TOOLKITS TO FACILITATE COMPUTATIONAL CHEMISTRY STUDIES

by

ANTHONY JAMES SCHAEFER

(Under the Direction of Steven E. Wheeler)

ABSTRACT

Computational quantum chemistry is a versatile tool for studying molecules and reactions. However, accurately computing properties requires manifold computations. Setting up, running, and processing these computations is a monotonous task. We have developed toolkits for facilitating these tasks. AARON, our automation tool, has been used to locate numerous transition state structures for an iridium-catalyzed C-H activation reaction for the purpose of assessing the accuracy of DFT methods. AaronTools is our Python toolkit for modifying structures and processing computations. AaronTools has been expanded work with several popular quantum chemistry software packages (Gaussian, ORCA, Psi4, Q-Chem, xTB, and SQM), plot simulated spectra, calculate several steric parameters, and generate molecular structures from simple input. SEQCROW, a ChimeraX bundle, provides a graphical interface for these features. Raven is an efficient reaction path search implementation that utilizes on-the-fly Gaussian process regression to locate guesses for transition state structures.

INDEX WORDS: computational chemistry, density functional theory, molecular descriptors, graphical tools, quantum chemistry, transition states

TOOLKITS TO FACILITATE COMPUTATIONAL CHEMISTRY STUDIES

by

ANTHONY JAMES SCHAEFER

BS, University of Saint Thomas, 2018

A Dissertation Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment of the Requirements for the Degree

DOCTOR OF PHILOSOPHY

ATHENS, GEORGIA

2022

TOOLKITS TO FACILITATE COMPUTATIONAL CHEMISTRY STUDIES

by

ANTHONY JAMES SCHAEFER

Major Professor:	Steven E. Wheeler
Committee:	Eric M. Ferreira
	Henry F. Schaefer III

Electronic Version Approved:

Ron Walcott
Vice Provost for Graduate Education and Dean of the Graduate School
The University of Georgia
August 2022

ACKNOWLEDGEMENTS

I thank the members of the Wheeler group, whose support and help has been invaluable. First, I thank Victoria. Without her major rework of AaronTools, this dissertation would look very different. I also want to thank Drew for believing in me, even if he didn’t want to. I thank Sharath for trying to teach me how to throw magnets at a whiteboard better. I thank Andrea for answering my questions and pretending that they weren’t stupid. I thank Bryan for smuggling me a 50/50 mix of candy and candy wrappers under our cubicles. I also want to thank Stephen for giving me rides to game night. I thank Laura for not abandoning me like everyone else did. In all seriousness, thank you to everyone for the helpful advice and assistant over the years. Finally, I thank Dr. Wheeler for his giving me his guidance and pushing me to do better.

I’d also like to thank my family, in particular my mom and brother, for supporting me throughout my education. I’d also like to express my gratitude for my friends, Trent and Gabe, for regularly helping me relax with video games and jedi audiobooks. Another thanks to my friends from UST for their interest, and also helping keeping spirits high in those times when we have to try hard: Steph, Cole, Brady, Kristen, Cecilia, and Josh.

TABLE OF CONTENTS

Page

ACKNOWLEDGEMENTS iv

LIST OF TABLES viii

LIST OF FIGURES ix

CHAPTER

1 INTRODUCTION 1

2 BENCHMARKING POPULAR DFT METHODS FOR REGIOSELECTIVITY OF AN IRIDIUM-CATALYZED C-H FUNCTIONALIZATION REACTION 5

Abstract 5

Introduction 5

Assessing DFT methods for Predicting Selectivity of C-H Activation 11

Methods 12

Results 13

Conclusions and Future Work 24

3 EXPANDING THE CAPABILITIES OF AARONTOOLS 27

Abstract 27

Introduction 27

Working with More Quantum Chemistry Software 29

Steric Descriptors 33

Simulating Spectra 35

Orbitals....36

Structure Building....36

Finders....37

Conclusions....38

4 SEQCROW: A CHIMERAX BUNDLE TO FACILITATE QUANTUM CHEMICAL APPLICATIONS TO COMPLEX MOLECULAR SYSTEMS....39

Abstract....40

Introduction....40

SEQCROW....41

Graphical Presets, Bond Editor, and Selectors....43

Structure Modification and Building....44

Setting up and Running Computations....46

Processing Computed Data....47

Structure Analysis....50

Installation and Availability....53

Conclusions....53

5 SEQCROW IMPLEMENTATION DETAILS....55

Abstract....55

Introduction....55

Adding Formats to the QM Input Builder Tool....56

Adding New Algorithms to the Transition State Structure Prediction Tool....58

Implementing New Job Types....58

Buried and Free Volume in 3D....60

SEQCROW as a Middleman 62

Conclusions 63

6 RAVEN: A DOUBLED-ENDED GROWING STRING IMPLEMENTATION ACCELERATED BY GAUSSIAN PROCESS REGRESSION 64

Abstract 64

Introduction 64

The Algorithm 69

Running Raven 71

Benchmark Set 72

Benchmarking Results 73

Iridium-Catalyzed C-H Activation 77

Conclusions and Discussion 78

7 CONCLUSIONS 81

REFERENCES 83

APPENDICES

A RAVEN BENCHMARKING SET 94

LIST OF TABLES

Page

Table 1: Site selectivity of the reaction in Scheme 1 with R1=CH3 based on the carboxylate additive 11

Table 2: Data for RMSE and correlation between relative barrier heights for experimental data and various DFT functionals. Superscripts denote methods used for single point energies where structures and frequencies were computed using a) DFT/6-31G(d,p)/LANL2DZ, b) def2-SVP/SDD, c) PCM implicit solvent, d) SMD implicit solvent. 21

Table 3: Most selective carboxylates according to different DFT methods based on quasi-RRHO free energies computed with DFT/def2-TZVP/SDD // DFT/def2-SVP/SDD 23

Table 4: Keywords for specifying miscellaneous when creating input files for different software packages 32

Table 5: AaronTools finder classes and their description 38

Table 6: Raven's default convergence criteria 71

LIST OF FIGURES

Page

Figure 1: More O'Ferrall-Jencks plot for C-H activation facilitated by an internal carboxylate.....7

Figure 2: 'Upright' (left) and 'sideways' (right) TSSs, and 2nd order saddle point (middle) connecting the two. Energies are in kcal/mol..14

Figure 3: Representative TSSs for the selectivity-determining step for the reaction in Scheme 1. Top left: activation of C2 with an ‘upright’ carboxylate; top right: activation of C7; middle: concerted activation of C2 with a ‘sideways’ carboxylate; bottom left: forming agostic intermediate at C2; bottom right: deprotonation of C2. 15

Figure 4: More O'Ferrall-Jencks plot of stationary points located with B3LYP/def2-SVP/SDD. The dashed line connects the average bond orders for reactant structures to that of the product structures...17

Figure 5: More O'Ferrall-Jencks plot of stationary points located with ωB97X-D/def2-SVP/SDD. The dashed line connects the average bond orders for reactant structures to that of the product structures...18

Figure 6: More O'Ferrall-Jencks plot of stationary points located with B97-D/def2-SVP/SDD. The dashed line connects the average bond orders for reactant structures to that of the product structures...19

Figure 7: More O'Ferrall-Jencks plot of stationary points located with BP86/def2-SVP/SDD. The dashed line connects the average bond orders for reactant structures to that of the product structures...20

Figure 8: DLPNO-CCSD(T)/cc-pVTZ/cc-pVTZ-PP // B3LYP/def2-SVP/SDD electronic energies vs. B3LYP/def2-TZVP/SDD // B3LYP/def2-SVP/SDD energies for TSSs with pentafluoropropionate 24

Figure 9: Computed VCD spectrum of methyl lactate 35

Figure 10: Two coordination complexes generated for a square planar coordination geometry with a chloride, carbonyl, and Quinox ligand 37

Figure 11: Overview of features available in SEQCROW 42

Figure 12: TSS for a Rh-catalyzed asymmetric hydrogenation of acetaldehyde represented with different graphical presets from SEQCROW: ball-stick-endcap, sticks, and VDW 43

Figure 13: 2D sketch of dexamethasone and the resulting 3D structure 45

Figure 14: Energies, enthalpies, and free energies (RRHO, quasi-RRHO, and quasi-harmonic) can be evaluated by combining energies from different QM packages at any specified temperature 48

Figure 15: Choosing conformers to generate an IR spectrum, and an ECD spectrum with the most populated conformers displayed separately 49

Figure 16: A bonding PNBO of formamide and the orbital-weighted Fukui dual function for pentafluorothioanisole 50

Figure 17: Sterimol parameters for a phenyl substituent, and the cone angle, buried volume, and steric map for a BINAP ligand on a rhodium coordination complex 51

Figure 18: Vectors representing the Sterimol parameters for several substituents 53

Figure *19*: SECQROW's QM Input Builder tool, with the file formats displayed 57

Figure 20: (left) Initial and (right) final visualization of free volume around a metal center. The metal and ligand are not shown. 61

Figure 21: Structures located a poor-quality initial pathway (top) and optimized pathway (bottom) for the Claisen rearrangement of allyl vinyl ether 61

Figure 22: A snapshot of the DE-GSM method applied to a simple potential energy surface; the line represents the pathway at this iteration, and white dots (nodes) are locations where the gradient is computed during this iteration..67

Figure 23: Example config file for running Raven 72

Figure 24: Key performance metrics for Raven. A) RMSD between Raven's guess TSS and the optimized TSS; B) computations required for Raven to converge in successful cases; C) difference in energy between Raven's successful guess TSSs and the optimized TSS; D) difference in energy between the TSS optimized from Raven’s guess and the TSS reported by Jónsson *et al*...74

Figure 25: Mechanism reported by Jónsson *et al.* (top) and mechanism discovered by Raven (bottom); energies relative to the reactant are in kcal/mol 76

Figure 26: Raven's key TS guesses and intermediates for the decomposition of sulfolene to SO2 and 1,3-butadiene. R and P indicate the reactant and product, respectively......................77

Figure 27: Raven’s guess TSS (left) and optimized TS guess (right) for the C-H activation of N-acetyl indole 78

CHAPTER 1

INTRODUCTION

Computational chemistry has proven to be quite useful over the past few decades. It gives us a glimpse at molecules, which is often not available from experiment. As a result, we can look at transient structures and identify the interactions that govern reactivity and selectivity. Quantum mechanical methods are also capable of simulating spectra, which can help support structure assignments in cases where experimental data is absent or incomplete. Computational methods can also directly compute the energy of a given structure of a molecule, which allows for the prediction of relative rates of reactions or equilibrium constants.

Quantum mechanical computations have been particularly useful for catalyst design, where materials and lab time can be costly. Density functional theory (DFT) is the go-to method for studying transition metal-catalyzed reactions. Due to its reasonable balance between computation time and accuracy, DFT can often be used to map out the energy of intermediates and transition state structures of a catalytic cycle. The catalyst can be adjusted (*e.g.*, changing a ligand), and the energy of key energy minima and maxima along the catalytic cycle recomputed. This can indicate if the changes to the catalyst make it more effective, which can then guide future experiments.

More and more, reactivity is being correlated with calculated or computed parameters.[1, 2] Some examples of this are Tolman cone angles[3] or computed atomic charges. Using such parameters can reduce computation time, compared to explicit computations of reaction barrier heights, as their evaluation typically only requires a truncated portion of the catalyst or substrate. Local minima may also be sufficient for computing parameters, so the more expensive transition state optimizations can be skipped. Parameters correlating with reactivity can shed light on what is chemically important, which can help guide catalyst design. Catalysts or ligands that do not meet a threshold for key parameters can be discarded before progressing to more thorough computational or experimental stages of development.

Accurate predictions of catalyst efficacy can require hundreds of computations, locating many transition state structures (TSSs), and choosing an appropriate level of theory. Setting up and running hundreds of computations manually is tedious and error prone. A mistyped setting or bad molecular structure could lead to misleading or erroneous results. It is not uncommon for quantum chemical computations to end with errors. Inexperienced users might have a difficult time identifying and addressing the source of the error. Errors aside, finished computations will need to be processed.

To facilitate many of these computational tasks, the Wheeler group has developed AARON and AaronTools.[4] AARON is a command line utility that can take a set of template structures, swap substituents or ligands, run Gaussian computations, and process the resulting energies. Some rotamers of substituents can also be explored automatically. This helps with some of the more monotonous parts of a computational study. AARON can also try to fix common errors and adjust structures if optimizations appear to be going awry. AaronTools is at the foundation of AARON, handling the individual tasks that AARON requires (building and modifying molecular structures, constructing input files and parsing output files, *etc.*). With AARON, users should have more time to work on other things (*i.e.*, they can think about the chemistry, not the computations).

Although AARON has been used successfully in some chemical applications,[5-10] it has some limitations. One is that it can only run certain types of computations, namely geometry optimizations, frequency computations, and single-point energies, and only using Gaussian. Although these job types are generally the bread and butter of computational studies, others are often useful. AARON also requires the user to supply templates of TSSs if optimizing saddle points. Locating transition state structures is one of the more challenging and time-consuming aspects of computational kinetics studies.

Another issue with AARON and many other tools for computing tasks is that they can only be used on the command line. When the primary goal is to learn more about chemistry or computational methods, learning to use computational tools should take as little time as possible so that users can focus on their project. Having to learn esoteric terminal commands is time-consuming and should not be necessary. The command line is little more than a means to an end.

Tools with a graphical interface can be much easier to learn, particularly for those without backgrounds or experience in high-performance computing. However, computational chemistry tools are overwhelmingly made by computational chemists. Chemists who have built tools are probably experienced with the command line and would consider learning the command line to be tacitly required to conduct a computational study. Moreover, developing graphical interfaces is not a common skill for computational chemists since their focus tends to be on developing ‘more powerful’ command-line tools. The command line was once essentially required to work with high-performance computing clusters, but this is no longer the case. Server hardware from the last decade can run 3D graphical applications that can be accessed through a web interface.[11, 12] With such modern hardware, it is now feasible to run computations using the latest high-performance computing resources through a graphical interface.

Herein, I describe four projects. The first (Chapter 2) is the application of AARON to benchmark the performance of DFT in the prediction of the outcome of an iridium-catalyzed CH functionalization. This highlights both how powerful tools like AARON are, as well as areas where AARON falls short. Next, in Chapter 3, I detail additions I have made to AaronTools to make it more generally useful, namely by adding the ability to calculate several descriptors, process computations to generate simulated spectra, and add compatibility for multiple quantum chemistry software packages. In Chapters 4 and 5, I describe the features and development of a graphical interface for these tools called SEQCROW. Finally in Chapter 6, I introduce an efficient implementation of a transition state search algorithm using on-the-fly Gaussian process regression to locate guesses for transition state structures.

CHAPTER 2

BENCHMARKING POPULAR DFT METHODS FOR REGIOSELECTIVITY OF AN IRIDIUM-CATALYZED C-H FUNCTIONALIZATION REACTION

2.1 Abstract

DFT has been widely used in efforts to understand transition metal catalyzed reactions. However, the accuracy of these methods when applied to regioselective transition metal catalyzed reactions remains understudied. We study the iridium-catalyzed site-selective C-H activation of N-acetyl indole with a variety of DFT methods to assess popular DFT methods based on their ability to reproduce experimentally observed regioselectivity. Multiple pathways were located for activation at the C_2 position. It seems unlikely that application of popular DFT methods will accurately reproduce experimental product ratios, but it may still be possible to use DFT to predict more selective catalysts.

2.2 Introduction

Transition metal-catalyzed selective C-H functionalization is important for synthetic organic chemistry. Being able to target a specific C-H bond allows for late-stage modifications. However, controlling which C-H bond is activated can be quite challenging. Understanding the mechanism of these reactions is key for developing more selective reactions and catalysts. Gaining detailed insight from experimental studies alone is often not feasible due to the lability of reactive intermediates and the complexity of the synthetic conditions. Thus, computational quantum chemistry is frequently relied upon to gain insight into the mechanism of catalysts and substrates that promote selective C-H activation.

Fagnou *et al.* have used competition studies to show that some Pd(II) catalysts with carboxylate ligands target more acidic C-H bonds.[13] A kinetic isotope effect (KIE) study revealed a k_H/k_D of 3.0 for C-H activation of 1,3-difluorobenzene, indicating significant involvement of the proton on the rate-limiting step. Theoretical studies suggested a concerted metalation-deprotonation (CMD) event that is facilitated by an acetate ligand. This CMD mechanism offered a rationale for Pd(II) catalysts often tending to activate more acidic C-H bonds. However, not all base-assisted Pd(II)-catalyzed C-H activation reactions are consistent with this trend. For example, Gevorgyan *et al.* investigated the C-H activations of indolizines, and found no KIE.[14] This and other evidence pointed towards an electrophilic substitution mechanism.

In order to shed light onto the disparity in the behavior of different transition metal catalysts for different C-H activation reactions, Carrow and Wang computed bond orders for C-H activations TSSs for several catalysts and substrates.[15] These bond orders were used to create More O'Ferrall-Jencks plots (see Figure 1). On these plots, the bond orders are plotted on different axes. The reactants and products will be in opposite corners, and TSSs and intermediates can be elsewhere on the plot. The farther a TSS or intermediate is from the diagonal, the more asynchronous the mechanism. The extreme asynchronous cases would either form a Wheland intermediate before deprotonating, or the base would deprotonate before metalation starts.

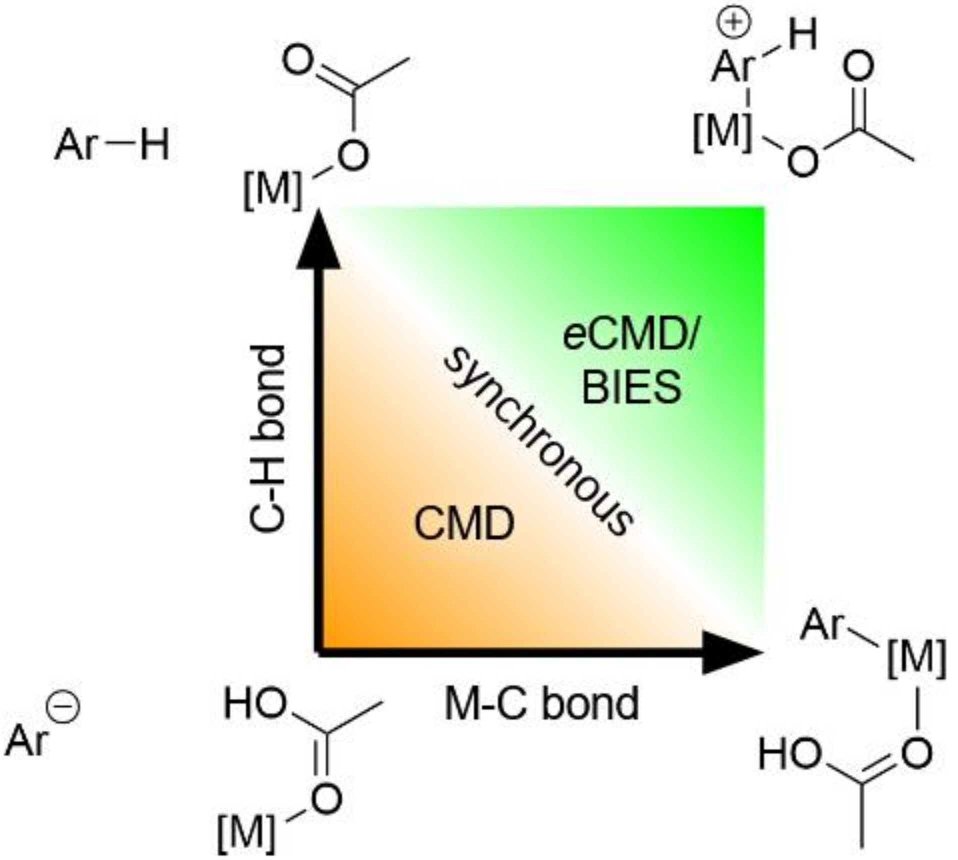
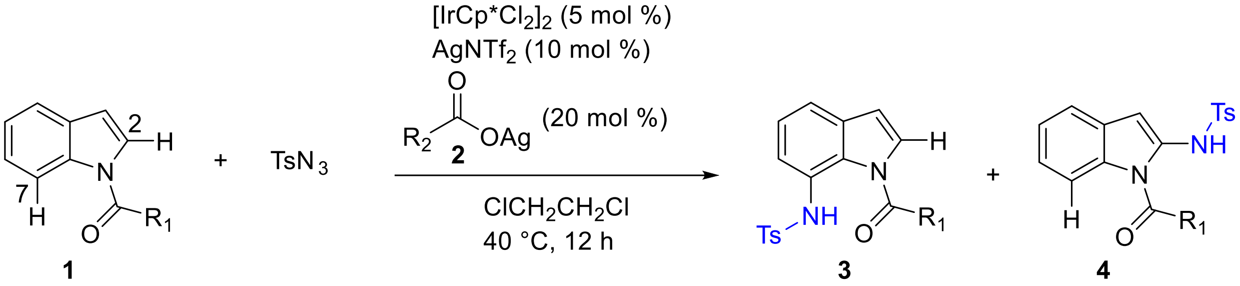


Figure 1: More O'Ferrall-Jencks plot for C-H activation facilitated by an internal carboxylate, with the C-H bond on the y-axis and the metal-C bond on the x-axis.

