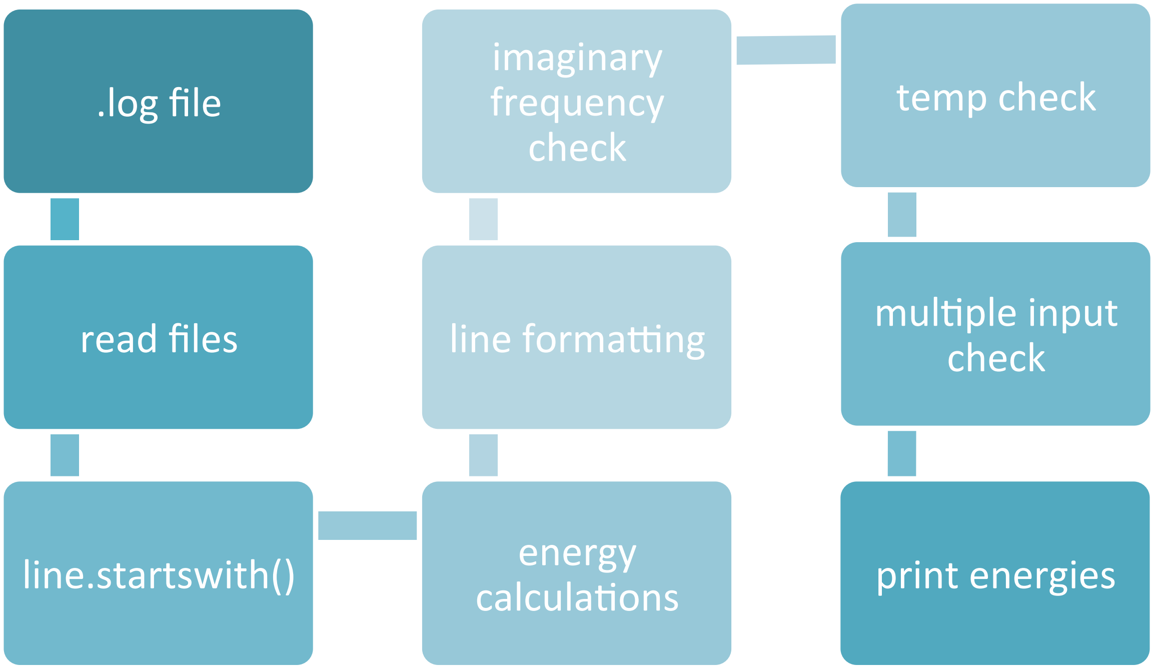

Figure 3. Flow Chart of thermo.py

Interpretation of Flow Chart:

.log files are entered in command line by user. They are the converted using the open() function, in coordination with the sys.argv command. We then find the thermodynamic energies in the output file by using conditional logic with the line.startswith() command. We then calculate the multiple energies that are not calculated directly by using the formula S = (H – G)/T, where S is entropy, H is enthalpy, G is free energy, and T is temperature.[11] We then use string formatting commands, which gives the print statements a fixed position and makes for easy importing into a program like Excel. Additional features were added to help the user identify common errors in the optimization. Our program checks for imaginary frequencies by utilizing conditional logic for which we can edit the filename variable to output with an asterisk. We also check for the correct temperature by creating a global temperature variable and utilizing the same line.startswith() command to change this global variable only if the temperature is not room temperature. This allows for the filename to be conditional changed with an "+". Finally, our program checks for multiple files through running a while loop that checks the length of the sys.argv command until all of the files are read. For more information, including that of a user tutorial and future improvements, please see Appendix 3 and 6 (source code).

Geom.py: Computed geometries can be compared to experimental X-ray structures, or can give us insight into species that cannot be isolated in the laboratory. A typical geometry optimization output file can have upwards of 50 geometries. Geom.py gives the user flexibility to quickly analyze the geometries of interest. See Appendix 4 and 7 for more information.

Freq.py: Frequencies are related to experimental infrared spectra and they tell us about the shape of the potential energy surface for an optimized structure. Freq.py extracts all frequencies (in cm^{-1}). See Appendix 8 for the source code.

Imag.py (See Figure 4): If imaginary frequencies are found then the structure is not minimized. Imag.py gets the optimized geometry and checks for imaginary frequencies. The geometry is then displaced along the normal mode for that frequency, which returns a new geometry that can be re-optimized. This program is an example of how a new program can incorporate the previously written modules (geom.py and freq.py). See Appendix 5 and 9 for more information.