Performing More O’Ferrall-Jencks analysis revealed different types of asynchronous mechanisms were at play. Electron poor d^6 and d^8 catalysts tended to form the metal-carbon bond faster than the C-H bond broke. Conversely, electron rich d^8 and d^{10} metals tended to break the C-H bond faster. The electron-richness of d^8 metals can be controlled to some extent with ligands. How asynchronous the mechanism is appears to be independent of the substrate, though the substrate could shift the TSS earlier or later. They suggested that this provided a theoretical understanding of Fagnou’s CMD mechanism and ones that behave more like electrophilic substitution, which they termed “*e*CMD”. Carrow and Wang noted that catalysts that use the *e*CMD mechanism favor more electron rich C-H bonds, so as to complement the electron deficiency of the metal.

Unfortunately, the terms "CMD" and "*e*CMD" conflate the synchronicity of the mechanism with the number of steps in the mechanism. Without the context of a More O'Ferrall-Jencks analysis, CMD might simply be used to describe any metalation-deprotonation event where there is no evidence of an intermediate. Instead of *e*CMD, Ackermann described this mechanism as a base-assisted internal electrophilic substitution (BIES).[16] Ackermann *et al.* studied several reactions involving Ru(II), Rh(III), Co(III), and Ni(II).[17] They were able to locate agostic intermediates in each of their reactions. Using More O'Ferrall-Jencks analysis revealed that their TSSs and intermediates all fall in the BIES region.



Scheme 1: Ir-catalyzed C-H amidation of N-acyl indoles reported by Chang *et al.*[18]

Of course, the catalyst is just one method to control regioselectivity. Chang *et al.* have studied the functionalization of N-acyl indoles shown in Scheme 1.[18] Their synthesis takes advantage of the acyl directing group, which, when coordinating the iridium catalyst, only allows the metal to reach the C_2 and C_7 positions of the indole. Chang noted that using bulkier acyl groups favors functionalization at C_7. For example, with R_1 = *t*Bu and R_2 = Me, they were able to achieve a **3**:**4** ratio of > 20:1. This was rationalized by the bulkier acyl group having more unfavorable steric interactions when the carbonyl is facing C_2, thus promoting C_7 activation.

Chang *et al.* were also able to achieve good selectivity for C_2 functionalization with the iridium catalyst by modifying the carboxylate additive.[18] This is in spite of Carrow and Wang's suggestion that the d^6 metal should preferentially activate the less acidic C_7-H bond.[15] Using more electron deficient carboxylates, such as trifluoroacetate, was shown to heavily favor C_2 activation.

In order to better understand this trend in reactivity, Chang *et al.* conducted a computational study to look at the pathways to activate C_2 and C_7. The presumptively relevant TSSs and intermediates were located for both pathways with pivalate and trifluoroacetate. An agostic intermediate was identified for both activation pathways where the C-H bond has replaced a Ir-O bond with the carboxylate. The agostic intermediate going towards C_7 activation was found to be lower in energy for both carboxylates. This was attributed to the C_2-H bond being more electron deficient, which would make it a worse electron donor. For trifluoroacetate, the agostic intermediate in both activation pathways is more similar in energy to the indole only coordinating with its directing group, when the carboxylate is a κ^2 ligand. This could be explained by the weaker electron donor strength of trifluoroacetate, being more similar to a weakly-donating agostic interaction. The computational studies also revealed a qualitative agreement with experiment: C_7 activation was favored for pivalate, and C_2 activating was favored for trifluoroacetate.

In order to establish a quantitatively predictive model for reactivity, Chang *et al.* investigated several descriptors proposed by Sigman *et al.*,[1] including vibrational frequencies, Sterimol parameters, and NBO charges for the carboxylic acid analogues and acyl directing groups. The Sterimol parameters of the R_2 group of the carboxylate showed poor correlation with experimental product ratios. However, NBO charges on the oxygen atoms appeared to be a strong indicator of selectivity. A linear regression model was established relating the average NBO charge of the carboxylic acid's oxygen atoms to the relative barrier height calculated from experimental product ratio. The model has an R^2 of 0.91 across 23 different carboxylates. These cases only include experimental data for N-acetyl indole, so this regression model does not include the steric effect of the directing group on indole. A multivariate model incorporating the B_1 Sterimol parameter of the acyl directing group with the NBO charges achieved an R^2 of 0.94 across several combinations of carboxylates with various N-acyl indoles.

Density functional theory (DFT) is typically the go-to method for studying organometallic system. DFT computations have helped support mechanistic studies, such as those of Fagnou and Carrow and Wang.[13, 15] DFT was also used to build predictive models of reaction outcome.[1, 2, 18] It is often assumed that DFT strikes a good balance between accuracy and computational cost. This is often grounded in benchmarking studies that look at DFT predictions of minima or reaction barrier heights.[19-21] These studies offer hope, as DFT energies typically fall within a few kcal/mol of reference energies.

However, predicting a product ratio requires relative energies. If errors do not cancel favorably, the predictions could be off by much more than these benchmarking studies suggest. The performance of DFT for predicting regioselectivity of transition metal-catalyzed reactions has not been well-studied. This can be attributed to the difficulty of obtaining quality reference data to draw comparisons. Many transition metal-catalyzed reactions involve large substrates and/or large ligands, which are often not feasible to study using higher levels of theory. Additionally, such a study would require numerous transition state optimizations, which are notoriously annoying and time-consuming. Studying the performance of popular DFT methods for such reactions should clarify how much faith to put into computational results.

2.3 Assessing DFT methods for Predicting Selectivity of C-H Activation

In order to assess the predictions of various DFT methods, we used DFT to predict the product ratios reported by Chang *et al.* for the regioselective functionalization of N-acetyl indole (see Scheme 1).[18] Chang *et al.*'s data provides a range of product ratios, allowing us to judge not only DFT's ability to reproduce experimental catalyst selectivity, but also gauge the ability to predict trends in selectivity. Performing well with either of these would indicate that DFT studies can be used to make predictions about the efficacy catalysts which have not been tested experimentally. We do not build parameter-based regression models, like Chang *et al.* did. Instead, we compute the energy of conformer ensembles for the relevant TSSs. In order to avoid a more prolonged conformer search, we limited our study to the subset of the 16 less flexible carboxylates shown in Table 1.

Table 1: Site selectivity of the reaction in Scheme 1 with R_1=CH_3 based on the carboxylate additive

entry	R_2	3:4	$\Delta\Delta G^{\ddagger}$	entry	R_2	3:4	$\Delta\Delta G^{\ddagger}$
1	*t*Bu	2.5:1	0.56	9	(4-F)C_6H_4	1:2.2	-0.49
2	*i*Pr	1.6:1	0.30	10	(4-Cl)C_6H_4	1:2.3	-0.51
3	Et	1.5:1	0.26	11	C_6H_5	1:2.7	-0.63
4	Me	1.4:1	0.19	12	(4-Br)C_6F_4	1:6.0	-1.12
5	CH_2Cl	1:2.6	-0.58	13	CHF_2	1:12.3	-1.56
6	CH_2F	1:2.8	-0.64	14	$CHCl_2$	1:13.3	-1.61
7	(4-CF_3)C_6H_4	1:1.8	-0.36	15	CF_3	1:61.9	-2.57
8	(4-NO_2)C_6H_4	1:1.8	-0.38	16	C_2F_5	1:73.1	-2.67

2.4 Methods

The methods we will be testing are B3LYP, B3LYP-D3, B3PW91, ωB97X-D, B97-D, BP86, PBE0, M06, M06-2X, M06-L, and M06-HF. These were found to be some of the most used DFT methods for studying organometallic catalysts.[22] Each of these methods were tested in conjunction with the def2-SVP basis set, with the SDD effective core potential (ECP) on the iridium. The def2-TZVP basis set was also used to compute single-point energies on structures optimized using the def2-SVP basis set. For select methods, the 6-31G(d,p)/LANL2DZ and cc-pVDZ/SDD basis sets and ECPs were also tested, as this has long been a popular basis set/ECP combination for organometallic systems. A few methods were also tested with the inclusion of implicit solvent, either throughout the geometry optimization or only during a single-point energy evaluation for structures optimized in the gas phase. The vast majority of these computations were carried out using AARON, which searched for different conformers of the R_2 group on the carboxylate.[4] DFT computations were conducted using the Gaussian 09 software package.[23] Relative barrier heights were calculated for enthalpy, RRHO free energy, and quasi-RRHO free energy with ω=100 cm^{-1}.[24]

For structures optimized using select DFT methods, single-point energies for were also computed at the DLPNO-CCSD(T) level paired with the cc-pVTZ basis on non-metal atoms and the cc-pVTZ-PP basis set and SK-MCDHF-RSC ECP on the iridium. The DLPNO computations were conducted using ORCA 4.2.[25] By comparing the DFT energy of a structure to that of a higher level of theory, we can assess whether DFT predictions are sound.

2.5 Results

2.5.1 Mechanistic Details

It became clear that this reaction was not as straightforward as anticipated. Initially, we only looked for TSSs like the ones published by Chang *et al.*,[18] where the TSS connects the agostic intermediate to the metalated intermediate and the carboxylate is in an 'upright' orientation. However, we found that alternative pathways exist for C-H activation of C_2 for certain carboxylates with some DFT methods. In one, the carboxylate adopts a 'sideways' orientation. In addition, some fluorinated carboxylates were found to deprotonate and metalate in separate steps. Figure 3 shows representative TSSs for these different mechanisms.

Although we have not located all mechanisms with all the DFT methods we tested, we cannot rule out the possibility that some of these mechanisms are inoperable for some DFT methods. TSSs with 'sideways' carboxylates have been found for nearly all of the R_2 groups for B3PW91, B3LYP, PBE0, and BP86. The 'upright' and 'sideways' pathways are generally similar in free energy, with the 'sideways' pathway typically being slightly favored. On top of this alternate pathway being slightly lower in free energy, having more reaction pathways leads to a more accessible transition state ensemble, which will impact predictions of regioselectivity. Thus far, alternate pathways have only been found for activation of C_2.

We looked in more detail at the sideways and upright pathways for benzoate at the BP86/def2-SVP/SDD level of theory (see Figure 2). In this case, the difference in electronic energy between the two first order saddle points is 0.8 kcal/mol in favor on the 'sideways' TSS. A second order saddle point has been located for the interconversion of the 'upright' and 'sideways' TSSs. This second order saddle point is only 0.9 kcal/mol higher in energy than the 'sideways' TSS. This highlights the flatness of the potential energy surface in the region surrounding these two transition state structures, which likely underlies the difficulty in locating one or the other TSS at different levels of theory.

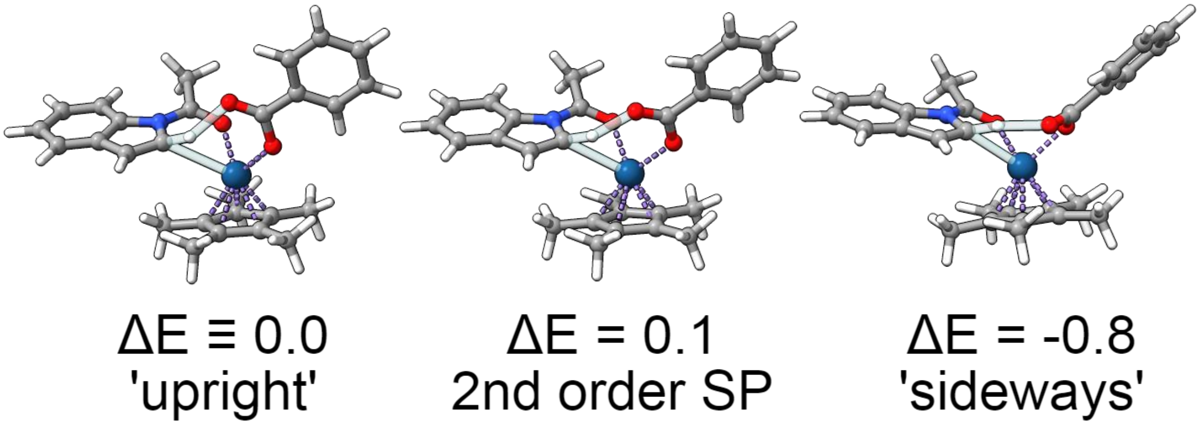


Figure 2: 'Upright' (left) and 'sideways' (right) TSSs, and 2nd order saddle point (middle) connecting the two. Energies are in kcal/mol.

Several functionals been found to have a qualitatively different C_2 activation mechanism than the M06 mechanism Chang *et al.* reported. Several methods, including B3LYP, ωB97X-D, and B97-D, lack an agostic intermediate for most carboxylates on the C_2 activation pathway. With these functionals, only fluorinated carboxylates (namely CHF_2, CF_3, and C_2F_5) seem to have agostic intermediates. Furthermore, BP86 lacks an agostic intermediate for most carboxylates for the C_7 pathway. In the cases where distinct proton-transfer and metalation steps have been found, the difference in energy between these two steps is typically less than 1 kcal/mol, so both steps will have some impact on reaction rate. Furthermore, in some cases, which step is rate-limiting depends on if you look at electronic energy or free energy. Unfortunately, experimental data does not clarify if this is a multistep mechanism, or which step is limiting if it is multistep.

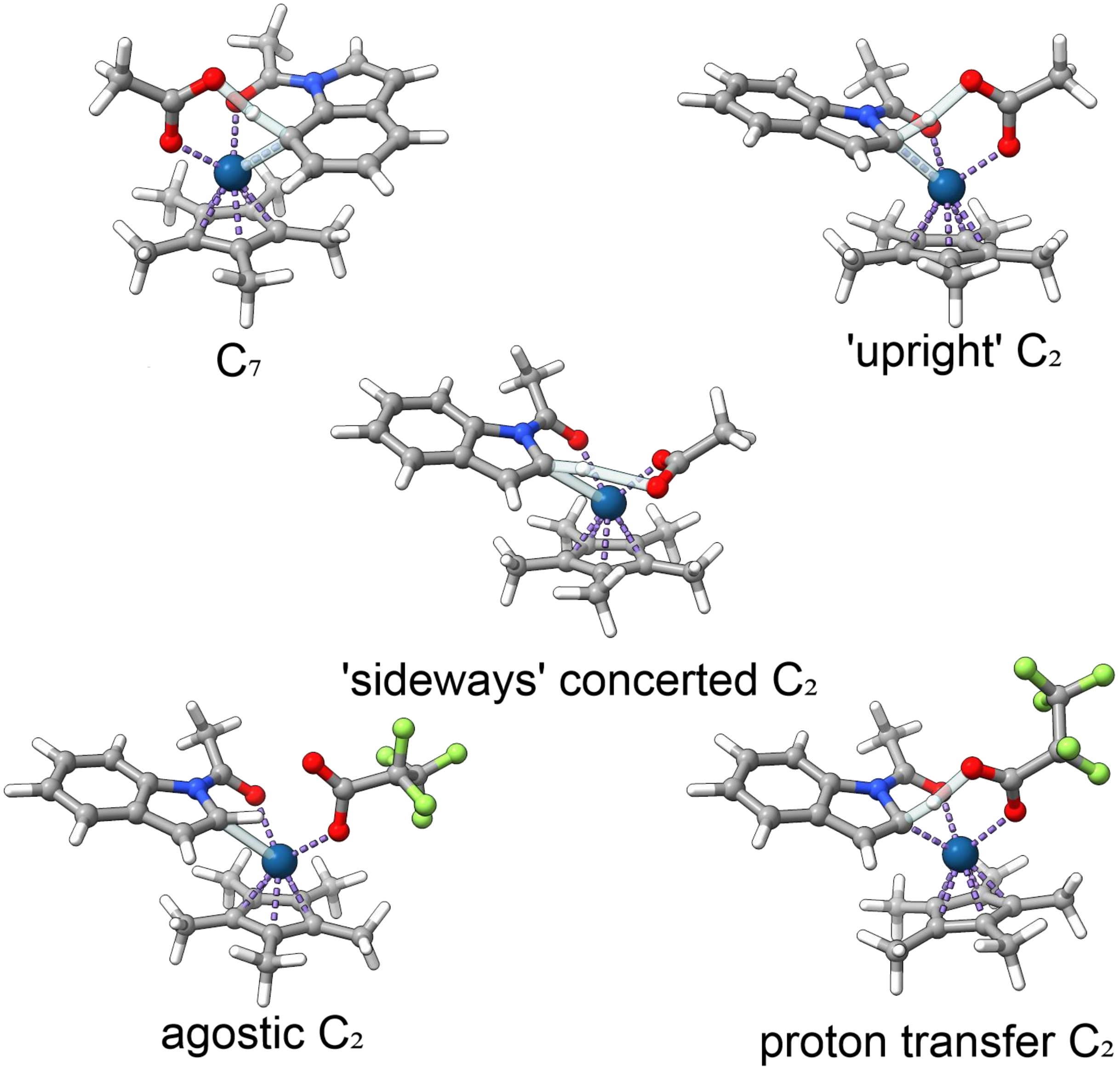


Figure 3: Representative TSSs for the selectivity-determining step for the reaction in Scheme 1. Top left: activation of C_2 with an 'upright' carboxylate; top right: activation of C_7; middle: concerted activation of C_2 with a 'sideways' carboxylate; bottom left: forming agostic intermediate at C_2; bottom right: deprotonation of C_2.

In order to better understand this reaction and the continuum between concerted and stepwise mechanisms, we computed Wiberg bond orders at different stages along the selectivity-determining step for the B3LYP/def2-SVP/SDD structures. Using the NBO package included in Gaussian 09, bond orders were computed for transition state structures, intermediates, reactants, and products of the selectivity-determining step located using B3LYP/def2-TZVP/SDD. Figure 4 shows a More O-Ferrall-Jencks plot of the activated C-H and C-Ir bonds orders (*i.e.*, bonds involving either C_2 or C_7, depending on which was being activated). As noted above, transition state structures above the dashed line would fall in the base-assisted internal electrophilic substitution (BIES) region. Below the line lies the concerted metalation-deprotonation (CMD), where more asynchronous mechanisms deprotonate first and metalate second.

In agreement with Carrow and Wang's findings,[15] the transition state structures all fall in the BIES (*e*CMD) region. While the position of the C_7 TSS's is consistent across all carboxylates, there is considerable variation in the nature of the C_2 activation pathways. The location of the agostic intermediates on the C_2 pathway range from near the 'sideways' TSSs to near the proton transfer TSSs. Although agostic intermediates are common in transition metal-catalyzed C-H activations reactions, it was not observed for all carboxylates. Less basic carboxylates seem to be more likely to form the agostic intermediate, whereas more basic carboxylates seem to deprotonate while the agostic interaction is forming.

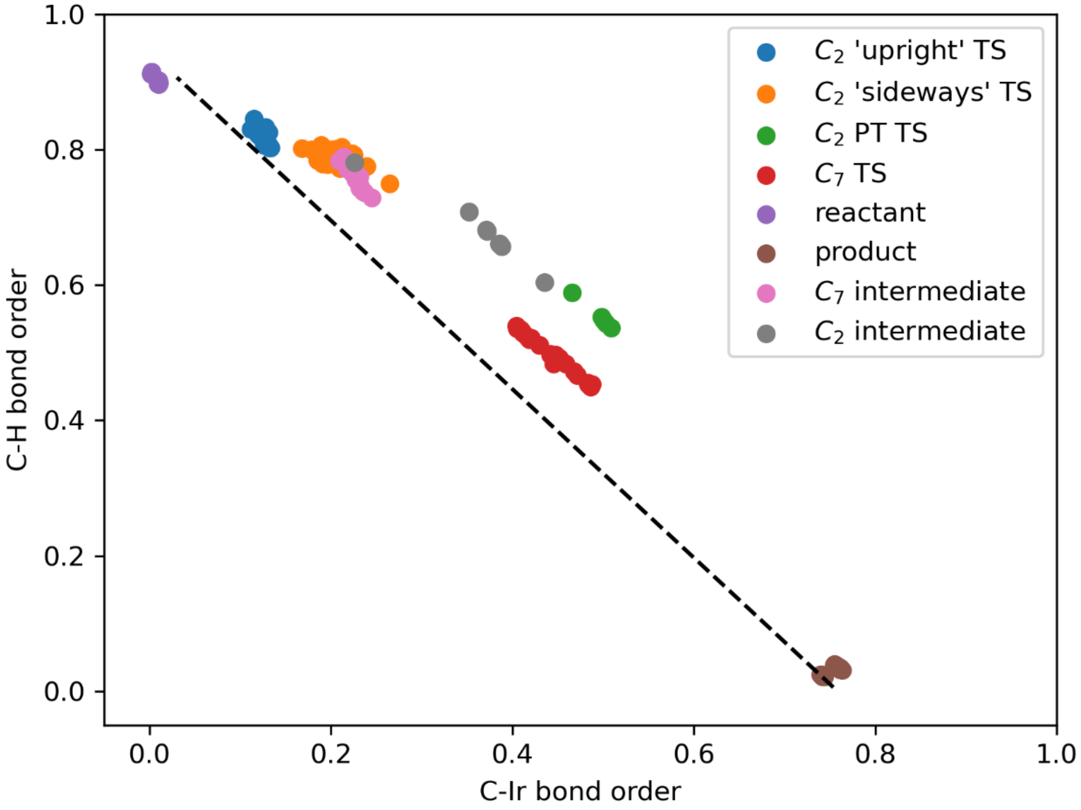


Figure 4: More O'Ferrall-Jencks plot of stationary points located with B3LYP/def2-SVP/SDD. The dashed line connects the average bond orders for reactant structures to that of the product structures.