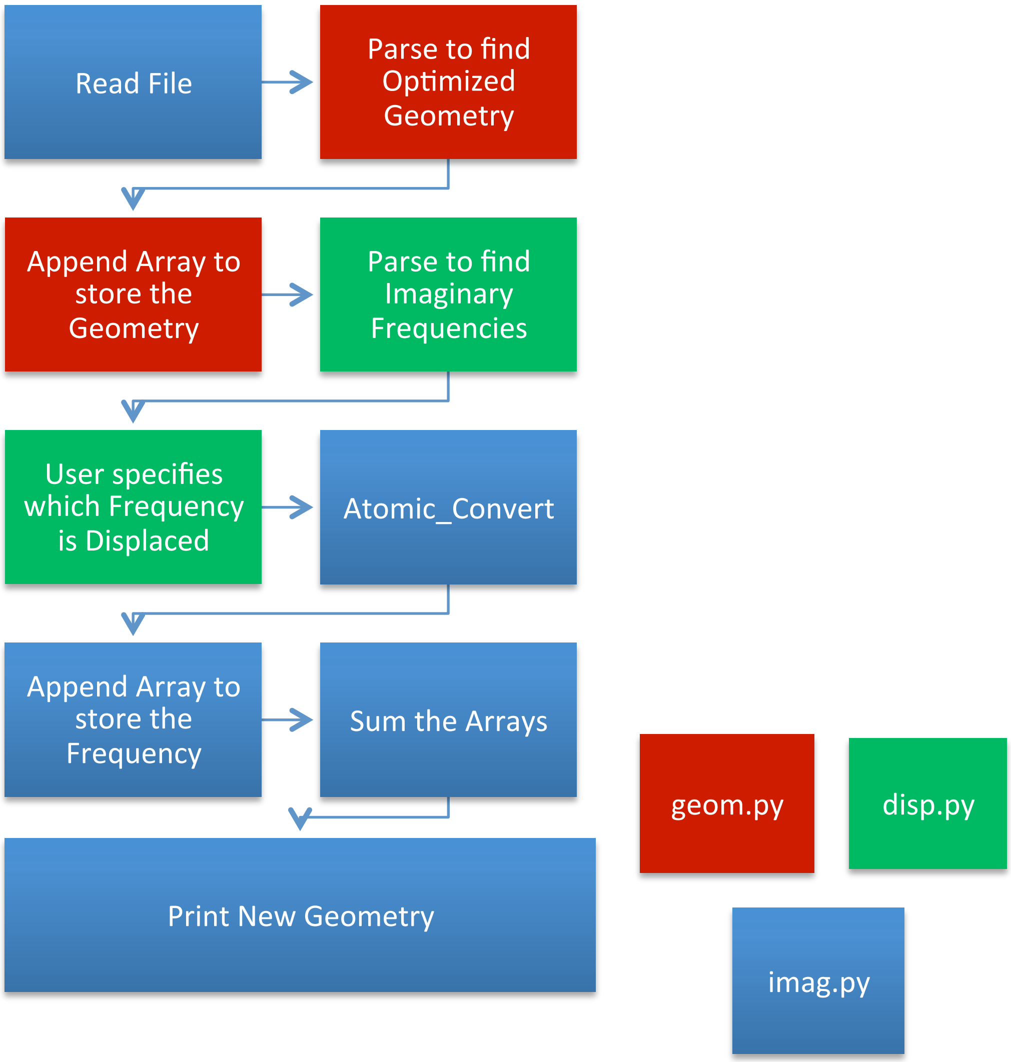
Read File
Parse to find Optimized Geometry
Append Array to store the Geometry
Parse to find Imaginary Frequencies
User specifies which Frequency is Displaced
Atomic_Convert
Append Array to store the Frequency
Sum the Arrays
Print New Geometry
geom.py
disp.py
imag.py

Figure 4. Flow Chart of Imag.py

Interpretation of Flow Chart:

Read file: A filename that is inputted on the command line by the user is called using the system module.

Parse to Find Optimized Geometry: Regular expressions are utilized to compile a pattern which the program parses each line of the input file to find matches.

Append Array to store the Geometry: The module numpy allows for multidimensional arrays to be called.

Parse to find Imaginary Frequencies: Again, regular expressions are utilized to compile and patterns just as seen in "Parse to Find Optimized Geometry" is used. We are now looking for the imaginary frequency displacements.

User specifies which Frequency is Displaced: The user answers which geometry, or geometries, he or she wants. Then the user chooses between atomic symbols, names, or numbers.

Atomic_Convert: An external program is called that formats the atom information according to the user input.

Append Array to store the Frequency: Similar to storing the geometries, numpy is utilized to manipulate the multidimensional arrays. However, the inner calculations have differences that are specified in Appendix 5.

Sum Arrays: Geometry and frequency arrays are consolidated with a sigma value that determines the magnitude of the displacement.

Print New Geometry: Prints the data in a convenient format for the user.

Chemistry

With our programs in hand, we wanted to apply them to a real life system. One of our collaborators, Professor Ken Caulton at Indiana University, had recently identified an unusual and intriguing vanadium species. His lab works on metal-containing molecules that are capable of making important molecules like carbon dioxide become reactive.[24] Unfortunately, their molecules are often difficult to isolate and study in detail, so they looked to us for computational insight into the electronic structure of their newly acquired vanadium species. To better understand the electronic and geometric structure of $[(Hbtzp)VCl_2O]$ that was identified by crystallography, we addressed the following questions with density functional theory: (i) what are the oxidation states of the metal and ligands in the lowest energy spin state of $[(btzp)VCl_2O]^0$?, (ii) which N atom does the H atom prefer to bind to in this vanadium complex?, and (iii) how does the electron distribution change when the H atom binds to btzp?

First, what are the redox/spin states of the ligand and metal without hydrogen attached to btzp? One can envision oxidation states for vanadium ranging from V^{III} to V^{V}, depending on the oxidation state of the oxo ligand and the btzp ligand. V^{IV} is expected if we assume that btzp is neutral and the oxo is 2–. Two spin states were explored for this species: doublet and quartet. The lowest energy optimized structure (doublet) can be seen in Figure 5 and a table of important bond lengths is presented in Table 1. The molecule is approximately C_{2v} symmetric so bond lengths are only reported for one tetrazine arm.

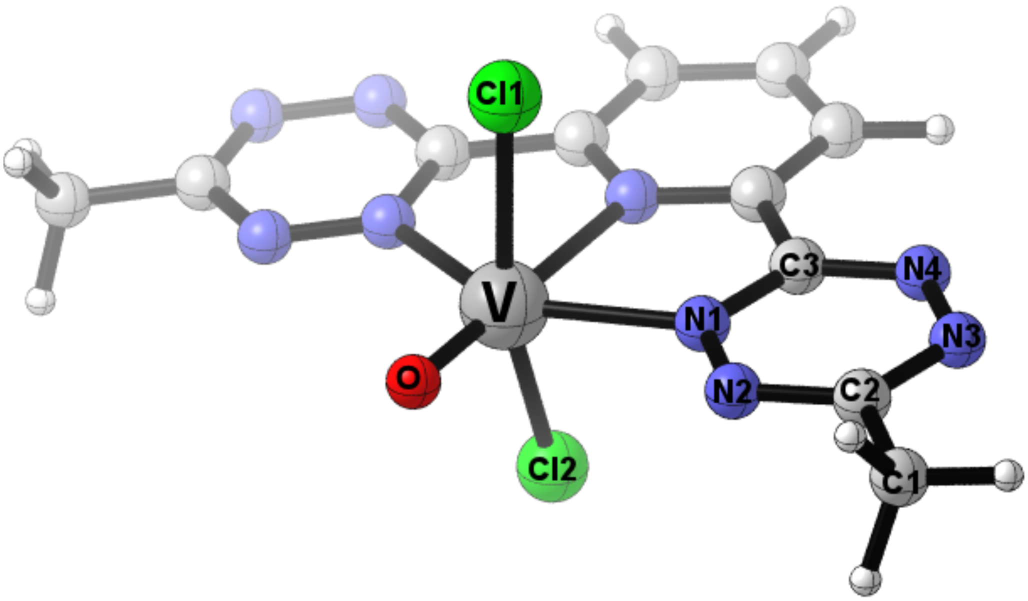

Figure 5. Structure of doublet $[(btzp)VCl_2O]^0$ with atomic labels for important atoms.