To see how such mechanistic conclusions might change with different DFT methods, bond orders were also computed for several other methods. More O’Ferrall-Jencks plots for ωB97X-D, B97-D, and BP86 are shown in Figure 5, Figure 6, and Figure 7, respectively. As with B3LYP, ωB97X-D and B97-D indicate a BIES mechanism. The ωB97X-D TSSs for the proton transfer step for C_2 and C_7 are a bit more synchronous than B3LYP. For B97-D, the C_2 proton transfer TSSs appears to be slightly more asynchronous than B3LYP and are close to the agostic intermediate. BP86 deviates significantly from the rest, with the C_7 TS being much earlier and eking into the CMD region. Only two agostic intermediates could be located for the

C_7 pathway, both in the BIES region. There appears to be qualitative differences between the mechanism depending on the DFT functional that is used, which casts doubt on all-to-common practice of drawing mechanistic conclusions from DFT predictions using a single functional.

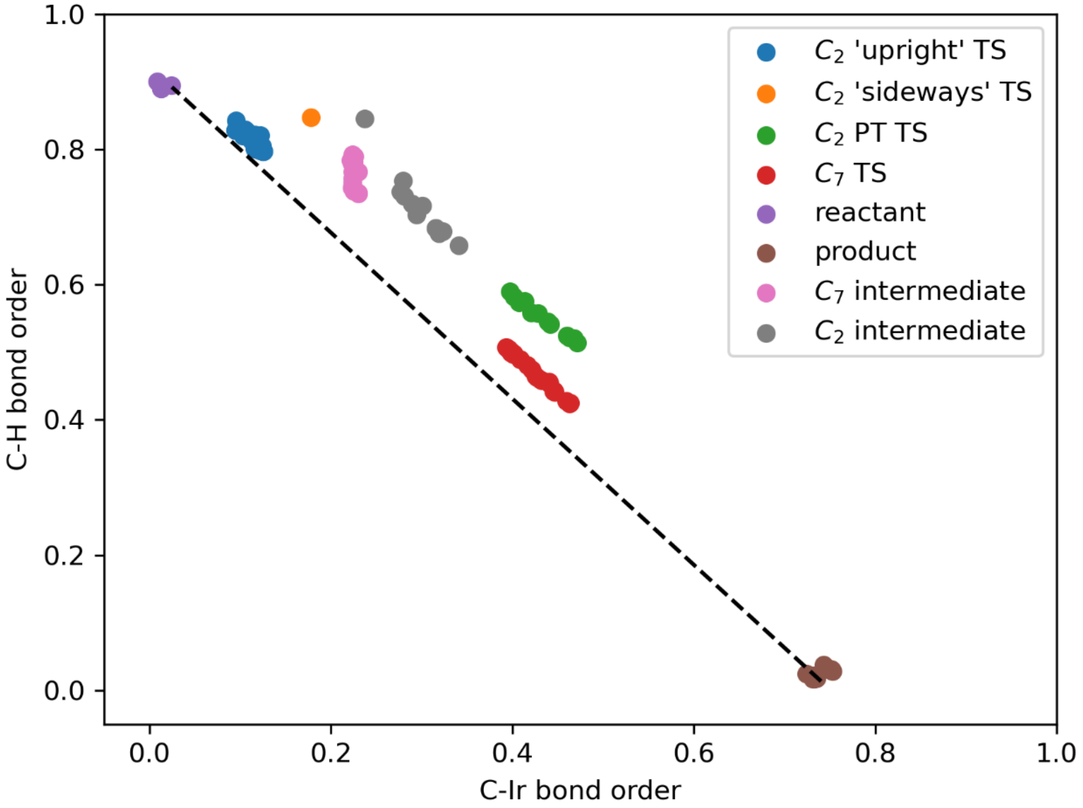


Figure 5: More O'Ferrall-Jencks plot of stationary points located with ωB97X-D/def2-SVP/SDD. The dashed line connects the average bond orders for reactant structures to that of the product structures.

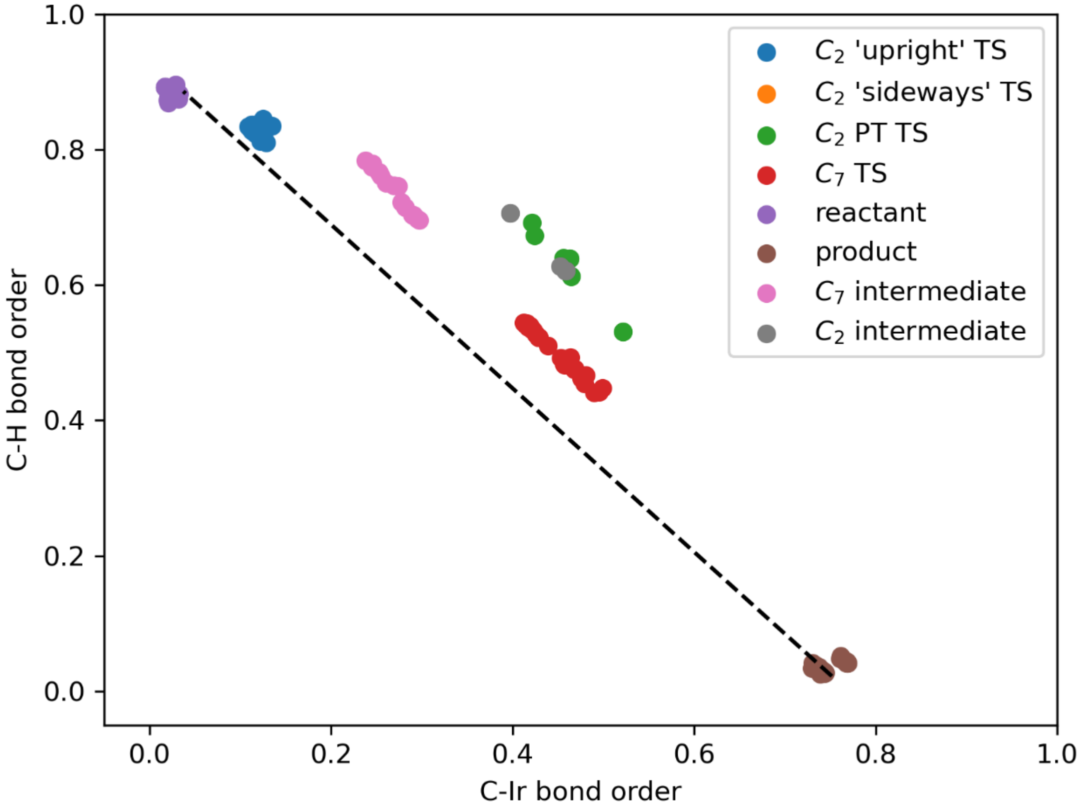


Figure 6: More O'Ferrall-Jencks plot of stationary points located with B97-D/def2-SVP/SDD. The dashed line connects the average bond orders for reactant structures to that of the product structures.

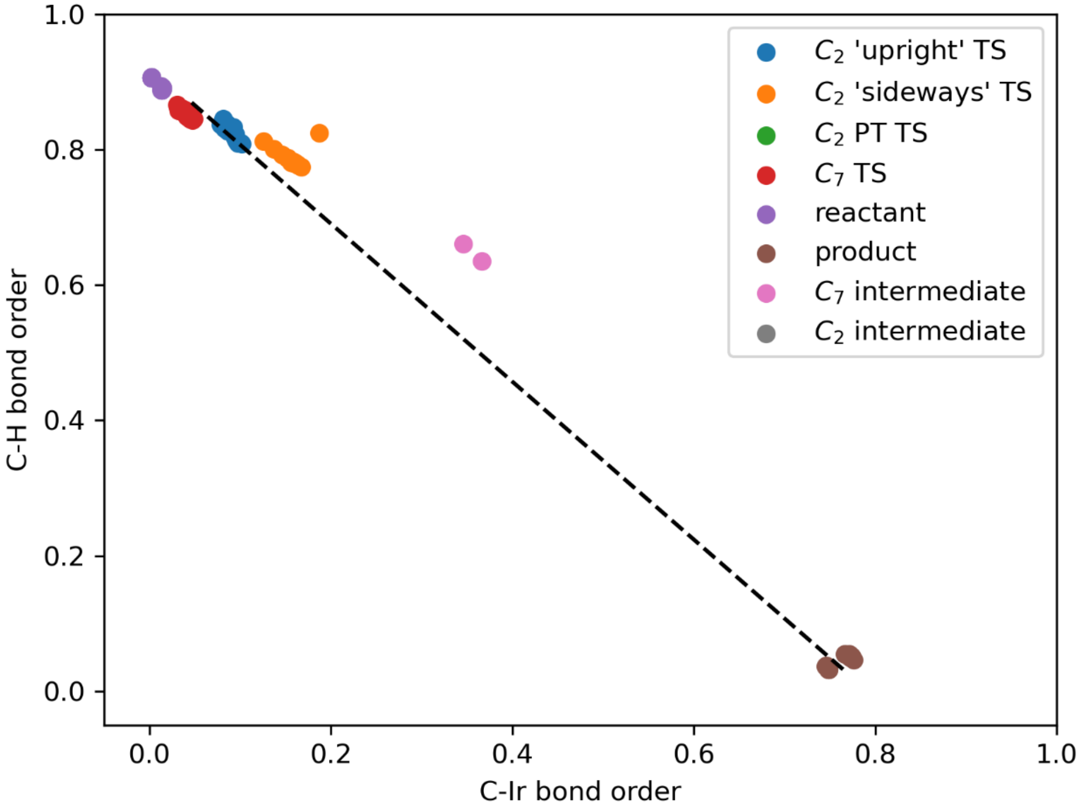


Figure 7: More O'Ferrall-Jencks plot of stationary points located with BP86/def2-SVP/SDD. The dashed line connects the average bond orders for reactant structures to that of the product structures.

2.5.1 DFT Benchmark

Next, we turn to assessing the performance of popular DFT methods in predicting the regioselectivity of this reaction. Due to either missing many TSSs or the ‘sideways’ TSSs not being found or ruled out, the only methods for which the data is complete are B3LYP, PBE0, and BP86. That being said, the current data for most DFT methods tested are summarized in Table *2*.

Table 2: Data for RMSE and correlation between relative barrier heights for experimental data and various DFT functionals. Superscripts denote methods used for single point energies where structures and frequencies were computed using a) DFT/6-31G(d,p)/LANL2DZ, b) def2-SVP/SDD, c) PCM implicit solvent, d) SMD implicit solvent.

			H		**G**		**Quasi-G**	
Functional	**Basis Set**	**ECP**	**RMSE**	**r^2**	**RMSE**	**r^2**	**RMSE**	**r^2**
B3LYP	6-31G(d,p)	LANL2DZ	2.0	0.81	1.3	0.72	1.9	0.81
B3LYP[a]	6-311+G(d,p)	SDD	0.7	0.84	0.5	0.77	0.4	0.85
B3LYP	def2-SVP	SDD	2.3	0.87	1.3	0.92	1.9	0.89
B3LYP[b]	def2-TZVP	SDD	0.5	0.84	0.9	0.89	0.4	0.85
B3LYP[c]	def2-SVP	SDD	0.8	0.88	1.3	0.71	0.9	0.91
B3LYP[d]	def2-SVP	SDD	2.9	0.61	3.7	0.01	3.2	0.18
B3LYP-D3	def2-SVP	SDD	4.9	0.65	3.8	0.83	4.4	0.79
B3LYP-D3	def2-TZVP	SDD	2.7	0.83	1.8	0.78	2.2	0.87
ωB97X-D	def2-SVP	SDD	5.5	0.87	4.4	0.62	5.1	0.85
ωB97X-D[b]	def2-TZVP	SDD	3.7	0.85	2.6	0.78	3.2	0.85
B97-D	def2-SVP	SDD	7.5	0.43	6.3	0.28	6.9	0.35
B97-D[b]	def2-TZVP	SDD	5.7	0.50	4.5	0.32	5.0	0.43
BP86	def2-SVP	SDD	1.7	0.52	1.8	0.40	1.6	0.35
BP86[b]	def2-TZVP	SDD	1.2	0.11	1.3	0.04	1.2	0.03
B3PW91	def2-SVP	SDD	3.3	0.04	1.9	0.24	2.1	0.04
B3PW91[b]	def2-TZVP	SDD	1.2	0.09	0.5	0.38	2.6	0.10
PBE0	def2-SVP	SDD	4.8	0.14	3.6	0.09	4.2	0.00
PBE0[b]	def2-TZVP	SDD	3.6	0.33	2.3	0.17	3.0	0.21
M06-2X[b]	def2-TZVP	SDD	2.4	0.85	1.6	0.42	2.1	0.68
M06-L	def2-SVP	SDD	1.7	0.20	1.5	0.29	1.4	0.31
M06	def2-SVP	SDD	3.9	0.59	3.1	0.68	3.7	0.64
HF[a]	6-311+G(d,p)	SDD	1.2	0.86	1.5	0.61	1.3	0.83

Of the methods where data is more complete, B3LYP had the lowest root mean squared error (RMSE) with respect to relative barrier heights calculated from experimental product ratios. When using a triple-ζ basis set to compute single point energies, B3LYP's RMSE fell below 1 kcal/mol. For other methods, using triple-ζ single point energies typically lowered the error by a few kcal/mol compared to the double-ζ energies. Alas, many methods' RMSE still exceeds 2 kcal/mol, with B97-D/def2-SVP/SDD "achieving" an RMSE of about 7 kcal/mol. The range of experimental product ratios only covers a range of 3.2 kcal/mol, so many methods provide wildly inaccurate predictions based on computed energies alone. That being said, most of the methods where no 'sideways' TSS has been located overestimate in favor of C_7 functionalization. Locating more pathways for C_2 activation could bring other methods in line with B3LYP.

Predicted relative barrier heights from several DFT methods correlate well with experiment. B3LYP, B3LYP-D3, and ωB97X-D all achieve an r^2 of about 0.8 or higher based on quasi-RRHO free energy. This can be compared to Chang *et al.*'s simple model based on NBO charges and Sterimol parameters that achieved an r^2 of 0.94. It seems as though a brute-force approach of looking at energies is less predictive than a simple descriptor-based model in this case.

When trying to predict more selective carboxylates, the overall trend might be less important than predicting the extremes. To see how well DFT methods predict the most selective carboxylates, we looked at the top four carboxylates that favor C_2 functionalization. The predictions for select DFT methods are in Table 3. Although many methods correctly predicted the most selecting carboxylate, some did not.

Table 3: Most selective carboxylates according to different DFT methods based on quasi-RRHO free energies computed with DFT/def2-TZVP/SDD // DFT/def2-SVP/SDD

Method	**1**	**2**	**3**	**4**
experiment	C_2F_5	CF_3	$CHCl_2$	CHF_2
B3LYP	C_2F_5	CF_3	CHF_2	$CHCl_2$
B3LYP-D3	C_2F_5	CF_3	$CHCl_2$	CHF_2
ωB97X-D	C_2F_5	CF_3	$CHCl_2$	CHF_2
B97-D	*p*-$NO_2C_6H_4$	C_2F_5	CF_3	CHF_2
BP86	*p*-$CF_3C_6H_4$	*p*-FC_6H_4	CH_2Cl	*p*-$NO_2C_6H_4$
PBE0	C_2F_5	C_6H_5	CHF_2	*p*-FC_6H_4
M06-2X	C_2F_5	CF_3	*p*-BrC_6F_4	CHF_2

B3LYP seems to be a frontrunner in terms of reliably predicting the product ratio as well as the overall trend in reactivity. To assess whether B3LYP ought to be predicting product ratios as well as it does, we computed DLPNO-CCSD(T) single point energies using the B3LYP/def2-SVP/SDD geometries. If B3LYP is getting the right product ratio for the right reason, the B3LYP energies should correlate with the DLPNO-CCSD(T) energies. Figure 8 shows a plot of DLPNO-CCSD(T) energies vs. B3LYP energies for several TSSs located for $R_2 = C_2F_5$. This is a representative example of the poor correlation between DFT energy with the higher-level DLPNO method, indicating that B3LYP is giving untrustworthy results.

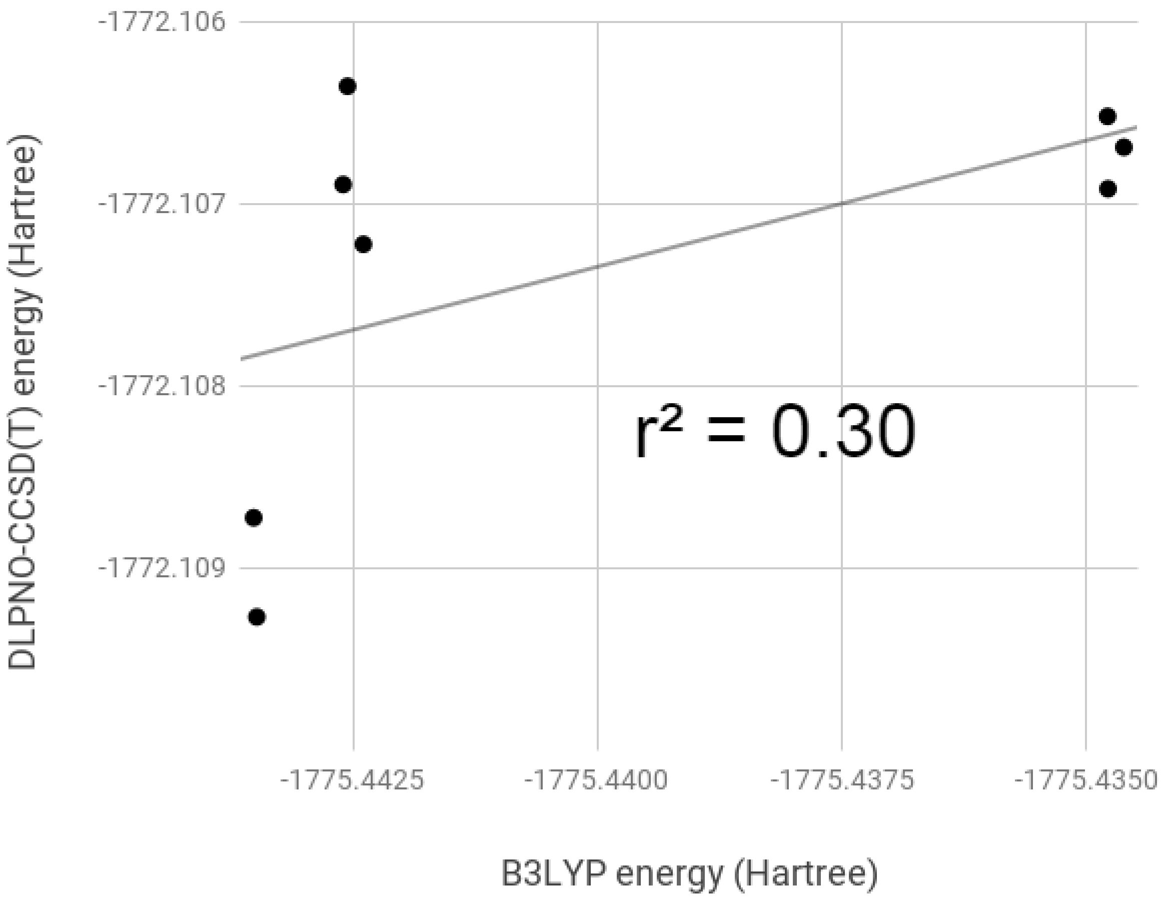


Figure 8: DLPNO-CCSD(T)/cc-pVTZ/cc-pVTZ-PP // B3LYP/def2-SVP/SDD electronic energies vs. B3LYP/def2-TZVP/SDD // B3LYP/def2-SVP/SDD energies for TSSs with pentafluoropropionate

2.6 Conclusions and Future Work

Using several popular DFT methods, over one thousand transition state structures have been located for the selectivity-determining step in an iridium-catalyzed C-H activation of indole. Multiple pathways for this reaction have been uncovered for various DFT methods. In the cases where multiple pathways have been located, the relative barrier heights of the pathways vary. The mechanism with an agostic intermediate seems to be more feasible with fluorinated carboxylates. Although two of the best-performing methods, B3LYP and ωB97X-D, did not have an agostic intermediate, comparison with DLPNO-CCSD(T) energies indicate B3LYP predicted relative energies are not trustworthy. Altogether, these results raise serious questions about how reliable DFT is for investigating reaction mechanisms. If different DFT methods give qualitatively different reaction pathways, perhaps less stock should be placed in computational mechanistic studies.

Moreover, DFT predictions for the product ratio seem quite unreliable. Few methods achieved less than 1 kcal/mol in error with respect to experiment. Although single point energies lower the error, this is not enough to make accurate product ratio predictions based on raw free energy. Instead, a correlation should be established relating the DFT energies to experimental catalyst performance. This is more likely to give a usable prediction. Although this seems to be a more reliable means of predicting catalyst performance, it usually fell short of Chang *et al.*'s simple descriptor-based model. Several DFT methods can predict either the trend in selectivity or the most selective carboxylates, however some could not.

Many scientists have turned to DFT to gain insight into their reaction. I think the assumption that DFT is a good balance between accuracy and performance should be re-evaluated. With different DFT functionals giving qualitatively different mechanisms, unreliable predictions for product ratio, and often differing trends in reactivity, the accuracy is questionable at best in this case. Although some might be satisfied with a prediction for a more selective catalyst, others want to understand their catalyst. Based on this study of a seemingly simple reaction, I do not believe DFT can give finer mechanistic details. If black-box predictions are desired, descriptor-based models may frequently outperform DFT.

More broadly, this project has highlighted some of the shortcomings of AARON. Although AARON was used to automate thousands of computations, many tasks still had to be done manually or with custom scripts. For instance, all the DLPNO-CCSD(T) and NBO computations had to be run and processed without AARON’s help. Additionally, AARON did not collate results for the different pathways. In some cases, TSS optimizations initiated from an ‘upright’ structure would converge to the same TSS as those initiated from a ‘sideways’ structure, and vice versa. Other tools had to be developed to properly weed out duplicate structures.

CHAPTER 3

EXPANDING THE CAPABILITIES OF AARONTOOLS

3.1 Abstract

The capabilities of AaronTools have been expanded considerably since the latest publication.[4] Now, AaronTools can set up and process computations from several quantum chemistry software packages: Gaussian, ORCA, Psi4, Q-Chem, xTB, and SQM. Computed vibrational frequency or excited state data can be used to produce a simulated IR or UV/vis spectrum, including spectra arising from an ensemble of conformations. AaronTools can also analyze structures to produce several popular steric descriptors that are useful for predicting reaction outcomes or catalyst performance. These new features make AaronTools a more well-rounded tool for facilitating computational chemistry tasks.

3.2 Introduction

AaronTools began as a Perl module for facilitating the manifold computations required to accurately predict the performance of asymmetric organocatalysts and organometallic catalysts.[4, 26] It could swap substituents or ligands on transition metal centers, create Gaussian input files, submit computations to a cluster, and read the structures and energies from the Gaussian output file. This, as well as several other structure modification and analysis features, make it well suited for automating many tasks. However, it has its limitations. AaronTools was not amenable to further development, meaning new features would take a considerable amount of time to implement. Furthermore, Perl has fallen out of popularity, so modern-day computational chemists are less likely to know this arcane programming language (compared to programmers of antiquity). While the inclusion of command line scripts did allow users to access much of the functionality of AaronTools without learning Perl, adding new features has been a major challenge for AaronTools developers due to the structure of the original AaronTools code.

Recently, Victoria Ingman reimplemented most of the features of Perl AaronTools in Python.[27] Python is generally considered to be easy to learn due to its syntax and error tracing. The Python implementation of AaronTools retained the structure editing and analysis features of the original AaronTools, as well as the ability to set up and process Gaussian computations. Although the feature set was largely the same, this would soon change. The basic structure of the code was reinvented in a way that made it much easier to add new features.

Many people run computations for reasons other than just computing energies or bond lengths. They may wish to simulate spectra or calculate molecular descriptors in order to build quantitative structure reactivity relationships (QSRRs). AaronTools lacked the ability to facilitate any part of this other than running the quantum mechanical computations. Consequently, some other software tool would have to be used (or created) to handle other important tasks. Many AaronTools scripts would also practically have to be used in tandem with a graphical interface, as the command line scripts or subroutines require the user to know, for example, the indices of a pair of atoms to calculate the distance between them.

Finally, AaronTools only worked with Gaussian 09, which lacks some useful methods that are in other electronic structure packages. For example, ORCA has the DLPNO coupled cluster methods, which can provide high-quality energies without the steep scaling of standard coupled cluster methods.[28] Being restricted to Gaussian limited the utility of AaronTools for many computational chemists. I wanted to address these shortcomings to make AaronTools a more well-rounded software package.

3.3 Working with More Quantum Chemistry Software

Previously, AaronTools could write Gaussian input files and read Gaussian output files. I have expanded the types of files AaronTools can create and process to include ORCA, Psi4, Q-Chem, and SQM. Creating input files is primarily handled by the new `AaronTools.theory` module. While most quantum chemistry packages provide the same overall set of methods and options, these are requested very differently within the corresponding input files. The `AaronTools.theory` module contains classes for storing most of the major settings in a software-independent manner, as described below. Many of these objects do not need to be created explicitly, as AaronTools will create them automatically when instantiating a `Theory` object or changing one of its attributes.

3.3.1 Method

The `Method` object is used to keep method keywords the same across different input file formats. As an example:

```
from AaronTools.theory import Method

pbe0 = Method("PBE0")
```

The PBE0 functional[43,44] is referred to differently by different software packages, but this is all handled automatically by `Method`. For example, when used to create a Gaussian input file, the `pbe0 Method` will use the correct keyword for Gaussian (PBE1PBE). For semiempirical methods, a particular basis set is integral to that method, so most software packages do not require the basis set to be specified to deploy a semiempirical method. As such, the `is_semiempirical` keyword argument can be passed when instantiating a `Method`.

There is also a `Method` subclass called `SAPTMethod`, which is designed for making Psi4 input files using a SAPT method (*e.g.*, SAPT0).[29] This allows users to define monomers for SAPT computations by adding them to the `components` attribute of a `Geometry` object. The charge and multiplicity of each component when using a `SAPTMethod` should be a list of integers. The first item in the list should be the total charge or multiplicity, and subsequent entries correspond to the different monomers.

3.3.2 Basis Sets

A BasisSet is a collection of Basis and ECP objects. This allows for the use of a split basis set, as well as the use of auxiliary basis sets. For example:

```
basis = BasisSet(
    [
        Basis("cc-pVTZ", [“C”, “H”, “N”, “O”]),
        Basis("cc-pVTZ", [“C”, “H”, “N”, “O”], aux_type='C'),
        Basis("cc-pVTZ-PP", [“Ir”]),
        Basis("cc-pVTZ-PP", [“Ir”], aux_type='C')
    ],
    [ECP("SK-MCDHF-RSC")]
)
```

This will use cc-pVTZ and cc-pVTZ/C for C, H, N, and O, and use cc-pVTZ-PP and cc-pVTZ-PP/C on any iridium atoms. The SK-MCDHF-RSC effective core potential will automatically be applied to transition metals.

3.3.3 Empirical Dispersion

The `EmpiricalDispersion` class is used to specify dispersion corrections.[30-32] The keywords to invoke a particular dispersion correction vary across the different software packages that AaronTools can work with. To create an `EmpiricalDispersion` instance, simply supply any of the common names or keywords for the desired dispersion correction. For example,

```
disp = EmpiricalDispersion("Grimme D2")

disp = EmpiricalDispersion("GD2")

disp = EmpiricalDispersion("D2")

disp = EmpiricalDispersion("-D2")
```

Using any of the above will result in the correct dispersion correction being requested when the input file is written.

3.3.4 Integration Grid

As with other objects in the `AaronTools.theory` package, the `IntegrationGrid` object provides a way to specify grids in a similar manner across different file formats. In some cases, the grid can only be approximated, because different software packages use different pruning schemes or integration methods to accelerate DFT computations. An `IntegrationGrid` can be instantiated with a keyword or a string with the number of radial and angular points:

```
grid = IntegrationGrid(“SuperFineGrid”)

grid = IntegrationGrid(“(99, 590)”)
```

3.3.5 Job Types

The `JobType` class is used to set up common job types. For example, use `OptimizationJob` to set up a geometry optimization or `FrequencyJob` to compute normal vibrational modes. For geometry optimizations, users can request a transition state optimization or a constrained optimization. Constrained optimizations can lock atoms, bond lengths, angles, or dihedrals in place. Atoms can also be constrained to stay in the same plane.

3.3.6 Implicit Solvent

`ImplicitSolvent` stores information about which implicit solvent model and solvent to use. When creating an input file, it will also check to ensure that the specified solvent is available in that software package.

3.3.7 Everything Else

Settings for which AaronTools does not have a dedicated class can still be modified. However, the user must be cognizant of the software they will use, as AaronTools does not know how to convert these settings. These settings can be applied by passing a dictionary or list to a particular keyword when writing an input file. Table 4 enumerates these keywords for different software packages.

Table 4: Keywords for specifying miscellaneous when creating input files for different software packages

Gaussian		
Keyword	**Description**	**List or dictionary**
link0	Link0 settings	Dictionary
route	Route line	Dictionary
end_of_file	After molecule specification	List
ORCA		
Keyword	**Description**	**List or dictionary**
simple	Simple input line	list

blocks	Various block sections	Dictionary
Psi4		
Keyword	**Description**	**List or dictionary**
settings	Global settings	Dictionary
before_molecule	Lines above the molecule	List
before_job	Lines between job and molecule	List
job	Arguments passed to job subroutines	Dictionary
after_job	Lines after the job	List
optking	OPTKing settings	Dictionary
pcm_solvent	PCM settings	Either
SQM		
Keyword	**Description**	**List or dictionary**
qmmm	qmmm namelist	Dictionary
Q-Chem		
Keyword	**Description**	**List or dictionary**
rem	General settings	Dictionary
section	Other settings	Dictionary
xTB		
Keyword	**Description**	**List or dictionary**
xcontrol	Blocks in the xcontrol file	Dictionary
command_line	Command line flags	Dictionary

3.3.8 Making an Input File

Input files can be created using the `write` method of the `Geometry` class. This method should receive an `outfile` argument to specify the name of the file. A `Theory` object should also be supplied. The various keywords in Table 4 may also be given. Alternatively, AaronTools users may use the `makeInput.py` command line script, which has options for the method, basis set, etc., as well as the software-specific options.

3.4 Steric Descriptors

Often, reactivity or reaction selectivity are impacted by steric interactions. In these cases, it might be possible to correlate some computed steric descriptor to reaction outcome, for example. This can reduce the complexity of optimizing a catalyst, as ones without the optimal value of the steric parameter value can be disregarded.

Perhaps the most widely used steric parameter for organometallic catalysis is the ligand cone angle. The ligand cone angle was originally formulated by Tolman, who used a to-scale physical model kit to calculate the angles between the M-L bonds and different substituents on phosphine ligands.[3] A method for calculating the 'exact' ligand cone angle has also been developed by Allen and co-workers.[33] I have implemented both of these algorithms in AaronTools. They can be calculated either using the `cone_angle` method of the `Component` class, or with the `coneAngle.py` command line script. Both of these will use the user-supplied structure.

Another popular ligand steric parameter is percent buried volume ($\%V_{bur}$).[34] This is a measure of how much space a ligand takes up in the vicinity of the reaction center. Splitting the buried volume into octants or quadrants around the center can also be useful for predicting the efficacy of a catalyst.[35] I have implemented an algorithm for calculating $\%V_{bur}$ in AaronTools that uses Monte-Carlo integration. The $\%V_{bur}$ can also be split into octants or quadrants. The `percentBuriedVolume.py` command line script can be used to calculate $\%V_{bur}$.

Finally, Sterimol parameters can be used to measure the size of a substituent that goes beyond more classical steric parameters.[36] Sterimol parameters quantify the length of the substituent, as well as its minimum and maximum widths. Sterimol parameters for substituents can be calculated with the `substituentSterimol.py` command line script. Sterimol parameters have also been defined for ligands.[2] Due to differences in calculating the length parameter, the `ligandSterimol.py` script should be used for ligands.

3.5 Simulating Spectra

AaronTools can now produce simulated IR and UV/vis spectra of a molecule using data parsed from normal mode or excited state computations. If optical activity is computed for the various vibrations or excited states, a vibrational or electronic circular dichroism spectrum can be generated. Single molecule IR and UV/vis spectra can be produced with the plotIR.py and plotUVVis.py command line scripts, respectively. In solution, many conformers of a molecule typically contribute to the overall spectrum. AaronTools also has `plotAverageIR.py` and `plotAverageUVVis.py` command line scripts to produce spectra based on a Boltzmann population of conformers. For example, see Figure 9.

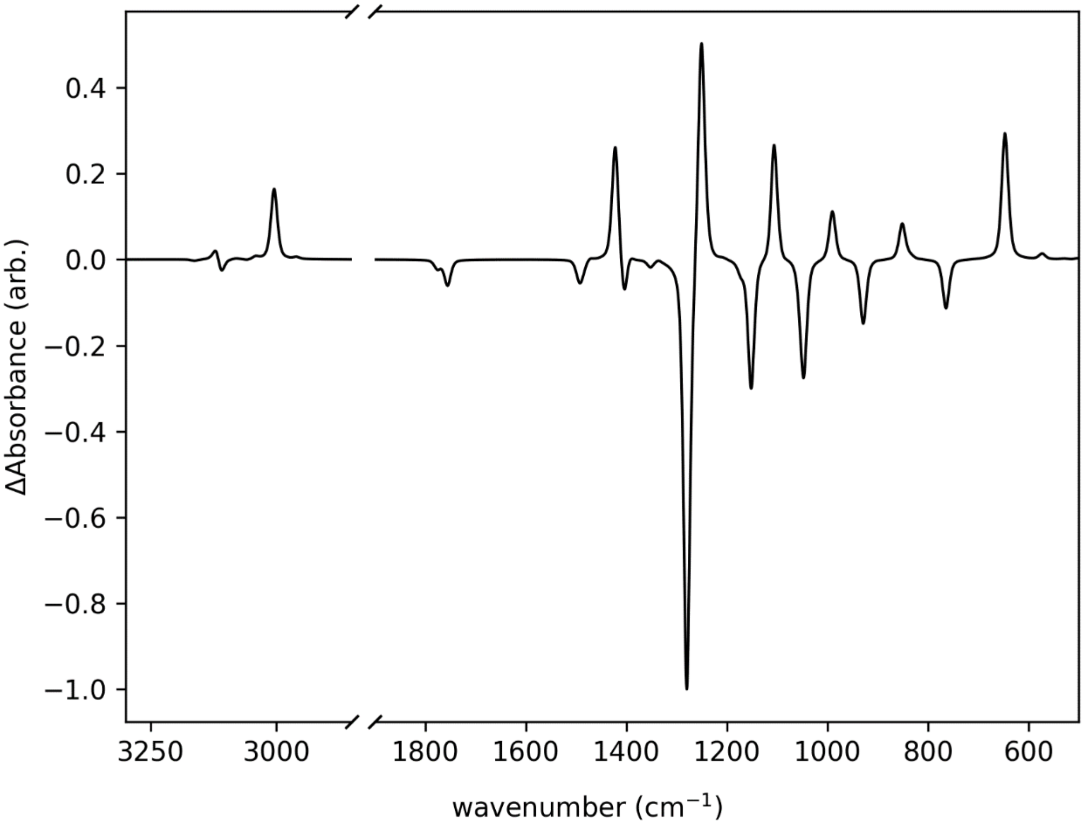


Figure 9: Computed VCD spectrum of methyl lactate

3.6 Orbitals

Molecular orbitals can be parsed from Formatted Checkpoint (FChk) files and ORCA output files. Other orbital data can also be parsed from NBO files. The orbital functions can be evaluated with the `printCube.py` command line script and printed to a Gaussian cube file, which can be opened in many graphical programs to visualize the orbital. This command line script can also print the SCF electron density and orbital-weighted Fukui functions[37] to a cube file. Fukui functions can be integrated in the volume around each atom to produce a value that indicates the atom's electrophilicity or nucleophilicity.

3.7 Structure Building

Building structures in a 3D molecule editor can be tedious, particularly for complex structures. AaronTools has a fetchMolecule.py command line script that takes an IUPAC name or SMILES and prints the 3D coordinates of a molecule. IUPAC names are converted to SMILES using the OPSIN web API,[38] and 3D structures are generated from SMILES using the NCI/CADD SMILES translator[4] or the RDKit Python module.[39]

Coordination complexes can be generated using the `getCoordinationComplexes.py` command line script. This script takes a list of ligands, a central atom, and a coordination geometry (square planar, octahedral, *etc*.), and produces all symmetry-unique coordination complexes using the predetermined coordination geometries from Simas *et al*.[40] For example, Figure 10 shows the two coordination complexes that are generated for a square planar coordination geometry with a chloride, carbonyl, and a Quinox ligand.

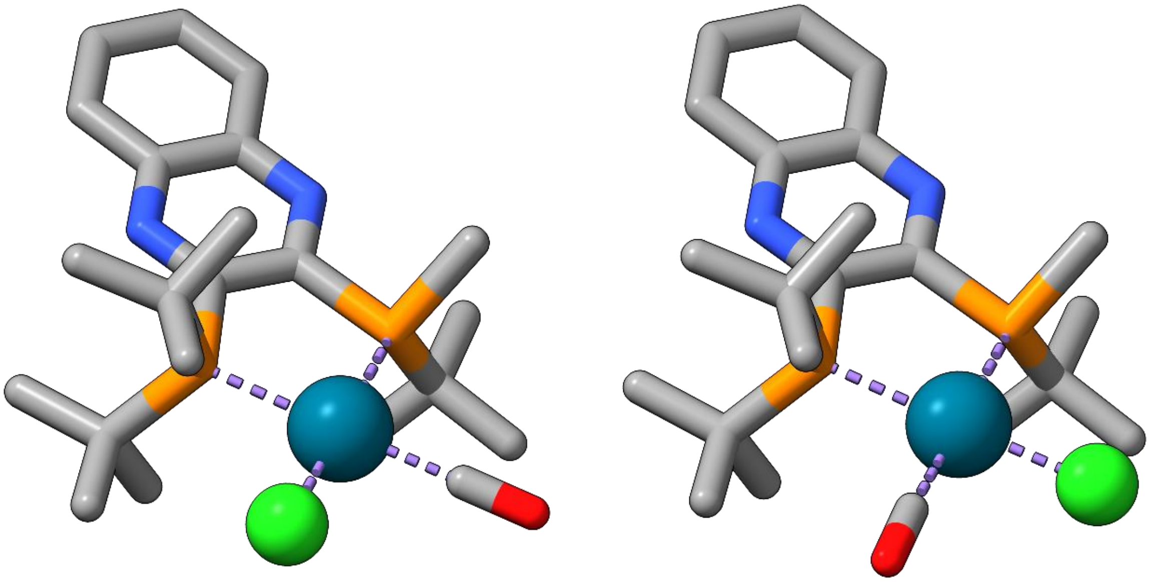


Figure 10: Two coordination complexes generated for a square planar coordination geometry with a chloride, carbonyl, and Quinox ligand

3.8 Finders

The AaronTools `Geometry` class has a `find` subroutine that makes it easy to retrieve atoms based on serial number or element. However, the name and serial number of an atom typically requires using this in tandem with a GUI, and it is common for a molecule to have multiple atoms with the same element. This makes it difficult to locate a specific atom, which would be useful for many AaronTools subroutines or command line scripts. For this reason, I have made the `AaronTools.finders` module. This contains several objects for locating atoms based on certain characteristics of the atom using the `Geometry.find` subroutine.

Table *5* contains a list of different finders. The `findAtoms.py` command line script can be used to print the indices of atoms that match the user-specified criteria.

Table 5: AaronTools finder classes and their description

Finder class	Description
`BondsFrom`	Atoms a certain number of bonds from another atom
`WithinBondsOf`	Atoms within a certain number of bonds of another atom
`BondedTo`	Atoms bonded to a particular atom
`WithinRadiusOfPoint`	Atoms near a specified point
`WithinRadiusFromArom`	Atoms near another atom
`NotAny`	Inverts the match criteria
`AnyTransitionMetal`	D block elements
`AnyNonTransitionMetal`	Non-D block elements
`HasAttribute`	Atoms with a particular attribute
`VSEPR`	Atoms with a certain VSEPR geometry
`BondedElements`	Atoms whose neighbors match the specified list of elements
`NumberOfBonds`	Atoms with a certain number of bonds
`ChiralCenters`	R or S chiral centers (configuration is not determined)
`FlaggedAtoms`	Atoms with the `flag` attribute set
`CloserTo`	Atoms closer to one of the specified atoms than the other
`AmideCarbon`	Carbon atoms in an amide functional group
`Bridgehead`	Atoms in more than one ring
`SpiroCenters`	Atoms in two ring systems that have no other common atoms

3.9 Conclusions

AaronTools is a versatile toolkit for facilitating quantum chemistry workflows using several popular electronic structure packages. Its capabilities have been expanded considerably, allowing for the processing of computed data and calculation of several widely used steric descriptors. Now AaronTools can be used to not only run computations, but also create simulated spectra or generate data that are useful for establishing quantitative structure-reactivity relationships.

CHAPTER 4

SEQCROW: A CHIMERAX BUNDLE TO FACILITATE QUANTUM CHEMICAL APPLICATIONS TO COMPLEX MOLECULAR SYSTEMS [a]

[a] A. J. Schaefer, V. M. Ingman, and S. E. Wheeler, J. Comp. Chem. 42, 1750 (2021).
Reprinted here with permission of the publisher

4.1 Abstract

We describe a bundle for UCSF ChimeraX called SEQCROW that provides advanced structure editing capabilities and quantum chemistry utilities designed for complex organic and organometallic compounds. SEQCROW includes graphical presets and bond editing tools that facilitate the generation of publication-quality molecular structure figures while also allowing users to build molecular structures quickly and efficiently by converting 2-D molecular figures into 3-D structures, mapping new ligands onto existing organometallic complexes, and adding rings and substituents to molecular structures. Other capabilities include the ability to visualize vibrational modes and simulate IR spectra, to compute and visualize molecular descriptors including percent buried volume and Sterimol parameters, to process thermochemical corrections from quantum mechanical computations, to generate input files for ORCA, Psi4, Q-Chem, SQM, xTB, and Gaussian, and to run and manage computational jobs.