Table 1. Bond Lengths (Å) and Energies (kcal/mol) for the two spin states of $[(btzp)VCl_2O]^0$.

	doublet (S = 1/2)	quartet (S = 3/2)
N1-N2	1.307	1.321
N1-C3	1.361	1.356
N2-C2	1.349	1.339
C2-N3	1.346	1.356
N3-N4	1.321	1.320
C3-N4	1.333	1.335
Relative Free Energy	0.00	+42.49

Based upon the energy difference of 42.5 kcal/mol, we conclude that the neutral species has a doublet spin state. We used B3LYP for our calculations, a functional that is known to overstabilize high-spin states, so the large energy difference is likely a lower bound. We look to spin density and molecular orbital diagrams to understand the electron distributions in this lowest energy spin state. Those diagrams are shown in Figure 6.

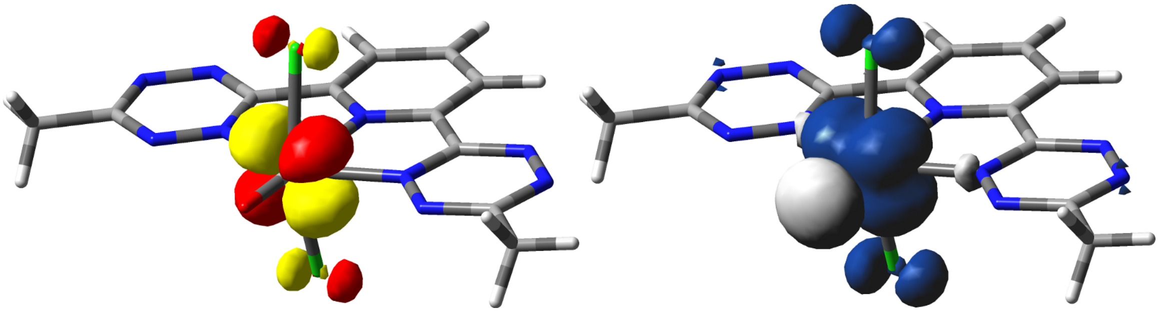

Figure 6. Isosurface plots of the singly occupied molecular orbital (0.05 au, left) and spin density (0.002 au, right).

Our spin density and SOMO show that the unpaired electron density is concentrated around the metal center with no concentration on the btzp ligand. This finding of one unpaired electron at the metal is consistent with our initial speculation that this species is V^{IV}. The spin density plot shows a slight excess of β spin at the oxygen; however, the corresponding orbital analysis (used to generate the SOMO in Figure 6) did not identify an unpaired electron on O. For an explanation of this spin polarization see SI-3 and Appendix 1.

To locate the thermodynamically preferred position of hydrogen atom binding to the Hbtzp ligand in [(Hbtzp)VCl_2O] we considered each of the three tetrazine N atoms (N2, N3, N4; see Figure 5 for atom labels) not bound to the vanadium center. Adding the hydrogen atom (S = ½) to the doublet species could result in either a singlet or triplet species, depending on whether the hydrogen's electron prefers to orient in the same or opposite direction to that of the V ion. Thus, we ran optimizations on a total of six species with two spin states for each N binding location. To simplify the discussions below, we will refer to these three isomers as **2**, **3**, and **4** (referring to H bound to N2, N3, and N4, respectively) with a subscript of either S or T to indicate singlet or triplet, respectively. The results are summarized in Table 2.

Table 2. Relative free energies in kcal/mol for various isomers in [(Hbtzp)VCl$_2$O].

Species	CalcID	Spin State	H Position	Relative Free Energy
$\mathbf{2}_S$	003	Singlet	N2	–1.54
$\mathbf{2}_T$	004	Triplet	N2	+0.38
$\mathbf{3}_S$	005a	Singlet	N3	0.00
$\mathbf{3}_T$	006	Triplet	N3	+1.36
$\mathbf{4}_S$	007	Singlet	N4	+7.47
$\mathbf{4}_T$	008	Triplet	N4	+7.21

2 and **3** were found to be most stable, with **4** higher in energy by ~7 kcal/mol. Thus, we focus our attention on **2** and **3**. For both isomers, the singlet is favored over the triplet by 1-2 kcal/mol, with **2** favored over **3**, but by an energy difference that is within the expected error of our methodology. This energetic ordering may also be influenced by the fact that our model optimizes the structures without solvation effects or intermolecular interactions (that are present in the solid state). We suspected there is artificial stabilization of the N2–H site due to its proximity to the anionic oxo ligand. See Appendix 2 for view showing H-bonding between N2 hydrogen and oxygen). We next compared the intra-ring bond lengths from $\mathbf{2}_S$, $\mathbf{3}_S$, and the experimental X-ray structure. We found the N–N and N–C bond lengths in the hydrogenated tetrazine arm of the btzp ligand to be most diagnostic and the comparison of these bond lengths is seen in Figure 7. A similar comparison of $\mathbf{2}_S$, $\mathbf{3}_S$, and $\mathbf{4}_T$ may be seen in Figure SI-1.

The N2–C2 X-ray bond length of 1.309 Å is closest to that of the shorter 1.305 Å in $\mathbf{3}_S$, rather than the longer 1.360 Å in $\mathbf{2}_S$. Similarly, the C2–N3 X-ray bond length of 1.349 Å is closest to that of the longer 1.365 Å in $\mathbf{3}_S$, rather than the shorter 1.312 Å in $\mathbf{2}_S$. $\mathbf{4}_T$ is the furthest from agreeing with experiment. Therefore, the $\mathbf{3}_S$ structure gives the best agreement with experiment. Given the small energy difference between $\mathbf{3}_S$ and $\mathbf{3}_T$, the singlet and triplet states, we wanted to use a structural comparison to further test our DFT-energy-based spin state assignment.

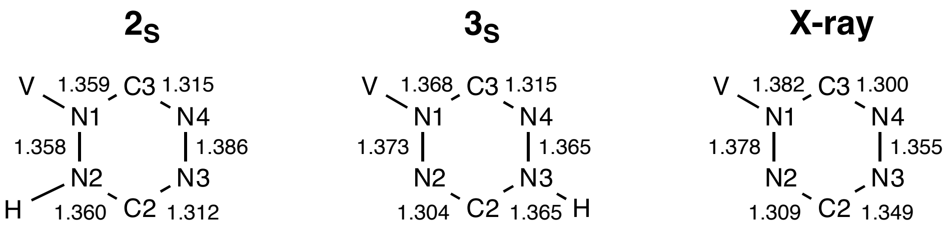

Figure 7. Comparison of [(Hbtzp)VCl$_2$O] bond lengths (Å) for the two lowest energy singlets with experiment.

As shown in Figure 7, the N2–C2 and C2–N3 bond lengths are most conclusive. The N2–C2 X-ray bond length of 1.309 Å is closest to that of the shorter 1.305 Å in **3**$_S$, rather than the longer 1.360 Å in **2**$_S$. Similarly, the C2–N3 X-ray bond length of 1.349 Å is closest to that of the longer 1.365 Å in **3**$_S$, rather than the shorter 1.312 Å in **2**$_S$. **4**$_T$ is the furthest from agreeing with experiment. Through this analysis, we unearthed that the **3**$_S$ structure gives the best agreement with experiment. Given the small energy difference between **3**$_S$ and **3**$_T$, the singlet and triplet states, we wanted to use a structural comparison to further test our DFT-energy-based spin state assignment. Figure 8 shows that the structures of *both* spin states agree well with experiment. The largest difference is in the N1–C3 and C3–N4 bond lengths where **3**$_S$ matches better with experiment by ~0.01 Å in both bonds. A more complete comparison of bond lengths between **3**$_S$ and the X-ray structure may be found in Figure SI-2 and Table SI-1.

Table 3. V–N1 bond lengths (Å) for calculated and X-ray species.

Species	V–N1
2$_S$	2.048
2$_T$	2.159
3$_S$	2.070
3$_T$	2.156
4$_S$	2.139
4$_T$	2.155
X-ray	1.9743(8)

Finally, we sought to use the V–N1 distance to discriminate among NH regioisomers. Table 3 shows the V–N1 comparison for all six calculated isomers. Although all calculated V–N1 bond lengths are longer than experiment, the short V–N1 distance in **3**$_S$ agrees well with experiment.

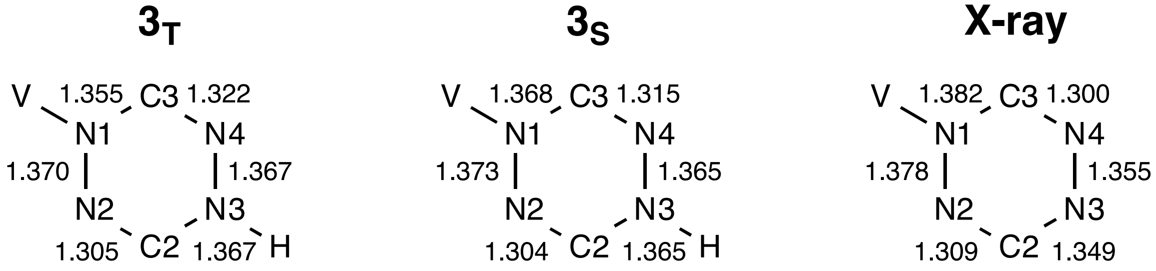

Figure 8. Comparison of bond lengths (Å) for the singlet and triplet spin states to the X-ray data.

The final question addressed was determining the redox/spin states of the ligand and metal within the isomer **3**$_S$ that is most relevant to the X-ray structure. The singlet was found to have an open-shell wavefunction: the restricted calculation with α and β confined to identical spatial orbitals was found to have a wavefunction instability and to be higher in energy than an "unrestricted" wavefunction where different spatial orbitals are allowed for the α and β electrons. We evaluated spin densities and corresponding orbitals (Figure 9) of the open shell singlet wavefunction to help with the redox/spin state assignments. This revealed spin density in the singlet to be on both the metal center and the ligand, but with opposite spins: spin density at vanadium is β (white) while the ligand is α (blue). It is also noteworthy that there is α spin density at oxygen. However, a Mulliken analysis shows this spin to be only ~0.2 and none of the singly occupied orbitals (*vide infra*) are based on the oxo ligand. We then conclude that this spin density is due to spin polarization rather than oxyl character.

The finding of spin density at only one tetrazine arm suggests that the added single hydrogen in this product injects an electron into the π* orbital of tetrazine upon binding to the ligand. For further clarification, we calculated the corresponding orbitals to determine which orbitals are singly occupied in this species. These results (Figure 9, bottom) show a β spin orbital on the metal center and an α spin orbital on the ligand with a small overlap of 0.36. Small overlap means that these two orbitals are spatially distinct, as is visually evident from Figure 9. It is also clear from the SOMO of the ligand that this is a π^*_{NN} orbital. Because there is one vanadium *d*-orbital singly occupied, this complex is best described as a V^{IV} center antiferromagnetically coupled to a tetrazinyl radical. Contrary to our working hypothesis at the end of the crystallographic work, the Hbtzp ligand is an *uncharged* radical in this species. Note that the metal SOMO is also of π symmetry with respect to the pincer plane, yet this does *not* mix significantly with the ligand SOMO ($S_{\alpha\beta}$ = 0.36); spatial separation of opposite spins is found to give a more stable electronic structure. It is worth noting that the analogous triplet species, $\mathbf{3_T}$, has essentially the same metal and ligand redox states, but with the two electrons ferromagnetically coupled.