4.2 Introduction

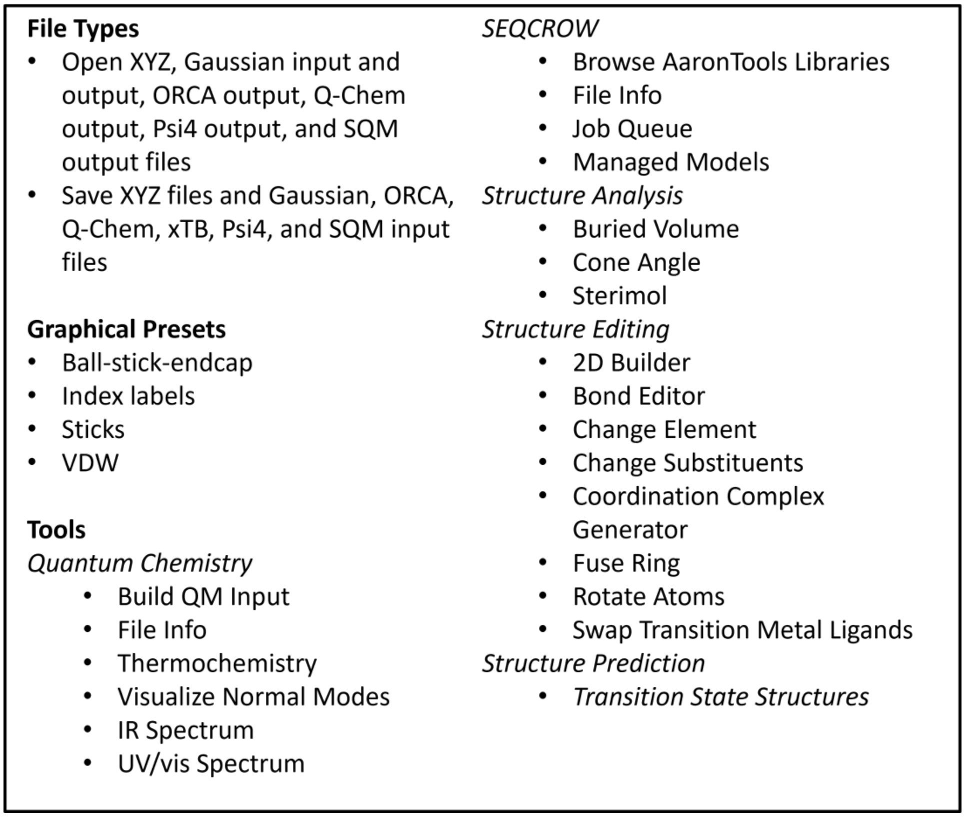
Computational quantum chemistry is playing an increasingly important role in studies of organic and organometallic systems, most notably those involved in homogeneous catalysis. Such studies can provide key insights into the origins of reactivity and selectivity and can aid in the identification and development of new reactions and improved catalysts.[22, 41-44] While methodological and hardware advances have opened the door for accurate quantum chemical studies of larger and larger molecules, graphical utilities for building and analyzing complex molecular structures and visualizing results have not kept pace.

For example, a common task in modern quantum chemistry applications to organometallic systems is the replacement of a given ligand on a metal center with other complex (and often chiral) ligands for each stationary point along a multi-step reaction pathway. In this way key barrier heights can be compared across multiple examples of a given reaction to unveil trends in reactivity and selectivity. However, popular graphical molecular builders (*e.g.*, GaussView/AGUI,[45] Avogadro,[46] IQmol,[47] *etc*.) are not well-suited for this task. First, none of these tools provide libraries of common modern ligands (e.g., BINAP, SEGPHOS, *etc*.), instead requiring the user to build these ligands from scratch or extract them from previously computed structures. Moreover, replacing a ligand in a structure using these graphical builders requires the user to first delete the old ligand and then manually place the new ligand appropriately. This process can be a tedious and error-prone when performed across dozens of stationary points along a reaction, particularly if the process must be repeated for many different ligands. Apart from mapping new ligands, other common molecular building and visualization tasks can be surprisingly cumbersome in many graphical user interfaces.

Herein, we describe SEQCROW, a free, open-source bundle for UCSF ChimeraX[48] that provides powerful graphical tools for building and manipulating the types of molecular structures encountered in modern chemical applications, representing results from quantum chemistry computations, and preparing and running diverse input files for popular quantum chemistry software packages.

4.3 SEQCROW

SEQCROW is an add-on to ChimeraX that provides new commands, atom selectors, graphical presets, and tools across many categories (see Figure 11). This includes reading structures, vibrational frequencies/normal modes, and other data from Gaussian,[49] ORCA,[25] Q-Chem,[50] or Psi4[51] output files. Much of the added functionality is available through both graphical menus and the ChimeraX command line. SEQCROW is part of a collection of tools for quantum chemistry called QChASM (Quantum Chemistry Automation and Structure Manipulation)[52] and uses the AaronTools Python package[4, 52] for structure manipulation and analysis, preparation of quantum chemistry input files, and processing of quantum chemistry output files. We highlight several features of SEQCROW below. Additional information and help are available through the built-in help browser of ChimeraX and the GitHub page for SEQCROW.[53]



File Types
- Open XYZ, Gaussian input and output, ORCA output, Q-Chem output, Psi4 output, and SQM output files
- Save XYZ files and Gaussian, ORCA, Q-Chem, xTB, Psi4, and SQM input files

Graphical Presets
- Ball-stick-endcap
- Index labels
- Sticks
- VDW

Tools

Quantum Chemistry
- Build QM Input
- File Info
- Thermochemistry
- Visualize Normal Modes
- IR Spectrum
- UV/vis Spectrum

SEQCROW
- Browse AaronTools Libraries
- File Info
- Job Queue
- Managed Models

Structure Analysis
- Buried Volume
- Cone Angle
- Sterimol

Structure Editing
- 2D Builder
- Bond Editor
- Change Element
- Change Substituents
- Coordination Complex Generator
- Fuse Ring
- Rotate Atoms
- Swap Transition Metal Ligands

Structure Prediction
- *Transition State Structures*

Figure 11: Overview of features available in SEQCROW

4.4 Graphical Presets, Bond Editor, and Selectors

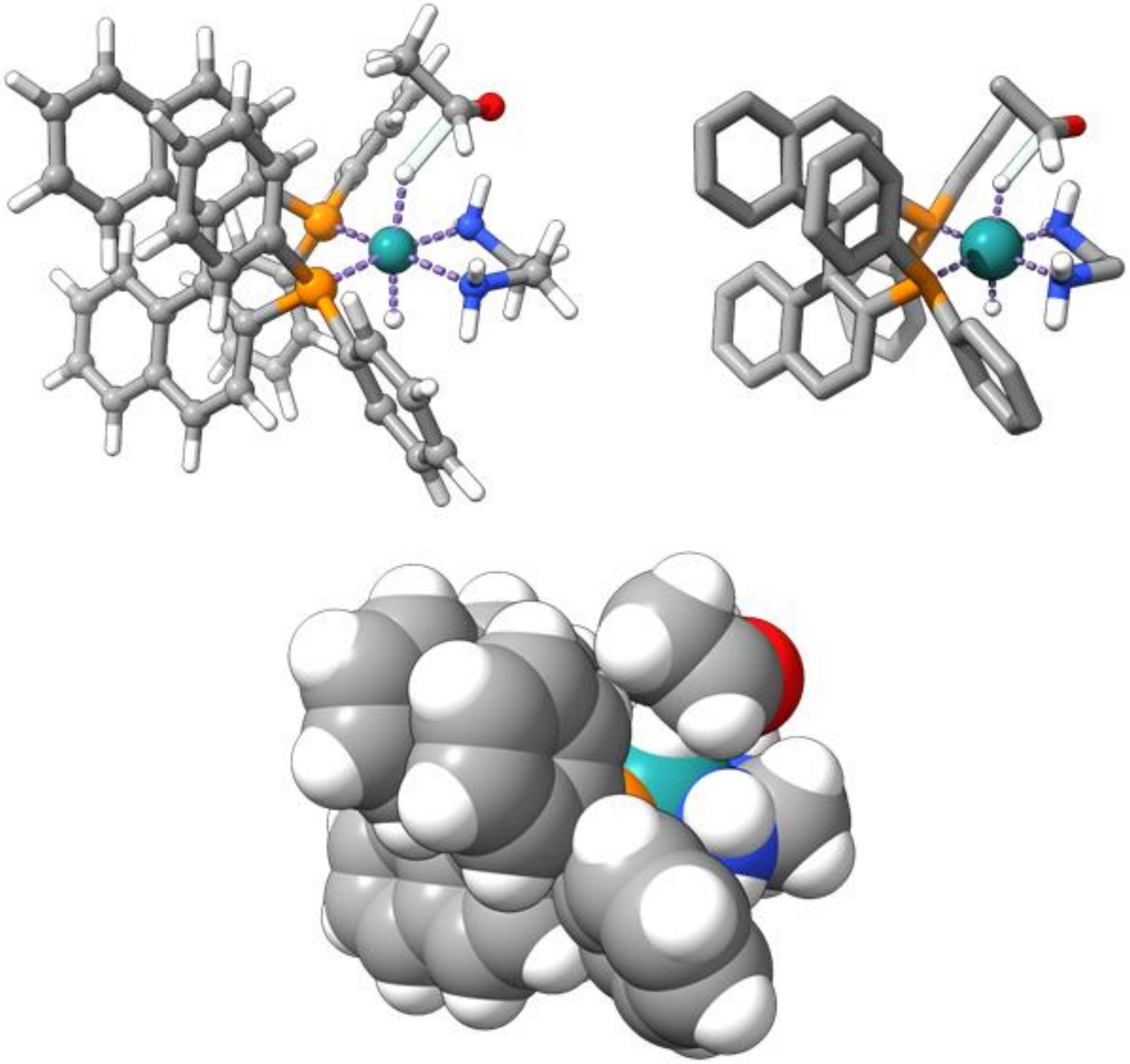


Figure 12: TSS for a Rh-catalyzed asymmetric hydrogenation of acetaldehyde represented with different graphical presets from SEQCROW: ball-stick-endcap, sticks, and VDW

Generation of illuminating visual representations of molecular structures is vital to effective communication of computational results. This is particularly important for transition state (TS) structures, in which one must convey the breaking and forming bonds. SEQCROW provides publication-quality presets for ball-and-stick, stick, and VDW representations of molecular structures. This includes customizable semi-transparent bonds to represent forming and breaking bonds in TS structures as well as bond types for non-covalent interactions and coordination bonds (see Figure 12). TS bonds are automatically detected and displayed when opening output files that contain imaginary vibrational frequencies, and a Bond Editor tool allows the user to create or erase each of these bond types as needed.

While ChimeraX natively provides selectors for structural components, these are primarily aimed at biomolecules. SEQCROW adds additional selectors for more general components of organic and organometallic molecules. For example, using SEQCROW one can select all transition metal atoms or instances of a given substituent type by name (Ad, Ph, Me, *etc.*) within a molecule or set of molecules, which can then be highlighted or modified. SEQCROW also adds selectors for atoms with certain VSEPR shapes, such as tetrahedral or trigonal pyramidal atoms. In order to more easily select a molecular fragment, SEQCROW adds a ’connected’ selector, which expands the current selection to anything on the same fragment. This ‘connected’ selector is also available as a mouse mode, allowing for a fragment to be selected with one click.

4.5 Structure Modification and Building

A key benefit of SEQCROW over many other graphical molecular builders is a series of unique structure modification tools. This includes the ability to quickly modify structures by adding substituents and rings or mapping new ligands onto existing molecular structures. Built-in libraries provide common (and not so common) substituents, ring types, as well as chiral and achiral ligands. SEQCROW allows users to browse these libraries or add their own, custom groups to these libraries. Multiple structures can be edited simultaneously, allowing, for example, derivatives of all stationary points along a reaction pathway to be generated. Users can also rotate entire molecules or selected atoms about any defined centroid and axis (including the axis normal to a set of atoms) using the Rotate Atoms tool.

Building structures atom-by-atom in 3D can be finicky and tedious, particularly in the cases of chiral molecules and structures with fused rings. Users may find it easier to use SEQCROW's 2D Builder tool. With this tool, a user can sketch a molecule in a ChemDoodle[54] 2D window, or import a MOLFile or ChemDoodle JSON file, and load a 3D structure of this molecule into ChimeraX (see Figure 13). The 3D structures are obtained from an NCI/CADD web API. This generally works well for organic molecules, but it is less reliable for organometallic complexes. As an alternative, SEQCROW has a Coordination Complex Generator tool. With this tool, the user specifies the element of the central atom, the coordination geometry, and some mono- or bidentate ligands from the ligand or substituent library. The tool will then generate all symmetry-unique coordination complexes using a database of unique coordination geometries determined by Simas *et al.*[40]

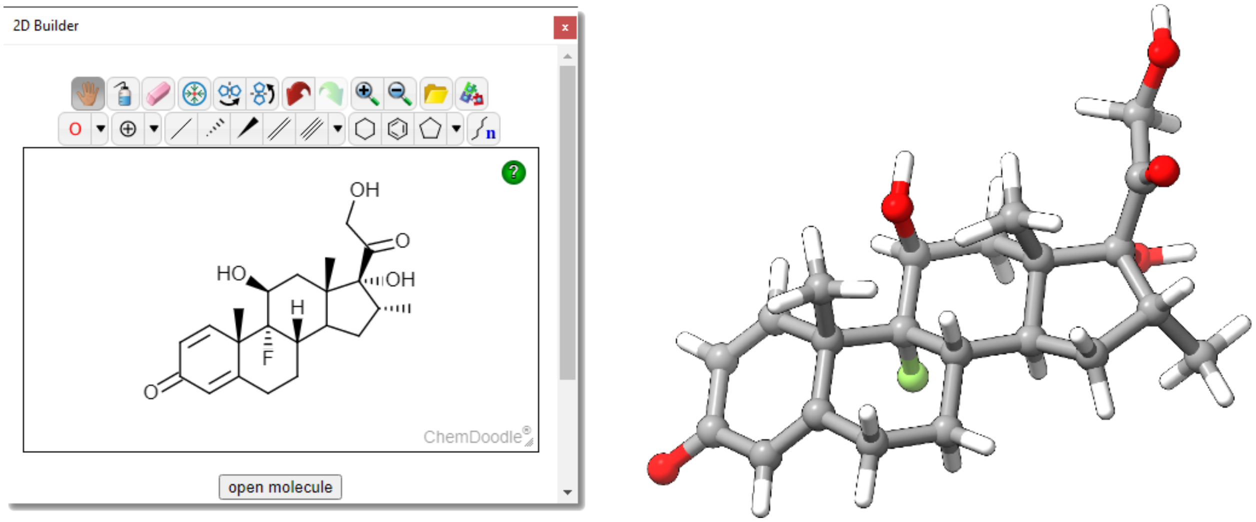


Figure 13: 2D sketch of dexamethasone and the resulting 3D structure

4.6 Setting up and Running Computations

SEQCROW provides a general interface for building input files for QM computations. Unlike many graphical interfaces that are either specific to particular quantum chemistry packages or limit the possible computation types, SEQCROW's QM Input Builder allows the generation of Gaussian,[49] ORCA,[25] Psi4[51], Q-Chem[50], xTB[55], and SQM[56] input files for nearly any type of QM computation. This includes routine job types (geometry optimizations, vibrational frequencies, *etc*.) but also more complex job types including constrained geometry optimizations, calculations with mixed basis sets/ECPs, as well as SAPT and F-SAPT computations.[29, 57-60]

The QM Input Builder will check for typos in user-entered methods and basis sets and offer suggestions if one of these is misspelled. The QM input builder will also check for other simple issues with the input, such as some elements not being included in a specified basis set. Some issues will be silently resolved. For example, the keyword to use dichloromethane as the CPCM implicit solvent in ORCA is "CH2Cl2". If the user enters "dichloromethane", the correct keyword will be used instead. Issues are only fixed silently in cases where it is clear what the user is requesting.

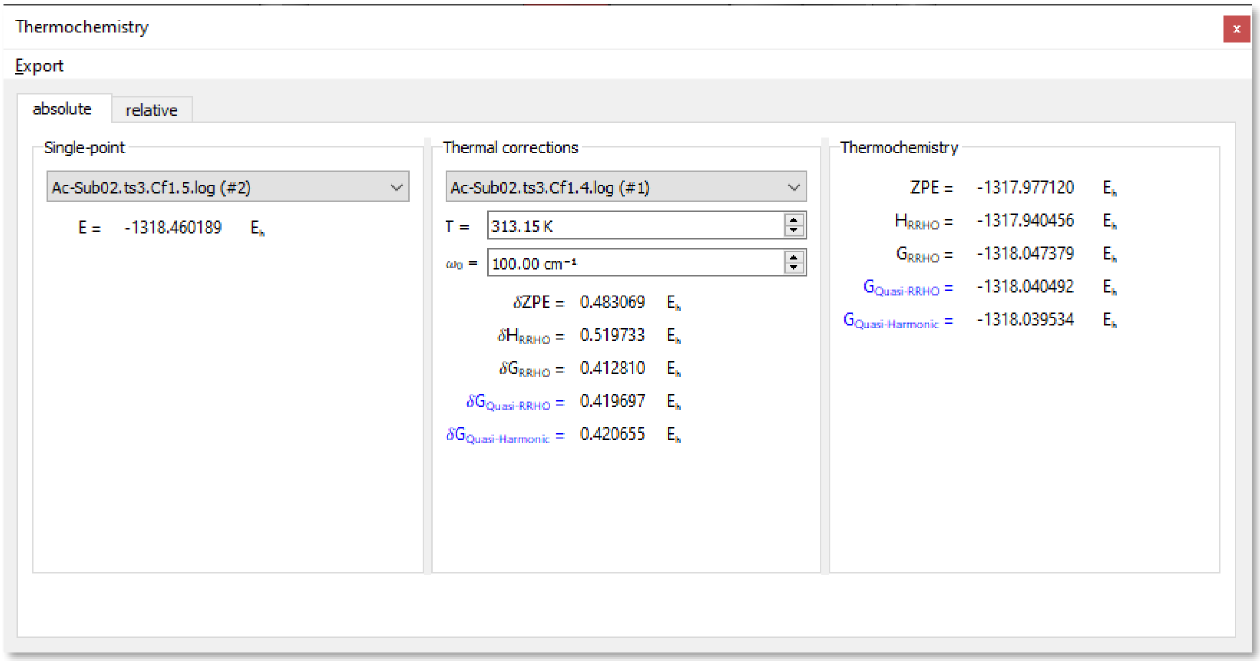
Custom job presets can be saved and later retrieved to streamline the generation of input files. These presets can also be exported and sent to other users. For example, this provides a simple way for an experienced computational chemist to assist less experienced users build complex input files. Python classes enable developers to easily add additional file formats to SEQCROW, allowing for the deployment of generalized input file builders for any desired QM code.

In addition to the general input file builder, SEQCROW has a separate tool for setting up automated transition state searches. This allows users to easily utilize TSS search algorithms included in certain quantum chemistry software packages (*i.e.*, nudged elastic band in ORCA, synchronous-transit guided quasi-Newton in Gaussian, freezing string in Q-Chem, and a metadynamics-based algorithm in xTB). These algorithms take the structures of the reactant and product as input, and the order of the atoms must be the same in both. The Transition State Structures tool allows users to swap the order of the atoms, as well as modify some algorithm-specific parameters.

Jobs can be run on the local machine or local computing cluster if the corresponding quantum chemistry software is installed. A built-in queue enables the user to monitor the progress of multiple jobs and automatically retrieve the output from completed or ongoing computations. From the queue, users can restart failed computations, with automatic error mitigation applied for common errors (*e.g.*, running out of SCF iterations or geometry optimization steps). Planned extensions of SEQCROW will allow for job submission and job management on remote clusters as well as the setup and execution of automated workflows using AaronJr (Automate Any Reaction or Optimization, Normally Just right), which is a QM workflow manager currently in development.[52, 61]

4.7 Processing Computed Data

Another unique feature among graphical molecular visualization and analysis tools is the ability of SEQCROW to process thermochemical corrections by combining data from multiple quantum chemistry computations (see Figure 14). SEQCROW's Thermochemistry tool allows the user to combine single point energies from Gaussian,[49] ORCA,[25], Q-Chem[50] or Psi4[51] with thermal corrections (including not only RRHO free energies but also those relying on the quasi-harmonic and quasi-RRHO approximations)[24, 62] from any output file containing vibrational frequency data. These values can be computed at any requested temperature or using any cutoff value for the quasi-RRHO and quasi-harmonic treatments of the entropic component of the free energy. The Thermochemistry tool is also able to calculate these quantities for one ensemble of conformers relative to another.