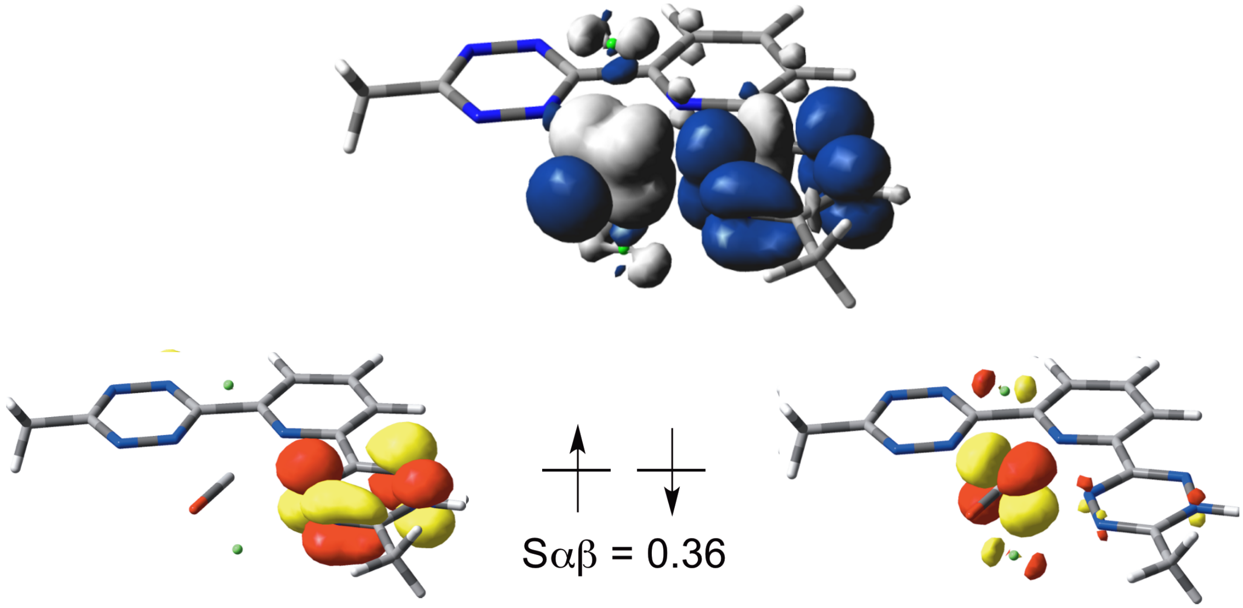

Figure 9. Isosurface plots for the spin density (top, 0.002 au) and the singly occupied corresponding orbitals (bottom, 0.05 au) of $\mathbf{3}_\mathrm{S}$. The overlap between the α (left) and β (right) orbitals is 0.36.

Mini-Chemistry Conclusion

In addition, we found a low energy isomer where the hydrogen transferred from the btzp ligand to the oxo species to give us a hydroxide. As such, our calculations suggested three comparably low isomers for this species: $\mathbf{2}_\mathrm{S}$, $\mathbf{3}_\mathrm{S}$, and $\mathbf{OH}_\mathrm{T}$. In turn, we sought to gain a fuller understand of which species is truly present in solution. One convenient way to identify species in solution is with spectroscopy. After testing various spectroscopies, we found differences in the absorption spectra of the three low energy compounds, suggesting they are colored differently, as seen in Figure 10. This prompted us to create a program that could be utilized for similar problems in the future.

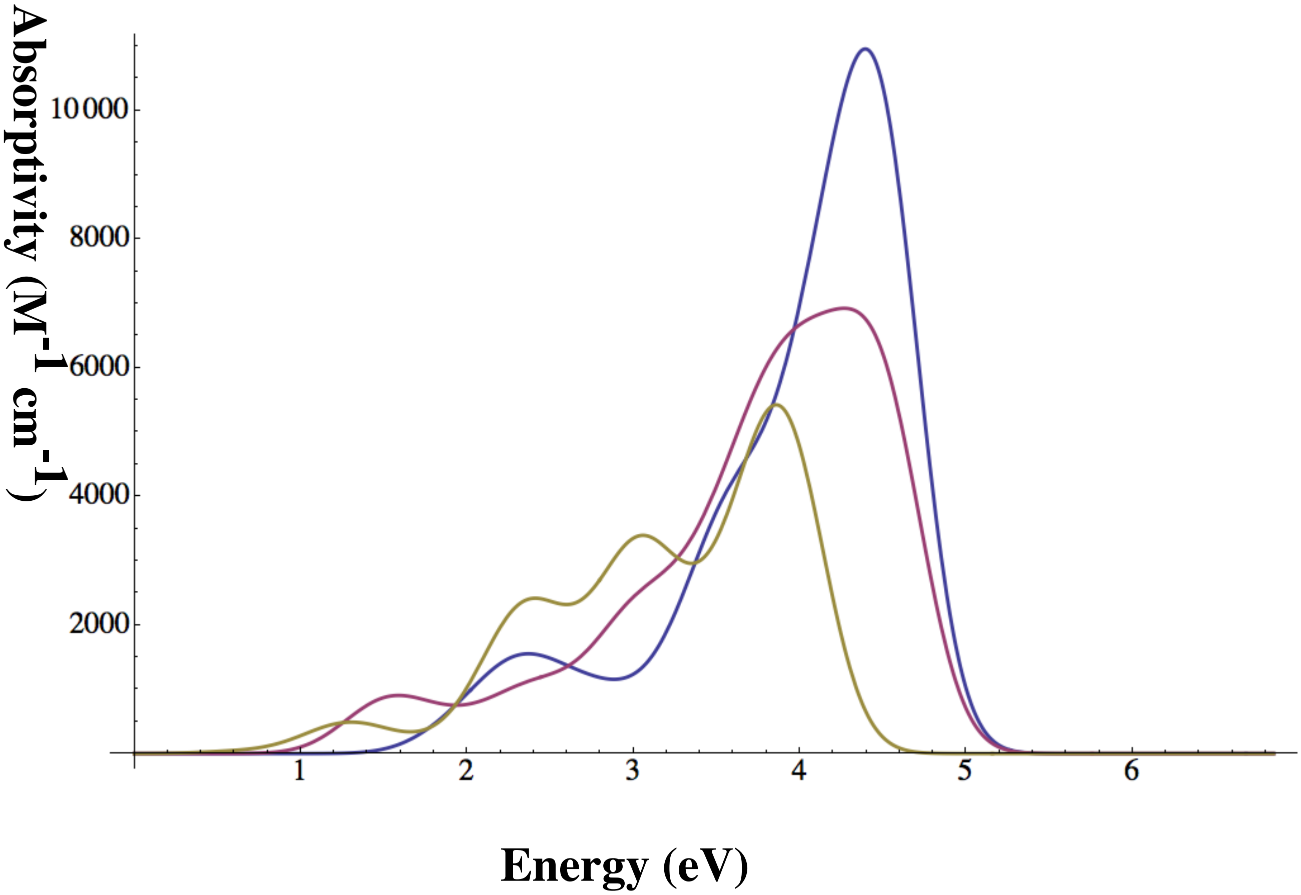

Figure 10. Simulated UV-Vis spectra for lowest energy isomers. The yellow, blue, and red lines correspond to **OH**$_{T}$, **2**$_{S}$, and **3**$_{S}$, respectively.

Future Work

Our ongoing efforts are twofold: (i) develop a Mathematica[25] notebook that interfaces with Gaussian to decide which excited states are significant, and (ii) automate interpretation for the nature of the excited states. Significant progress towards this goal has already been accomplished and is detailed below.

Python – UV-Vis

Reads in log file. Parses to find energies and oscillator strengths of all excited states. Stores information in two distinct arrays. Each array is then written to a .txt file, which is later read in by our Mathematica program. For further details, see the source code in Appendix 9 and 10.

Mathematica – UV-Vis

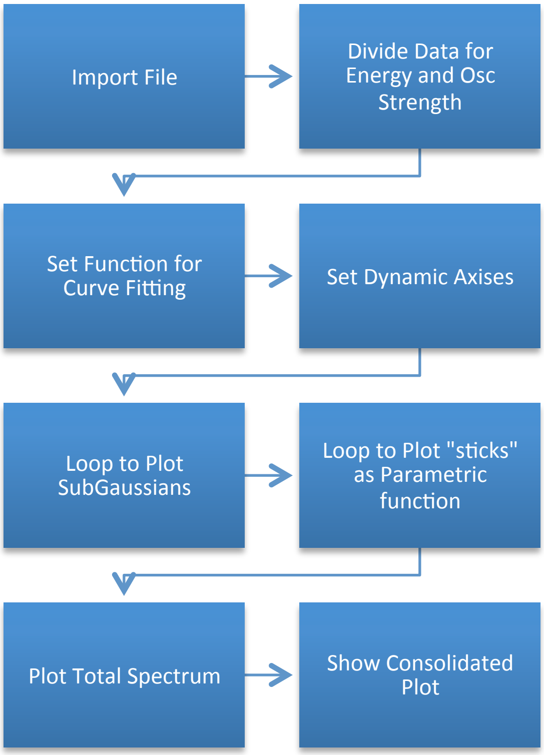
Import File
Divide Data for Energy and Osc Strength
Set Function for Curve Fitting
Set Dynamic Axises
Loop to Plot SubGaussians
Loop to Plot "sticks" as Parametric function
Plot Total Spectrum
Show Consolidated Plot

Figure 11. Flow Chart of Mathematica Program

Interpretation of Flow Chart:

Import file: Set directory and call for user input of the .txt file for which the previous program created.

Divide Data for Energy and Osc Strength: Our data is “flattened” and stores in two tables, one for oscillator strength and one for energy.

Set Function for Curve Fitting: We set function to use our table values in our Gaussian Function. See Appendix 10 for further details on this function.

Set Dynamic Axises: Dynamic InputFields are set with default values and allow for the user to specify the dimensions of the plots.

Loop to Plot SubGaussians: Dynamic Plotting is used for which we call our function to plot our table holding our SubGaussians.

Loop to Plot "sticks" as Parametric function: We utilize ParametricPlot to plot our “sticks” that correspond to our maximum oscillator strength at each respective peak.

Plot Total Spectrum: Our total spectrum is calculated by taking the sum of our SubGaussians Function. This new function is then plotted in the same manner as our SubGaussians were plotted.

Show Consolidated Plot: An image is created that contains the SubGaussians, “sticks”, and total spectrum. This image can be manipulated for size and resolution, and can be saved and stored.

Examples of the output our program produces are found in Figures 10, 12, and 13. The spectrum of butadiene is primarily due to four important excited states (green sticks), as shown, out of the 25 excited states that were modeled. Many excited states make negligible contributions to the spectrum, as seen in Figure 12. Our program allows the user to quickly and easily focus on these 3-4 states that are important.