Single-point

Ac-Sub02.ts3.Cf1.5.log (#2)

$E =$ -1318.460189 E_h

Thermal corrections

Ac-Sub02.ts3.Cf1.4.log (#1)

$T =$ 313.15 K

$\omega_0 =$ 100.00 cm^{-1}

$\delta ZPE =$ 0.483069 E_h

$\delta H_{RRHO} =$ 0.519733 E_h

$\delta G_{RRHO} =$ 0.412810 E_h

$\delta G_{Quasi\text{-}RRHO} =$ 0.419697 E_h

$\delta G_{Quasi\text{-}Harmonic} =$ 0.420655 E_h

Thermochemistry

$ZPE =$ -1317.977120 E_h

$H_{RRHO} =$ -1317.940456 E_h

$G_{RRHO} =$ -1318.047379 E_h

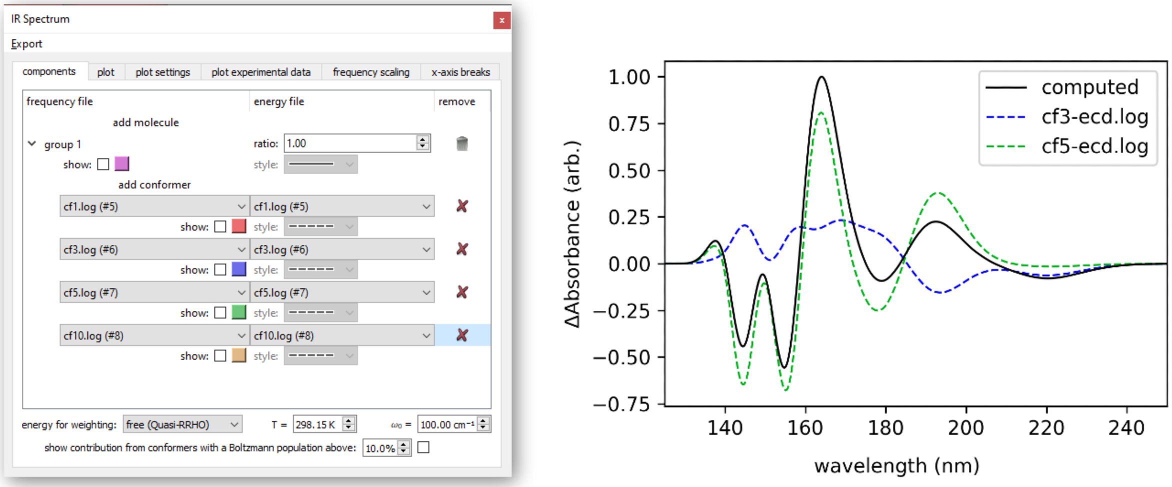
$G_{Quasi\text{-}RRHO} =$ -1318.040492 E_h

$G_{Quasi\text{-}Harmonic} =$ -1318.039534 E_h

Figure 14: Energies, enthalpies, and free energies (RRHO, quasi-RRHO, and quasi-harmonic) can be evaluated by combining energies from different QM packages at any specified temperature

SEQCROW provides a simple yet customizable interface to visualize normal vibrational modes as well as IR spectra. Normal modes can be visualized statically via displacement vectors or animated. If requested, the appropriate vibrational scaling factors for the corresponding level of theory are automatically retrieved from either the NIST Computational Chemistry Comparison and Benchmark Database (CCCBDB)[63] or the database maintained by the Chemical Theory Center at the University of Minnesota (UMN CTC).[64, 65] An additional tool allows users to determine bespoke scaling factors based on a least-squares fitting of user-provided anharmonic vibrational frequencies.

A single conformation of a molecule is generally not responsible for all of the signals in an experimental spectrum. Thus, SEQCROW has tools for generating computed IR or UV/vis spectra from a Boltzmann distribution of conformers. These tools allow for contributions of individual conformers to be plotted separately (see Figure 15), an arbitrary number of x-axis interruptions for uninteresting portions of the spectrum, and peak locations to be shifted or scaled. To simplify comparisons, spectra can be imported from or exported to CSV files.



frequency file	energy file	remove
group 1	ratio: 1.00	
show: □	style:	
cf1.log (#5)	cf1.log (#5)	✗
show: □	style:	
cf3.log (#6)	cf3.log (#6)	✗
show: □	style:	
cf5.log (#7)	cf5.log (#7)	✗
show: □	style:	
cf10.log (#8)	cf10.log (#8)	✗
show: □	style:	

energy for weighting: free (Quasi-RRHO) T = 298.15 K ω_0 = 100.00 cm⁻¹

show contribution from conformers with a Boltzmann population above: 10.0% □

Figure 15: Choosing conformers to generate an IR spectrum, and an ECD spectrum with the most populated conformers displayed separately

Inspecting orbitals may be useful for understanding the reactivity of a molecule. SEQCROW can display orbitals from formatted checkpoint files (FChk), ORCA output files, and NBO files from the Orbital Viewer tool. This tool allows users to modify the colors of the orbital lobes and resolution at which the orbital is calculated. Additionally, the SCF electron density and orbital-weighted Fukui functions[37] can also be displayed. These can provide additional insight into the reactivity of a molecule.

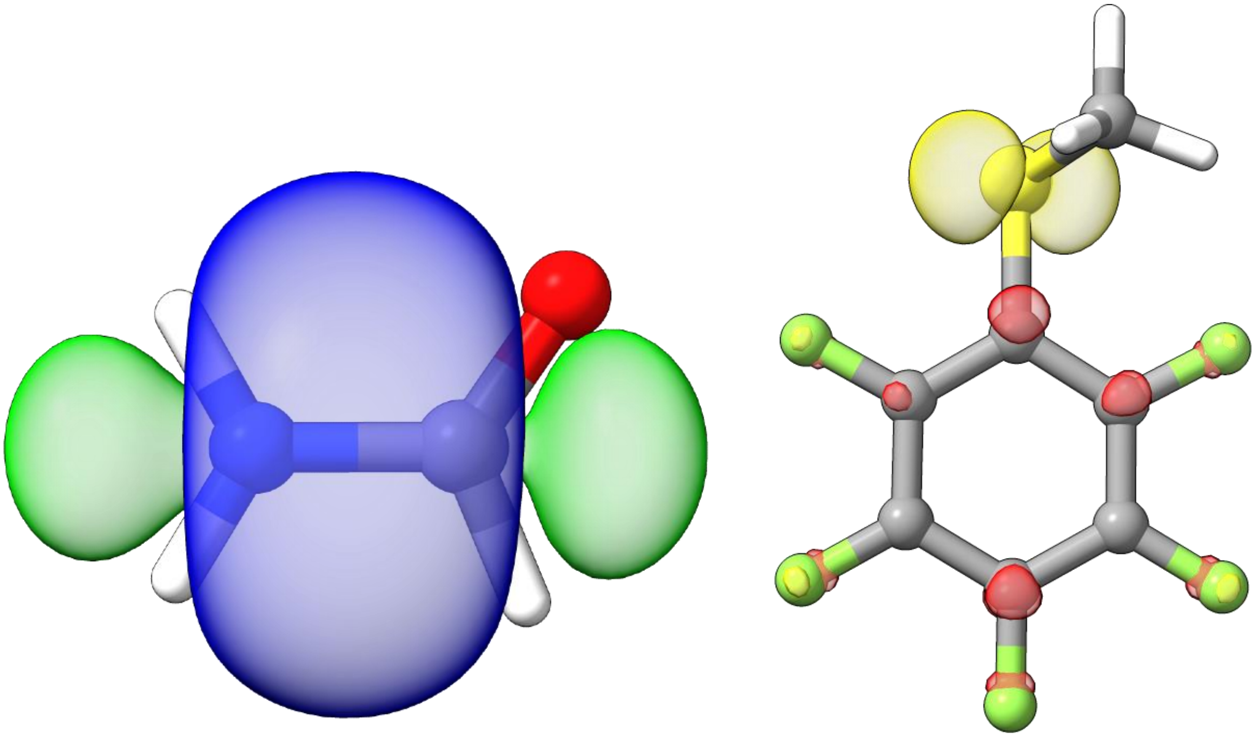


Figure 16: A bonding PNBO of formamide and the orbital-weighted Fukui dual function for pentafluorothioanisole

4.8 Structure Analysis

Vital to many modern quantum chemistry applications is the evaluation of molecular descriptors.[2, 66-68] SEQCROW provides an interface to compute and visualize three key descriptors: Sterimol parameters[69], percent buried volume (%V_{bur})[34, 70-72], and ligand cone angles.[3, 33] Sterimol provides a multidimensional measure of the steric bulk of substituents, whereas cone angles and %V_{bur} quantifies the steric crowding around a metal center or other reactive center.

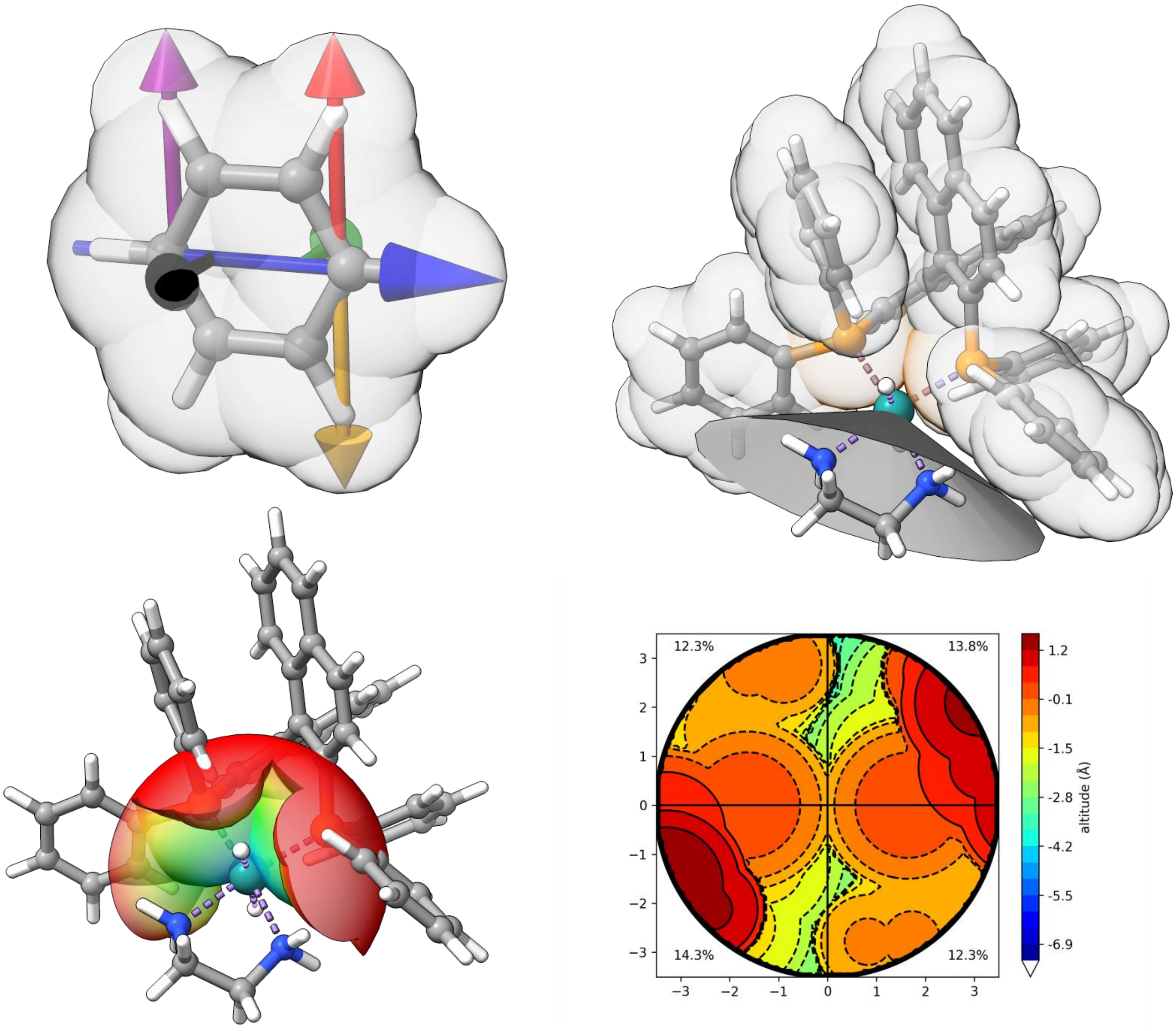


Figure 17: Sterimol parameters for a phenyl substituent, and the cone angle, buried volume, and steric map for a BINAP ligand on a rhodium coordination complex

With SEQCROW, these quantities can be evaluated and visualized in ChimeraX. For example, Figure 17 shows the evaluation and visualization of the B_1-B_5 and L Sterimol parameters for the phenyl substituent. While other tools are available for computing these parameters,[73, 74] the visualization of the corresponding vectors can facilitate the communication of key steric information in publications. Figure 17 also shows visualizations for two ligand steric parameters: cone angle and $\%V_{bur}$, as well as a steric map with the buried volume partitioned into quadrants.

These structure analysis tools can calculate these parameters for multiple structures simultaneously. This could be useful for studying a large number of different ligands/substituents or conformers. As an example, we will calculate the Sterimol parameters for 10 substituents at once. Ideally, we would use optimized structures for this, but structures generated by NCI/CADD SMILES translator. The structures we will use can be loaded into ChimeraX by running the command: `open smiles:C smiles:CC smiles:C(C)C smiles:C(C)(C)C smiles:C1=CC=CC=C1 smiles:CF smiles:C(F)F smiles:C(F)(F)F smiles:CCl smiles:C(Cl)Cl` on the ChimeraX command line.

We will say that the first hydrogen on each of these is standing in for the rest of the molecule. Thus, the ‘substituent’ portion of these can be selected by running the command: `select ~@H1` on the ChimeraX command line. Now, all atoms except one hydrogen should have a green outline. The substituents should be methyl, ethyl, isopropyl, *tert*-butyl, phenyl, fluoromethyl, difluoromethyl, trifluoromethyl, chloromethyl, and dichloromethyl. To calculate the Sterimol parameters for all of these, simply click the “calculate parameters for selected substituents” button on the Substituent Sterimol tool. All parameter values will be in the table on the tool. If the “show vectors” and “show radii” options are selected on the tool window, the structures should look like what is shown in Figure 18.

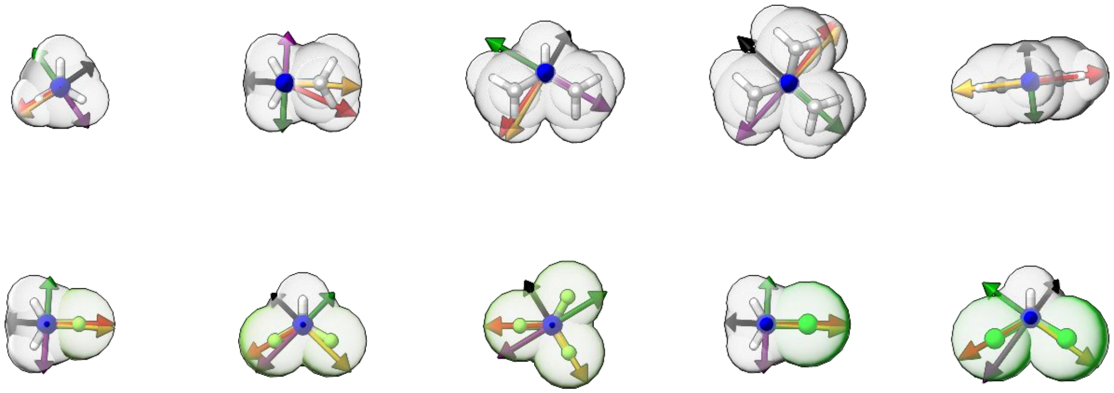


Figure 18: Vectors representing the Sterimol parameters for several substituents

4.9 Installation and Availability

SEQCROW is free and open-source and can be installed automatically through the 'More Tools' option on the 'Tools' menu in ChimeraX. Additional information and help is available through the SEQCROW GitHub page.[53]

4.10 Conclusions

As quantum chemistry applications have tended toward larger and more complex molecules, popular graphical user interfaces have not kept pace. The result is that building modern organic and organometallic molecules is often a cumbersome task. SEQCROW is a free, open-source bundle for UCSF ChimeraX[48] designed to make it easier to build, manipulate, and analyze such molecules.

The long-term goal is to develop SEQCROW into a complete graphical interface for quantum chemistry applications across different electronic structure packages. Such a tool will broaden access to these powerful tools. There has long been tension regarding the merits of making quantum chemistry more accessible to non-experts. Indeed, opening the powerful tools of quantum chemistry to those with inadequate training is a recipe for the generation and possible publication of dubious computational results. Such data, once in the literature, can dilute the impact of more rigorous computational studies, to the detriment of the field.

In our view, the prudent response to this dilemma should not be to limit access to quantum chemical tools but to facilitate training for those seeking to run computations. In this context, SEQCROW has been built with two audiences in mind. First, SEQCROW provides tools that will streamline modern quantum chemistry applications by experienced computational chemists. At the same time, SEQCROW provides access to the power of modern quantum chemistry programs without the need to learn to work on the Linux command line. Our hope is that SEQCROW will provide a platform for training those without experience in high-performance computing or computational quantum chemistry, including both experimental researchers with a casual interest in computational chemistry and undergraduate students.

CHAPTER 5

SEQCROW IMPLEMENTATION DETAILS

5.1 Abstract

In this chapter, I show how to extend SEQCROW and provide details about how some SEQCROW features are implemented. This includes adding new file formats to the QM Input Builder tool and the Transition State Structure prediction tool, as well as allowing for those jobs to be run through SEQCROW. I will also discuss how the buried volume visuals are created, and how SEQCROW works as an interface between ChimeraX and AaronTools.

5.2 Introduction

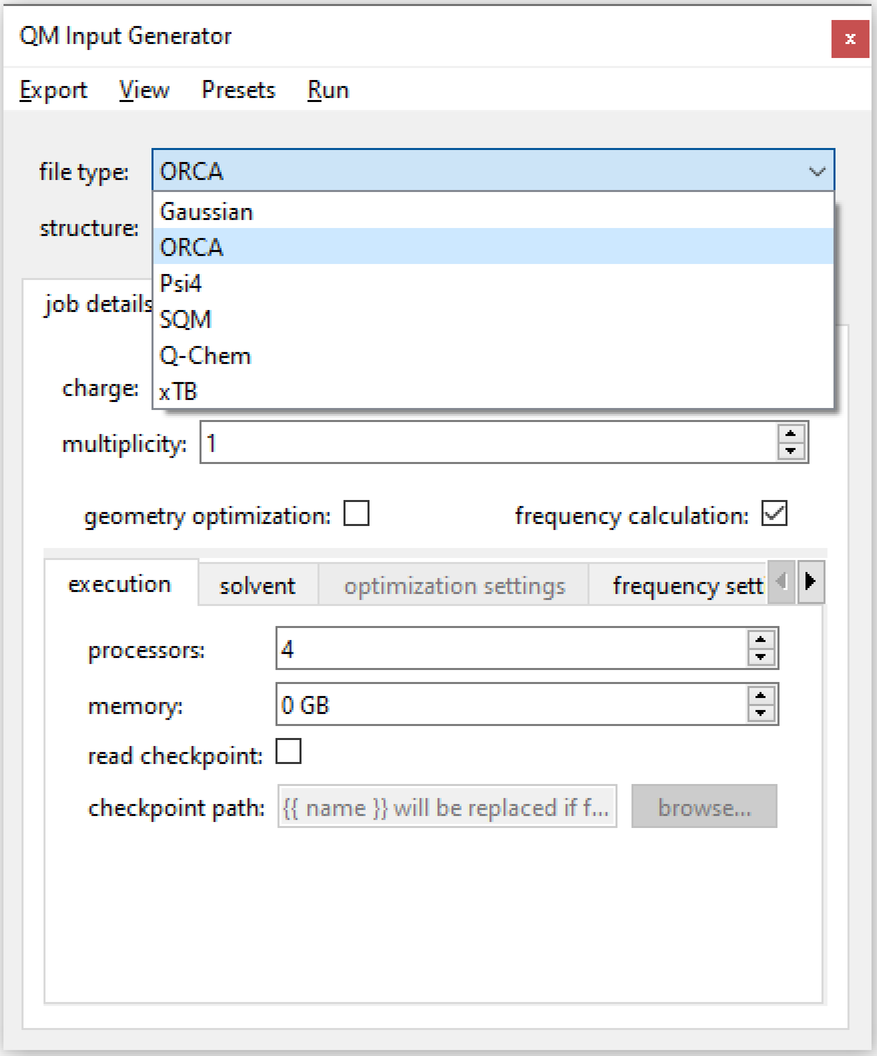
The various features of SEQCROW were developed over several months. AaronTools, [4] which does a lot of the heavy lifting for SEQCROW, has resulted from the work of multiple Wheeler group members since before SEQCROW's inception. Creating tools like SEQCROW and AaronTools takes a lot of time, thought, and code. Even so, SEQCROW is a plugin for ChimeraX, because building a standalone graphical program with all the functionality of SEQCROW would require significantly more time, thought, and code.

In an effort to ease the creation of GUI-based tools, SEQCROW is built so that other ChimeraX plugins can utilize various SEQCROW interfaces. As mentioned in the previous chapter, new formats can be added to the QM Input Builder tool. Similarly, algorithms can be added to the Transition State Structure prediction tool. For both tools, plugin developers may choose to write additional code allowing their new file formats to be executed through the ChimeraX GUI.

In this chapter, I will describe how to add new input file types and jobs to SEQCROW. I will also describe other facets of SEQCROW that may be of interest to someone looking to make visualizations similar to SEQCROW's. In particular, I will describe the algorithm for generating 3D %V_{bur} visuals and how SEQCROW acts as a middleman between ChimeraX and AaronTools. I will assume the reader is familiar with the basics of ChimeraX bundle development, Python 3, and AaronTools.

5.3 Adding Formats to the QM Input Builder Tool

Input file formats are tracked using ChimeraX's `ProviderManager` structure. A plugin must add entries in the bundle_info.xml file for the QM input file formats it adds. SEQCROW's manager for this is called "`seqcrow_qm_input_manager`". The names of the providers will be used as the label for the input file format on the QM Input Builder tool, shown in Figure 19.



file type: ORCA

multiplicity: 1

geometry optimization: □ frequency calculation: ☑

processors: 4

memory: 0 GB

read checkpoint: □

checkpoint path: {{ name }} will be replaced if f...

Figure 19: SECQROW's QM Input Builder tool, with the file formats displayed

The bundle’s API must also be able to respond appropriately when these providers are called upon. The provider is invoked through the bundle API’s `run_provider` method. When SEQCROW tries to run a provider, the bundle should give an instance of a `QMFileInfo`. This is a SEQCROW class for organizing all options available on the QM Input Builder tool, which includes a list of methods, basis sets and auxiliary basis sets, implicit solvents, and sections for the “additional options” tab. A detailed explanation of all these and how to specify them can be found in the source code for the `QMFileInfo` class.

A developer will also need to write a method to create the input file(s) for a software package given an AaronTools `Theory` instance. For software that requires more than one input file, such as xTB, the method should return a dictionary. The keys of the dictionary should be either the standard file extension for one of the files or the entire name of the file if it requires a fixed name. The values of the dictionary should be the contents of the corresponding file as a string. The method should also return a list of warnings for potential issues with the user-specified settings.