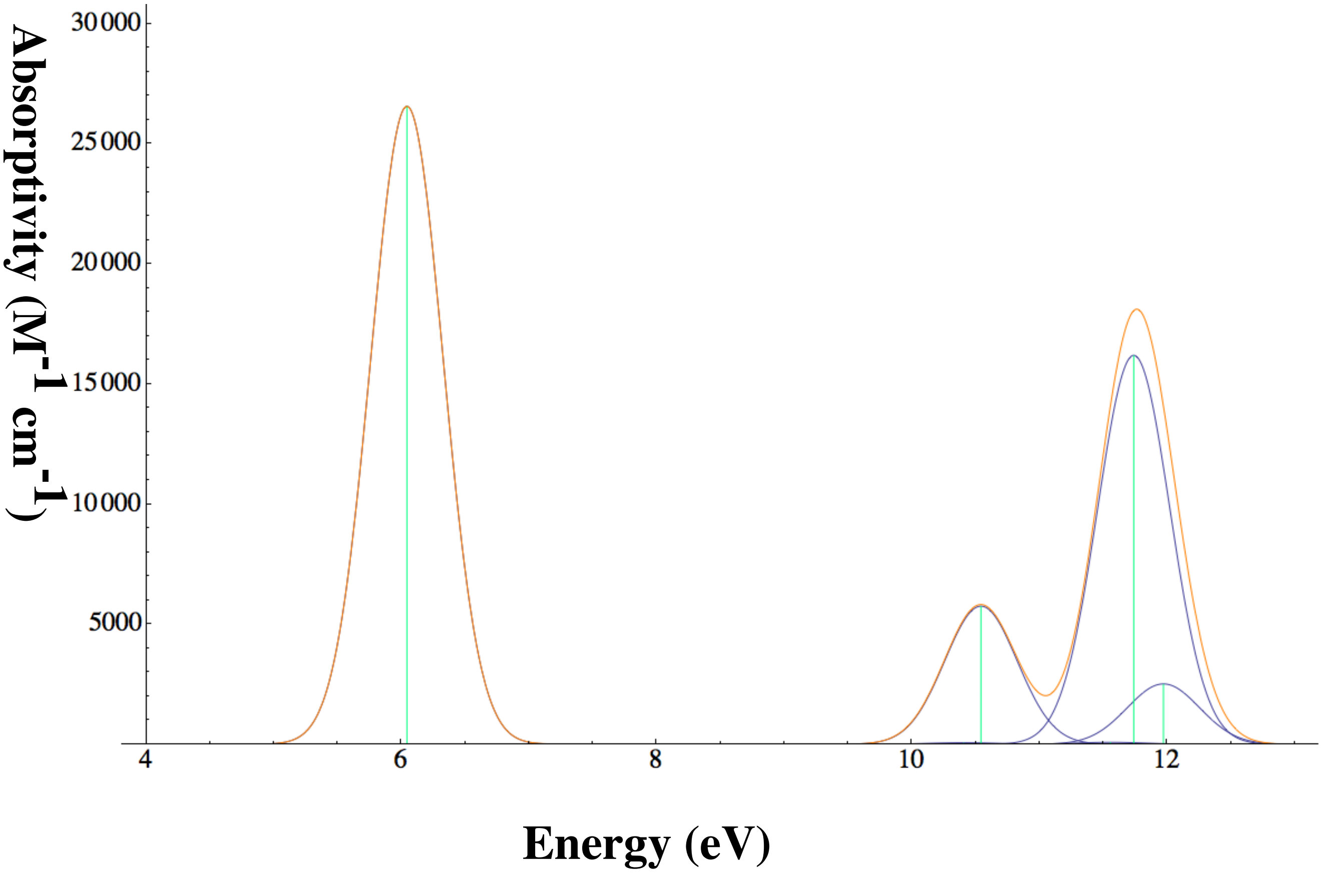

Figure 12. Example UV-Vis for simple system.

For our vanadium-oxo complexes, many excited states make significant contributions, which can make the analysis complicated and time-consuming, as seen in Figure 13. Future work will include post-calculation analysis of the nature of each excited state so that the user can spend more time analyzing the chemical meaning. We have two approaches to this problem that we are pursuing: natural transition orbitals and natural bonding orbital analysis. Technical issues were encountered for both which delayed this component of the research project.

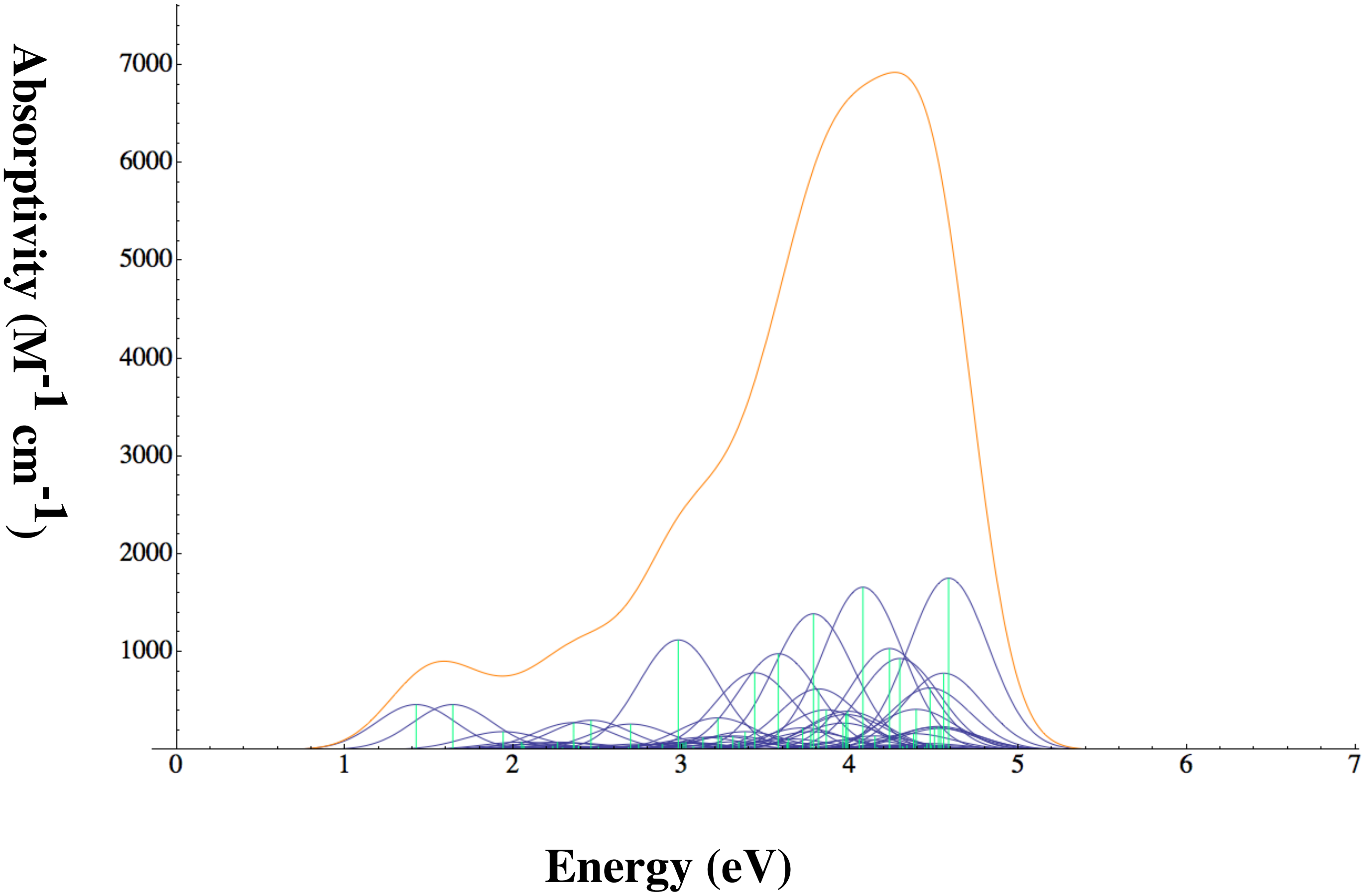

Figure 13. Example UV-Vis for complex real-world system.