5.4 Adding New Algorithms to the Transition State Structure Prediction Tool

Algorithms for the Transition State Structure prediction tool are tracked with SEQCROW’s “`tss_finder_manager`” provider manager. When run, the provider should return an instance of a `TSSFinder` subclass. This class contains information about what software can be used for the TSS finding algorithm, the algorithm options, a method for adjusting the AaronTools `Theory` object, and methods for obtaining the contents of an input file for the software.

5.5 Implementing New Job Types

Jobs that run on the same computer as ChimeraX are implemented as a `QThread` subclass called `LocalJob`, which can be imported from SEQCROW’s `jobs` module. Developers should create a subclass of `LocalJob` for their new job types. `LocalJob` objects are instantiated with a job name, ChimeraX `session`, AaronTools `Theory`, an AaronTools

`Geometry` (though a `Geometry` should also be associated with the `Theory`), and various keyword arguments for other job-related options. Options can be added by modifying the `exec_options` class attribute. This is a dictionary, where the keys are the option text, and the values are ChimeraX `Option` objects, along with the arguments needed to instantiate the `Option` objects.

To execute the job, the run method must be defined. The standard procedure is to create a scratch directory in SEQCROW's scratch directory (defined in the settings), write the input files, and start the job in a subprocess. For convenience, the `LocalJob` class has a `write_file` method, which will use the method associated with the `QMFileInfo` to create the input files in the scratch directory. The subprocess should be set as the job's `process` attribute so it can be stopped if the user requests it. The `output_name` attribute should also be set to the path to the output file(s) created by the job that can be opened in ChimeraX. This should be done before starting the job subprocess in the event ChimeraX is closed before the job completes.

Cluster jobs all use the same job type: `LocalClusterJob`. The difference is that this will receive an object that is a subclass of both `ClusterSubmitTemplate` and `ProgramSubmitTemplate`, as well as several cluster-related options at instantiation. The template defines how to submit the template to the cluster and what the output files are. Because of this, it should not be necessary to make subclasses of `LocalClusterJob` for standard jobs.

Jobs launched from the Transition State Structure tool are similar to standard jobs. If running on local hardware, make a subclass of `TSSJob`. If running on a cluster, make a subclass of `ClusterTSSJob`. The main difference between these and the regular job types is they take a reactant and a product. `ClusterTSSJobs` also takes a `TSSFinder`.

There are several managers for jobs. For standard local jobs, add a provider for the "`seqcrow_job_manager`" provider manager. Running these providers should return the job's class (not an instance). For cluster jobs, the queue type (*e.g.*, Slurm or PBS) should be in the "`seqcrow_cluster_scheduling_software_manager`" manager. Running this provider should return a `ProviderManager`, which keeps track of how to run jobs on different cluster types. Running a provider for one of these managers should return an object that is a subclass of both `ClusterSubmitTemplate` and `ProgramSubmitTemplate`. These define default job execution templates, and a method for submitting jobs to the cluster.

5.6 Buried and Free Volume in 3D

Shapes rendered in ChimeraX (and many other 3D graphics programs) are comprised of three pieces of information: vertices, triangles, and normal vectors. Triangles create a surface between three vertices. Normal vectors to the vertices determine how the vertex is shaded based on lighting.

The buried or free volume visuals, such as the one shown in Figure 17, are basically a bunch of spheres for the ligand atoms, with a larger sphere around the metal center. Certain vertices on these spheres are removed to show just what is necessary for the buried or free volume. Vertices on an atom's sphere that are outside of the large sphere around the center are removed. They are also removed if they are inside of a different atom's sphere. If we are displaying the buried volume, we keep vertices on the sphere around the metal center if they are inside of one of the atom's spheres. If we are displaying free volume, we keep the vertices that are not inside of an atom's sphere.

However, this does not create a very appealing visual (see Figure 20, left). This is because we are also removing all triangles that use those vertices, which creates larger gaps where atom spheres meet each other or the larger sphere around the metal center. To rectify this, we add vertices along those intersections. Unfortunately, this precludes using a predefined set of vertices, triangles, and normal vectors for each sphere.

Instead, SEQCROW constructs a list of regularly spaced vertices on a sphere and adds vertices at intersections. The SciPy module[75] is used to determine the convex hull of the vertices. The simplices of the convex hull are the triangles for the sphere. Next, vertices are deleted, as outlined earlier. The normal vectors to the vertices are just unit vectors that point from the center of the sphere to the vertices. The result is the final, smooth surface shown in Figure 20 (right).

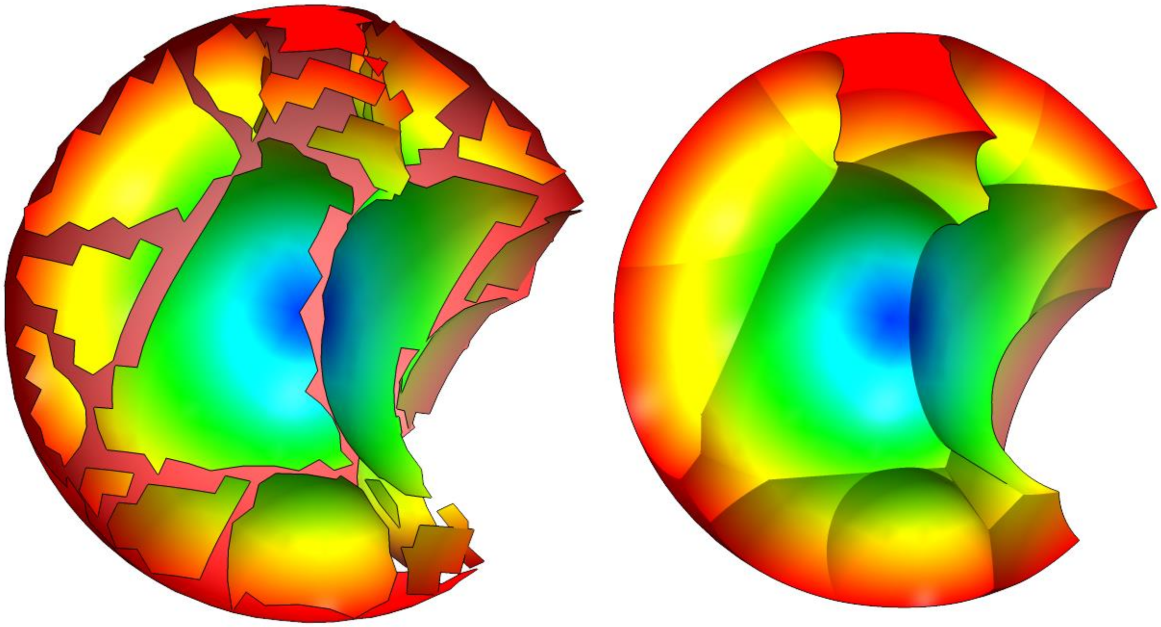


Figure 20: (left) Initial and (right) final visualization of free volume around a metal center. The metal and ligand are not shown.

5.7 SEQCROW as a Middleman

Many of the tools in SEQCROW simply take the input on the tool, convert any necessary structures to AaronTools-compatible objects, and then run an AaronTools subroutine. Compared to creating a command line script with AaronTools, the only extra step is converting objects to AaronTools equivalents. This greatly simplified adding several features to SEQCROW, as the subroutines did not need to be rewritten to work with ChimeraX `AtomicStructure` objects.

The AaronTools-compatible objects are in `SEQCROW.residue_collection`. There are two Geometry-like classes here: `Residue` and `ResidueCollection`. As the name implies, a `ResidueCollection` contains one or more `Residue` objects. There is also an `Atom` subclass. Converting a ChimeraX `AtomicStructure` to a `ResidueCollection` is as simple as passing the `AtomicStructure` to the `ResidueCollection` initialization method. For performance reasons, someone might want to only convert a portion of an AtomicStructure. In this case, they can also supply a list of ChimeraX `Residue`s when creating the `ResidueCollection`. A `ResidueCollection` can also be used to get a new `AtomicStructure` or update an existing one to match the `ResidueCollection`. This is done with the `get_chimera` and `update_chix` methods, respectively.

SEQCROW also has a `Finder` that is specific to atoms on these objects called `Atomspec`. This uses the atom specifier for the corresponding ChimeraX atom to locate the AaronTools equivalent. This is often used to locate the atoms that are selected, like when calculating the Sterimol parameters of a substituent.

5.8 Conclusions

This chapter described some of the inner workings of SEQCROW that could be useful to other developers. I described how to add new file formats for the QM Input Builder tool. I also outlined how to add new TSS finding algorithm implementations to the Transition State Structure prediction tool. The process for adding the ability to run these computations was also described. The algorithm for creating SEQCROW’s 3D buried volume representation was detailed. The approach of using SEQCROW to interface AaronTools with ChimeraX by making it easy to create AaronTools-compatible objects could also be employed to create plugins closely associated with other Python modules.

CHAPTER 6

RAVEN: A DOUBLED-ENDED GROWING STRING IMPLEMENTATION ACCELERATED BY GAUSSIAN PROCESS REGRESSION

6.1 Abstract

Raven is an implementation of the doubled-ended growing string method to locate minimum energy reaction pathways connecting a given reactant and product. It utilizes an on-the-fly Gaussian process model to predict gradients at any necessary points on the potential energy surface. The use of a Gaussian process model allows Raven to locate a good starting point for transition state structure search in the majority of cases while requiring fewer computations than traditional reaction path search implementations. Raven is benchmarked using a previously published set of 121 reactions and demonstrated by locating a TSS for the Ir-catalyzed CH functionalization reaction from Chapter 2.

6.2 Introduction

One of the major benefits of computational chemistry is the ability to locate transition state structures (TSS's). These transient structures are nigh impossible to see in an experimental setting. The analysis of TSS's can provide useful insight into the factors governing reactivity and selectivity, such as steric clashes or key electrostatic interactions. However, optimizing to a TSS will notoriously fail if the starting structure is too far from the actual first order saddle point on the potential energy surface. Obtaining a good starting point can be difficult in cases where the reaction mechanism is unintuitive or particularly novel. The use of a poor starting structure may also lead to locating a high-energy TSS, which will not typically be chemically relevant. TSS's for metal-catalyzed reactions can be particularly challenging, since these often involve multiple bonds forming and breaking, either in a stepwise or concerted fashion. Indeed, distinct stepwise and concerted mechanisms may exist for a given reaction; however, the lowest-energy pathway is usually the most chemically relevant.

There are algorithms for automating the search for transition state structures. One of the most popular and generally applicable methods is reaction path search, where the goal is to locate a pathway between two minima. This is generally done by interpolating a pathway between these minima and iteratively optimizing the pathway by computing derivatives of the energy with respect to molecular coordinates at several points and adjusting the pathway accordingly. An example of this approach is the standard string method (SSM).[76] In each iteration of the SSM, the gradient is computed at evenly spaced points along the interpolated pathway. The SSM can run into trouble if the initial pathway is unreasonable, leading to errors when computing the gradient or getting stuck in a high-energy pathway. For instance, structures near the middle of the path in the initial interpolated pathway often have atoms that are unreasonably close, leading to problems with SCF convergence, etc.

A more reliable algorithm is the double-ended growing string method (DE-GSM).[77] This starts by optimizing the pathway close to the supplied minima first and waits to optimize the middle of the pathway until the ends closer to the minima have been optimized (see Figure 22). Thus, the pathway (string) 'grows' in from both ends. The structures close to the minima are typically more reasonable than the structures more towards the middle of the pathway in the initial interpolated pathway, so errors in the computations are minimized. As an example, the top of Figure 21 shows several points on an initial pathway for the Claisen rearrangement of allyl vinyl ether determined by using a linear interpolation of Cartesian coordinates. Around the middle of this initial Claisen rearrangement pathway, many atoms are abnormally close to one another. This could result in issues when computing the energy or gradient of these structures. With reasonable structures, errors are much less common.

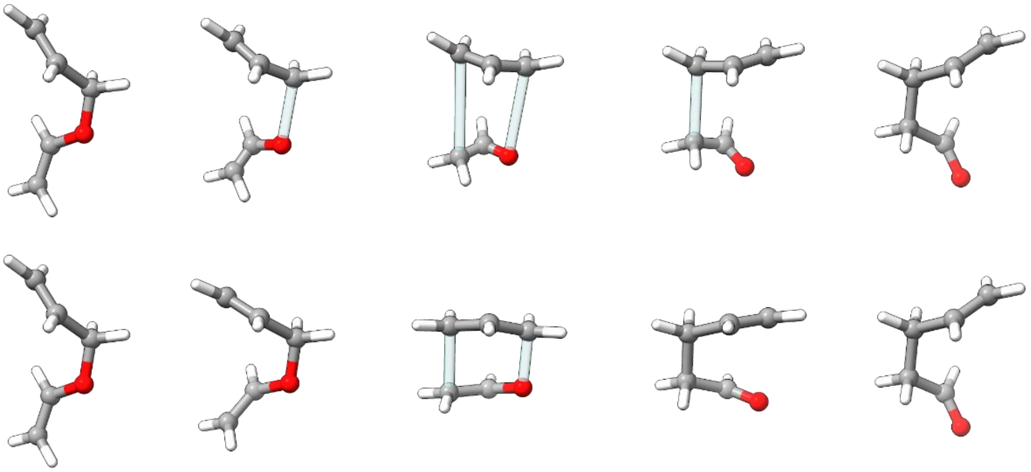


Figure 21: Structures located a poor-quality initial pathway (top) and optimized pathway (bottom) for the Claisen rearrangement of allyl vinyl ether

Another benefit to the DE-GSM approach is that the initial pathway does not have much of an impact on the outcome of the algorithm since successive segments of the string are built as the string grows. Figure 21 shows two pathways for the classic Claisen rearrangement of allyl vinyl ether. The top series shows an initial, poorly chosen reaction pathway connecting reactant and product. The bottom series of structures results from application of the DE-GSM. In this pathway, the forming/breaking bonds are much longer in the middle of the initial pathway. Applying the SSM to the initial pathway runs the risk of locating a dissociative mechanism instead of the concerted one.

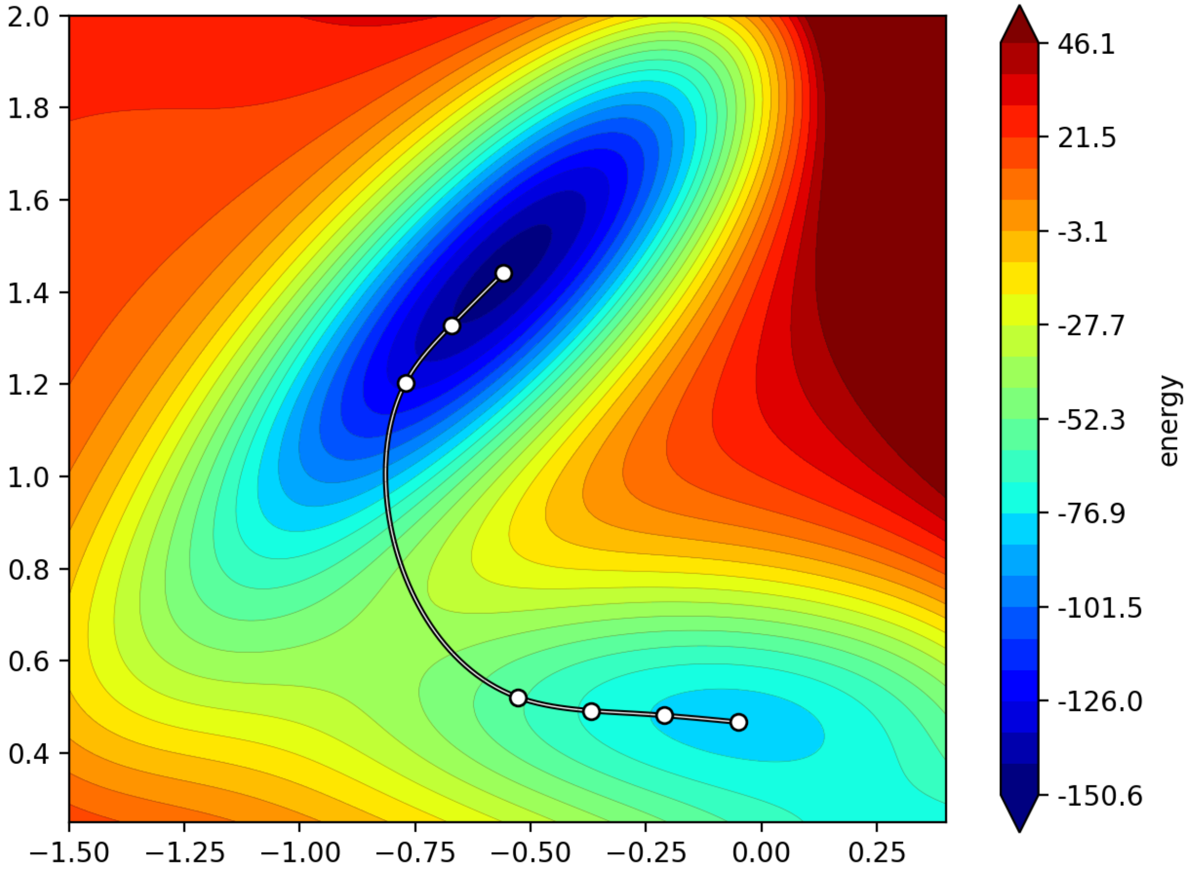


Figure 22: A snapshot of the DE-GSM method applied to a simple potential energy surface; the line represents the pathway at this iteration, and white dots (nodes) are locations where the gradient is computed during this iteration

Reaction path search algorithms can be quite costly, often requiring hundreds of gradients to be computed. This can limit the application of reaction path search techniques to small molecules or low-quality methods or force the use of a relatively small number of points along the pathway. Recently, Kästner *et al.*[78] described a reaction path search algorithm that uses Gaussian process regression (GPR) to greatly reduce the number of computations required to converge the reaction path search algorithm.

GPR is a machine learning model that gives a Gaussian probability distribution of output values for a given input. This distribution is characterized by a mean and variance, both of which are determined by a set of observed inputs and outputs. The similarity between a pair of points (x_1, x_2) is gauged using a kernel function. Any function could be used as a kernel, but generally the function should have a higher value for a pair of inputs that are similar, and it should be closer to zero for dissimilar input pairs. Two popular simple kernel functions are the radial basis function:

$$k(x_1, x_2; l) = \exp\left(-\frac{\|x_1 - x_2\|^2}{2l^2}\right) \tag{6.1}$$

and the Matérn kernel:

$$k(x_1, x_2; l; v) = \frac{2^{1-\nu}}{\Gamma(v)}\left(\sqrt{2v}\frac{\|x_1 - x_2\|}{l}\right)^{v} K_\nu\left(\sqrt{2v}\frac{\|x_1 - x_2\|}{l}\right), \tag{6.2}$$

where K_ν is the modified Bessel function of the second kind. A value of $v = 5/2$ is often chosen. In this case, the kernel is:

$$k(x_1, x_2; l) = \left(1 + \sqrt{5}\frac{\|x_1 - x_2\|}{l} + \frac{5\|x_1 - x_2\|^2}{3l^2}\right)\exp\left(-\sqrt{5}\frac{\|x_1 - x_2\|}{l}\right). \tag{6.3}$$

The value of these kernels for the pair of points (x_1, x_2) is inversely proportional to the distance between the points. The parameter l determines how quickly the kernel values fall off.

In order to obtain the distribution of output values for a new input value, x_*, the kernel function is evaluated for each pair of points in the training set, as well as for each training point with x_*, and x_* with itself. The mean of this distribution is given by

$$\langle f \rangle = K_*^T(K + \sigma^2 I)^{-1} y, \tag{6.4}$$

where K_* is a vector containing the kernel values for each training point with x_*, K is a matrix containing the kernel evaluated for each pair of training points, y is the output for each training point, I is the identity matrix, and σ is a regularization parameter to account for noise in the training data. Because the mean is the most probable value for a Gaussian distribution, this is simply chosen to be the predicted output for this new input. Note that $(K + \sigma^2 I)^{-1}y$ only includes training data and a parameter. We can store this value as a vector until we need to rebuild the model. Predicting the mean then simply requires the evaluation of the kernel function to build K_* and taking the dot product with the $(K + \sigma^2 I)^{-1}y$ vector. This is generally quick for simple kernels. The variance for the distribution is:

$$var(f) = K_{**} + \sigma^2 I - K_*^T (K + \sigma^2 I)^{-1} K_*, \tag{6.5}$$

where K_{**} is the value of the kernel function evaluated for the new point with itself.

For reaction path search, the input to the GPR model can be the coordinates of the atoms. The output could be the energy or gradient for those coordinates. Kästner's approach utilized a GPR model to predict many of the required gradients instead of running QM computations. However, their method was based on the nudged elastic band (NEB) algorithm for reaction path search. NEB is similar to SSM: low-quality initial interpolations can lead to errors or failure to converge to a lower-energy pathway.