Conclusion

We set out to answer one main question: “can we manage ‘big data’ within the computational chemistry laboratory?” As we grew to comprehend the magnitude of “big data”, we simultaneously gained a deeper understanding of the underlying complexities we often take for granted. For instance, we realized there are many aspects of our calculations that are nearly impossible to calculate by hand and must be automated. In turn, we sought to understand the fullest picture for which the electronic structure problem is solved. As we delved into the theory, we found parallel variables in our log files like that of energies and geometries. Unfortunately, how much time it took to find these variables and the overwhelming magnitude of the variables between numerous optimizations provided a bottleneck of sorts for analysis. As aspiring researchers faced with a road block, we overcame this road block by learning the Python programming language. With this knowledge, we were able to construct four programs that streamline the analyses of our chemical systems. Each with distinct benefits, we are now able to extract meaningful thermodynamic, geometric, and frequency data with a few keystrokes. Accordingly, if our optimization ever is not at a minimum, our fourth program sets out to displace along an “imaginary” frequency for which a new geometry is created that is at a minimum. Previously, one would have to process each imaginary frequency by hand using visualization software (~5 minutes per molecule). These programs worked well on test systems that had already been analyzed. However, where we really see the beauty of our programs was when we were offered an opportunity to work with a real life system. We gained insight into many attributes of the vanadium-oxo species that helped our collaborators understand the oxidation states within that molecule. We may not have realized until later reflection, we were able to focus solely on the analyses of the system and spent little time mulling over the magnitude of data in each log file, and future students in the lab may never have this realization (but this is a good thing!).

Big data is everywhere. In this paper we have demonstrated many instances that occur in computational chemistry. Moving forward, we look to write more complex programs that will further elucidate details that are often difficult to acquire with ease including, but not limited to, the UV-Vis program. Ultimately, we plan to disseminate that program and the others we have written for use in the computational chemistry community. We hope these programs will be a part of the grand solution for "big data" in computational chemistry.

Acknowledgements

AMT would dedicate this research to his best friend, Ashley Albone. AMT would like to acknowledge the help and encouragement from his family and friends as instrumental in this research process. AMT would like to thank Dr. Lord for the amazing opportunity and his constant support throughout the entire period of research. RLL acknowledges financial support from GVSU start-up funds and a GVSU-CSCE Faculty Research Grant-in-Aid. This work was made possible financially by a GVSU-OURS Student Summer Scholars award to AMT.

References

1) United States Census Bureau World Population Clock. http://www.census.gov/popclock/ (accessed September 5, 2013).

2) Mehl, M.R.; Vazire, S.; Ramirez-Esparza, N.; Slatcher, R.B.; Pennebaker, J.W.; *Science* **2007**, *317*, 82.

3) The World Bank Data. http://data.worldbank.org/ (accessed September 5, 2013).

4) IBM Big Data Platform. http://www-01.ibm.com/software/data/bigdata/ (accessed September 5, 2013).

5) Automation Definition. http://www.automationfederation.org/Content/NavigationMenu/General_Information/Alliances_and_Associations/The_Automation_Federation/About1/What_is_Automation_/What_is_Automation_.htm>. (accessed September 5, 2013).

6) a) Unleashing the Power of Big Data. http://www.whitehouse.gov/blog/2013/04/18/unleashing-power-big-data.(accessed September 5, 2013).

b) Manyika, James, et al. "Big data: The next frontier for innovation, competition, and productivity." (accessed September 5, 2013).

7) Harvard School of Public Health: The Promise of Big Data. http://www.hsph.harvard.edu/news/magazine/spr12-big-data-tb-health-costs/ (accessed September 5, 2013).

8) Feldman, Bonnie, Ellen M. Martin, and Tobi Skotnes. "Big Data in Healthcare Hype and Hope." (accessed September 5, 2013).

9) Buchanan, Matt. "Speak Softly and Carry a Big Data." http://www.newyorker.com/online/blogs/elements/2013/06/nsa-prism-big-data-national-security.html. (accessed September 5, 2013).

10) CERN Accelerating Science: *Computing*. http://home.web.cern.ch/about/computing. (accessed September 5, 2013).

11) Cramer, C. J. Essentials of Computational Chemistry: Theories and Models, 2nd ed.; John Wiley and Sons: Hoboken, NJ, 2004.

12) Atkins, P., and de Paula, J. Elements of Physical Chemistry. 6th ed.; W. H. Freeman, 2012.

13) M. Born and R. Oppenheimer, *Ann. Phys.* **1927**, *84*, 457.

14) Griffiths, D. J. Introduction to Quantum Mechanics; Prentice Hall: Upper Saddle River, New Jersey, 1995.

15) Kadanoff, L. P. *Journal of Statistical Physics*. **2009**, 137 (5–6): 777–797.

16) a) D.R. Hartree, *Proc. Cambridge Phil. Soc.* **1928**, *24*, 89.

b) V. Fock, *Z. Phys.* **1930**, *61*, 126.

17) Gaussian 09, Revision C.01, 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, T. Keith, 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, O. Farkas, J. B. Foresman, J. V. Ortiz, J. Cioslowski, and D. J. Fox, Gaussian, Inc., Wallingford CT, 2010.

18) Parr, R.G.; Yang, W. Density-functional theory of atoms and molecules; Oxford University Press: New York, 1989.

19) Vosko, S.H.; Wilk, L.; Nusair, M. *Can. J. Phys.* **1980**, *58*, 1200.

20) Lee, C.; Yang, W.; Parr, R.G. *Phys. Rev. B* **1988**, *37*, 785.

21) Becke, A.D. *J. Chem. Phys.* **1993**, *98*, 5648.

22) Stephens, P.J.; Devlin, F.J.; Chabalowski, C.F.; Frisch, M.J. *J. Phys. Chem.* **1994**, *98*, 11623.

23) a) Hay, P.J.; Wadt, W.R. *J. Chem. Phys.* **1985**, *82*, 270. b) Wadt, W.R.; Hay, P.J. *J. Chem. Phys.* **1985**, *82*, 284. c) Hay, P.J.; Wadt, W.R. *J. Chem. Phys.* **1985**, *82*, 299.

24) Python Software Foundation. Python Language Reference, version 3.0. http://www.python.org.

25) Lewis, N.S.; Nocera, D.G.; "Powering the planet: Chemical challenges in solar energy utilization" *Proc. Nat. Acad. Sci. USA* **2006**, *103*, 15729-15735.

26) Wolfram Research, Inc., Mathematica, Version 8.0, Champaign, IL (2010). http://www.wolfram.com/mathematica/.