In this chapter, I describe Raven, a DE-GSM implementation that uses GPR to reduce the computational cost. Raven is built using AaronTools[27] to setup, process, and run all necessary computations. To evaluate the performance of this implementation, I compare its performance to Jónsson *et al.*'s NEB implementation, which does not use GPR, across a test set of 121 reactions.[79]

6.3 The Algorithm

Raven builds an interpolation between two structures by combining one-dimensional interpolations along each atom's X, Y, and Z coordinate. This is done using one of the 1-D interpolation methods available in the SciPy library.[75] In practice, we use the piecewise cubic

Hermite interpolating polynomial (PCHIP) method. PCHIP yields a monotonic function between each point, which seems to lend some stability to the interpolation. Before the interpolation is built, the Kabsch algorithm[80] is used to align all structures along the pathway to the reactant. This removes motion due to rotation or translation of the center of mass. Although the tangent to the interpolation can be calculated at any point along the pathway, it might not be accurate for the frontier nodes while the string is still growing due to the larger gap in between the nodes. For a potentially more accurate tangent at the frontier, we use the linear direction from the frontier to the adjacent node.

In Raven, the input for the GPR model is the coordinates of a structure, and the output is the gradient of the potential energy surface at that point. If the maximum square root of the variance exceeds a threshold for any node on the pathway, the gradient for that structure will be explicitly computed at the user-specified level of theory using the requested QM package (Raven can use gradients from Gaussian, ORCA, or Psi4). This computation will then be added to the set of observations, and the GPR model will be retrained. Additionally, when a new node is added to the pathway, the node's gradient is always computed and used to retrain the model. By default, Raven uses the Matérn kernel with $\upsilon = 5/2$ and $l = 10$ Å.

As gradients are readily accessible from the GPR model, Raven uses gradient-descent-based line search strategy to optimize points on the pathway. The tangent to the interpolated pathway is projected out of the gradient. This determines the step direction. Raven will take up to 10 steps in this direction, ensuring not to move any atom too far from its previous position (default is 0.01 Å). Of these 10 steps, Raven will use the one with the lowest energy. The negative change in energy for each step is calculated from the dot product of the gradient at that step with the step displacement vector. This scheme is also used to locate minima and maxima along the interpolated pathway.

Once a frontier node meets the convergence criteria shown in Table 6, a new node is added a bit farther along that side of the pathway. Note that the criteria include the number of iterations without a new node. This is because Raven often gets stuck optimizing nodes without making much improvement from one iteration to the next. This could be due to known convergence issues with gradient-based optimization algorithms or insufficient training data in the GPR model. Once the pathway has the desired number of nodes, and each node meets the convergence criteria, optimization stops. Raven will then locate local maxima and minima along the pathway. By default, Raven will use 26 nodes. This is significantly more than traditional reaction path search implementations, though more nodes incur little computational expense due to the use of the GPR model.

Table 6: Raven's default convergence criteria

<table>
<tr><td>RMS $F_{\perp}$</td><td>Max. $F_{\perp}$</td><td>RMS displacement</td><td>Max. displacement</td></tr>
<tr><td>$< 5 \times 10^{-3}$ a.u.</td><td>$< 7.5 \times 10^{-3}$ a.u.</td><td>$< 3.5 \times 10^{-3}$ Å</td><td>$< 5 \times 10^{-3}$ Å</td></tr>
<tr><td colspan="4">or</td></tr>
<tr><td colspan="2">Iterations without new node</td><td colspan="2">Predicted ΔE</td></tr>
<tr><td colspan="2">&gt; 250</td><td colspan="2">< 0 and $> -5 \times 10^{-4}$ a.u.</td></tr>
<tr><td colspan="4">or</td></tr>
<tr><td colspan="2">Iterations without new node</td><td colspan="2">Predicted ΔE</td></tr>
<tr><td colspan="2">&gt; 500</td><td colspan="2">&lt; 0 a.u.</td></tr>
</table>

6.4 Running Raven

An INI-formatted AaronTools config file is Raven’s primary input (see Figure 23). In the `[Raven]` section of the config file, users must specify the paths to files containing the structures of the reactant and product. The default parameters for the GPR model, number of nodes, and convergence criteria can also be adjusted in the `[Raven]` section. The desired level of theory is specified in the `[Theory]` section. In the `[Job]` section, users specify the electronic structure package to run, the number of processors, and the amount of memory to use. Raven can be used with most program supported by AaronTools (currently Gaussian, ORCA, and Psi4).

```
[Theory]
method      =   M06-2X
basis       =   def2-SVP

[Job]
exec_type   =   ORCA
procs       =   2
exec_memory =   8
memory      =   6
wall        =   8

[Raven]
reactant    =   reactant.xyz
product     =   product.xyz
```

Figure 23: Example config file for running Raven

If Raven is being run in parallel mode, where all computations are submitted separately to a cluster, the `[Job]` section should also have the memory and wall time limits for each job. A template submission file should also be specified on the command line when running Raven. If Raven is being run in serial mode, where all computations are run one after another as a subprocess of Raven, the executable for the electronic structure package needs to be specified on the command line at run time.

6.5 Benchmark Set

In order to evaluate the performance of this implementation, Raven was tested for the 121 test cases published by Jónsson *et al.*[79] This test set comprises slightly modified versions of benchmark sets from Birkholz and Schlegel[81] and Zimmerman.[82] The reactant and product structures for each reaction were first reoptimized using ORCA 5.0.3 at the B3LYP-D3BJ/def2-

SVP level of theory. The reaction path was sought with the same level of theory using Raven with gradients and energies supplied by ORCA. The structures of the reactant, product, and optimized TSSs or guess TS structures from Raven can be found in Appendix A. In an effort to see if we found a reasonable pathway for these reactions, we also compared TSSs from Raven to those reported by Jónsson *et al.*[79]

6.6 Benchmarking Results

The starting TSSs located by Raven could be optimized to the correct TSS in 96 of the 121 test cases (about 80% success rate). TSSs were determined to be correct if optimizing starting from the TSS, displaced by a small amount along the imaginary vibrational mode, resulted in either the reactant or product structure. Distributions of key performance metrics can be found in Figure 24. The mean RMSD between Raven's guess and the optimized TSS across all 121 test cases is 0.3 Å; however, in several cases the RMSD was close to 0.8 Å.

Locating successful TSS guesses required an average of 167 computations per system, while one system required more than 500 gradient evaluations, the rest required around 400 or fewer. For comparison, the NEB-TS method of Jónsson *et al.* required an average of 305 gradients.[79] The DE-GSM implementation of Zimmerman required an average of 500 gradients for a subset of these reactions.[82] Our implementation is not directly comparable to these two, as these both incorporate saddle point optimization to locate a TSS. Thus, their methods give the TSS, not a guess. However, TS optimizations initiated from Raven's guesses were required to converge within ORCA's default number of optimization steps.

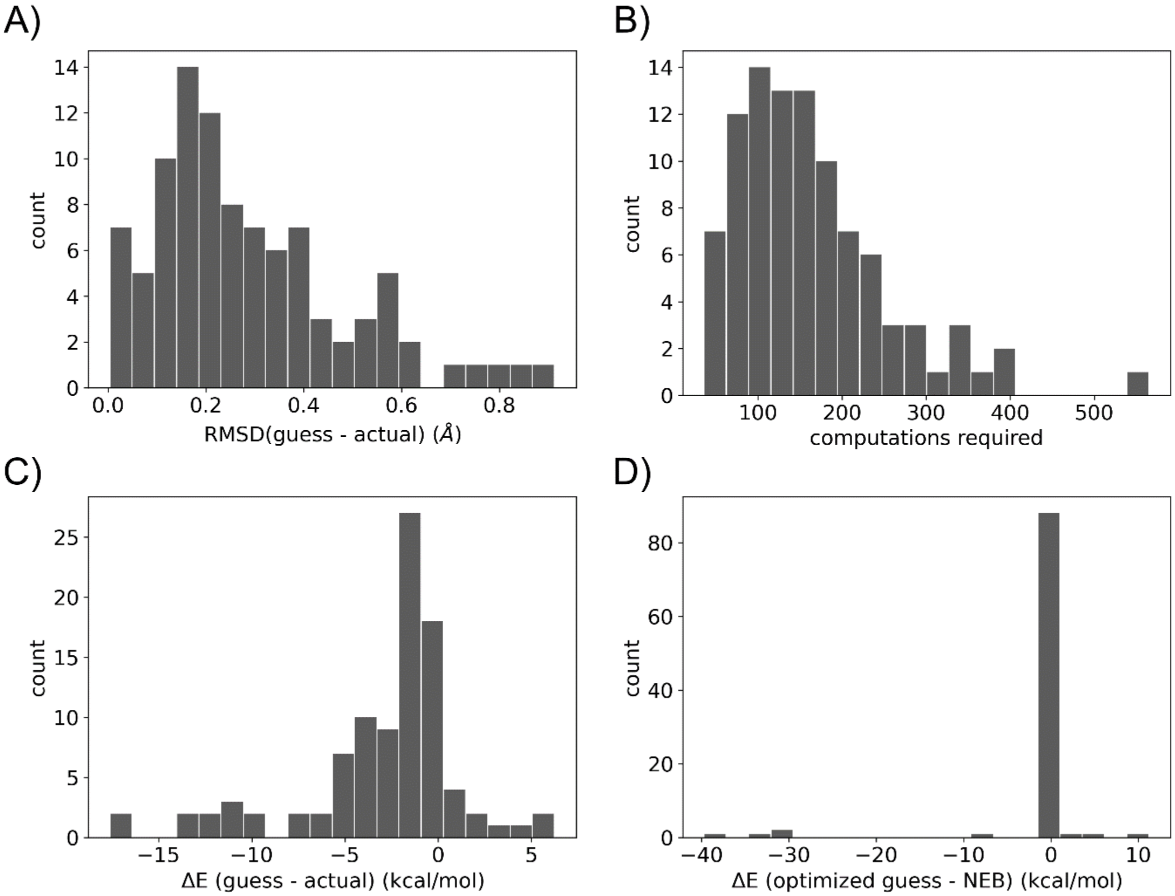


Figure 24: Key performance metrics for Raven. A) RMSD between Raven's guess TSS and the optimized TSS; B) computations required for Raven to converge in successful cases; C) difference in energy between Raven's successful guess TSSs and the optimized TSS; D) difference in energy between the TSS optimized from Raven’s guess and the TSS reported by Jónsson *et al.*

Unfortunately, Raven was unable to locate an adequate TSS guess in about 20% of the test cases. However, we can inspect some of the cases where Raven failed and speculate as to how the method can be improved. All but one of the test cases that failed (and many that succeeded) have Hessians with multiple negative eigenvalues. This indicates that some degrees of freedom were not fully optimized, such as a torsional rotation. A common pitfall when optimizing a TSS is that the optimization algorithm will choose the wrong degree of freedom to move to a saddle point. This could be the case with these failures. If these other degrees of freedom are close to a saddle point, it could be difficult for Raven to leave that saddle point. This is because the gradient will be nearly zero for that motion, meaning Raven will essentially not move from that location.

A more sophisticated optimization algorithm could alleviate these types of problems. Many quantum chemistry software packages utilize quasi-Newton methods for optimizing structures. These methods employ approximate second derivatives to avoid getting stuck on saddle points. We could take the Jacobian of the GPR's prediction for the gradient to get a Hessian. However, this Hessian will not necessarily be symmetric. This is a required property for Hessian matrices. More testing is required to determine if the Jacobian of the GPR-predicted gradient, or some other approximate Hessian, can be used to improve the performance of Raven.

Another possible reason that Raven could fail in some cases is that the GPR model does not incorporate enough training data to accurately model the shape of the potential energy surface. Test cases were also run with a tighter root variance threshold (1×10^{-3}). Although this should ensure the GPR will more closely match the actual potential energy surface, it did not seem to improve Raven's TSS guesses significantly. It did, however, increase the average number of computations by ~130.

It is also noteworthy that in five cases the pathway located by Raven had considerably lower energy barriers compared to those reported by Jónsson *et al.*[79] For example, Figure 25 shows two pathways for the addition of vinyl alcohol to formaldehyde to give 3-hydroxypropanal. Jónsson *et al.*[79] reported a single saddle point for this reaction, although they did not verify that this is a concerted mechanism. The TSS they reported is 98 kcal/mol above the reactant. Moreover, their reported TSS, even though it was deemed successful, does not appear to connect the reactant and product. Raven, on the other hand, identified a three-step mechanism that connects the reactant and product; however, the second step in Raven's pathway is the interconversion of two enantiomers, which is an artifact of the fact that atom indices are not automatically reassigned. The achiral product could be formed from either intermediate. After optimizing TSS guesses from Raven, the highest-energy TSS on the three-step pathway is ~30 kcal/mol lower than the barrier in the pathway located by Jónsson *et al.*[79] This could be a result of using more nodes, which increases the flexibility of the pathway, and enables multistep mechanisms to be uncovered.

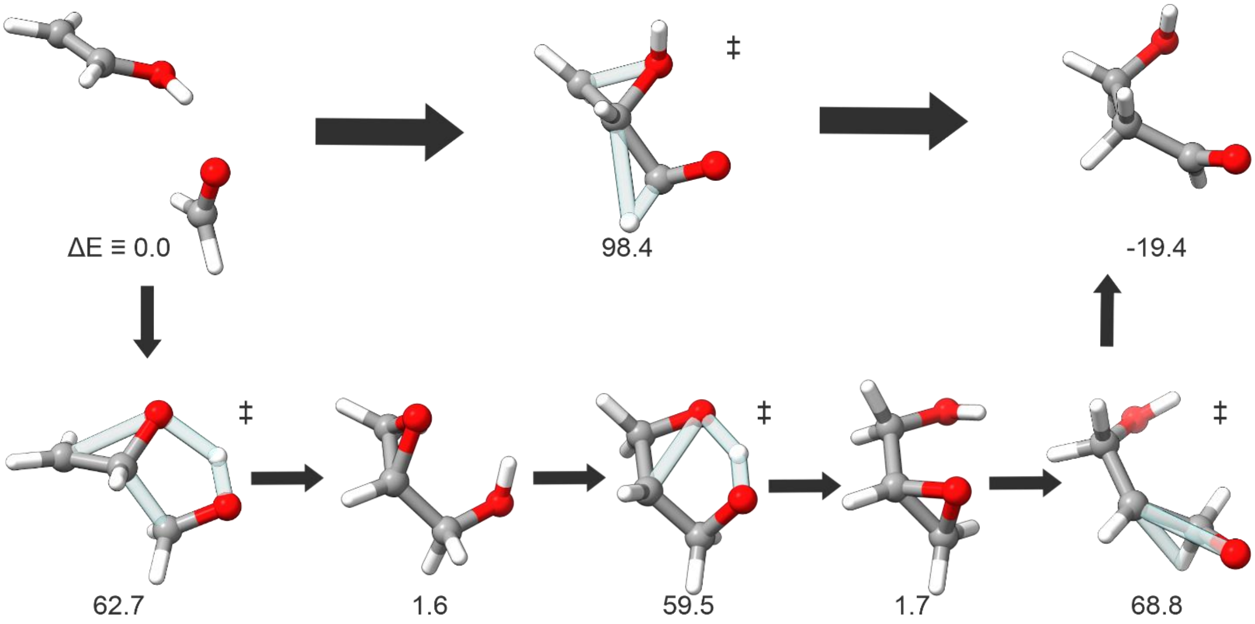


Figure 25: Mechanism reported by Jónsson *et al.* (top) and mechanism discovered by Raven (bottom); energies relative to the reactant are in kcal/mol

At least one of the reactions where Raven failed was due to the ordering of atoms: the decomposition of sulfolene to sulfur dioxide and 1,3-butadiene. Raven's pathway, shown in Figure 26, passes close to the correct TSS and the adjacent intermediate is constitutionally the same as the product, but the methylidene groups on butadiene need to be flipped to get the atom ordering correct. Consequently, Raven rotates both methylidene groups, forming and then opening a four-membered ring in the process. Although the initial decomposition was marked as a guess TSS, higher energy guesses were considered first for attempts to optimize to the actual

TSS. Of course, the TSSs for forming and opening the four-membered ring are much higher in energy, but an attempt to locate a TSS starting from one of these guesses failed.

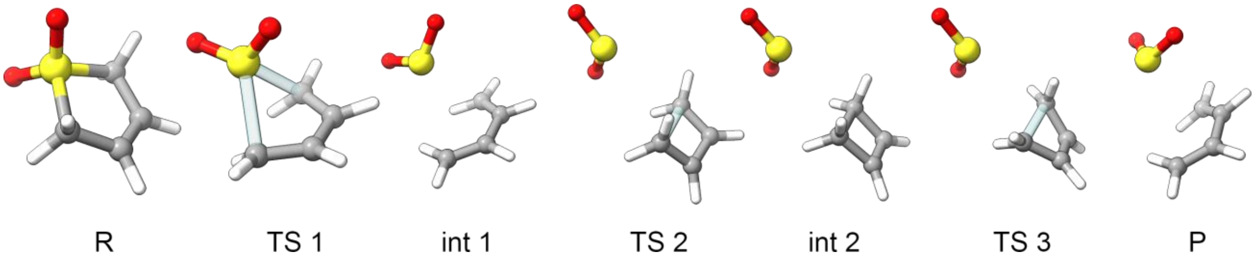


Figure 26: Raven's key TS guesses and intermediates for the decomposition of sulfolene to SO_2 and 1,3-butadiene. R and P indicate the reactant and product, respectively.

6.7 Iridium-Catalyzed C-H Activation

In addition to the benchmarking set, I wanted to test Raven on something a bit more interesting for me personally: the C-H activation of N-acetyl indole (see Chapter 2). For this, I used ORCA to run M06-2X/def2-SVP gradients where necessary. I adjusted the similarity falloff parameter to 20 Å, as 10 Å seemed to be running excessive computations. Default values were used for all other settings. Raven converged after 205 gradient computations and produced the guess TSS shown in Figure 27 (left), which has the carboxylate in a sideways orientation. During optimization, the carboxylate adopted an upright orientation. The RMSD between these two structures is 0.71 Å, though the energy difference is less than 4 kcal/mol.

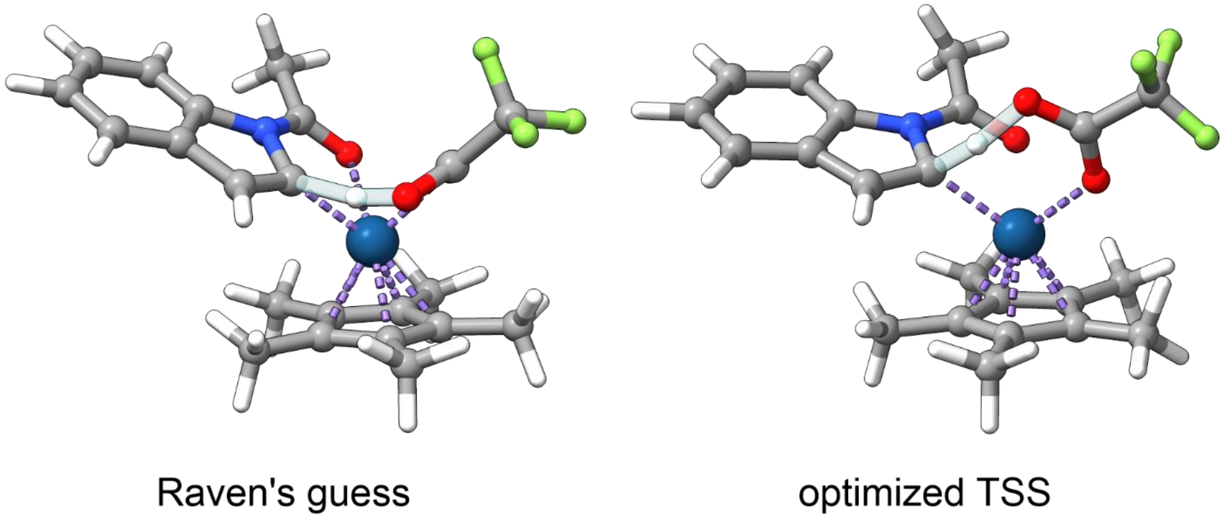


Figure 27: Raven’s guess TSS (left) and optimized TS guess (right) for the C-H activation of N-acetyl indole

6.8 Conclusions and Discussion

Raven was able to locate usable TSS guesses in ~80% of test cases for a test set of 121 reactions, as well as a TSS for the C-H activation of N-acetyl indole. It is a relatively efficient approach, requiring less than 200 energy and gradient computations per pathway optimization on average across the benchmarking set. This can be compared to ~300 computations per pathway reported for the same test set by Jónsson *et al.* using the NEB-TS method available in ORCA.[79] This efficiency can be attributed to the use of a GPR model to predict the gradient for most optimization steps. Incorporating GPR permits the use of more nodes without greatly increasing the computational cost. With more nodes, the interpolation is more flexible and can locate lower energy, multi-step pathways. More investigation is necessary to see if Raven’s performance can be increased further.

Although Raven’s pathway in Figure 25 is better given the constraints on the order of the atoms, it is not the best way to get from formaldehyde and vinyl alcohol to 3-hydroxypropanal. Reaction 16 (see Appendix A) achieves an equivalent transformation by means of a one-step mechanism with a barrier of just 12.3 kcal/mol. Finding this one-step pathway with Raven required less than half as many computations as the three-step mechanism in Figure 25. The dangers of using a bad atom order can also be seen in the case of the decomposition of sulfolene. Thus, the onus of finding the lowest energy and most efficient pathway still falls on the user. This can be problematic for cases where chemical intuition is lacking.

The Zimmerman group has developed ZStruct for finding reaction mechanisms without regard for the order of atoms.[83] ZStruct uses reaction graphs to identify bonds that need to be broken or formed, generates potential intermediates, and then uses DE-GSM to locate pathways connecting these intermediates. Intermediates are located until the desired product is found. Zimmerman *et al.* have used ZStruct to investigate the mechanism of C-N bond formation in the reductive elimination step of a Pd-catalyzed reaction.[84] Several unintuitive pathways were uncovered, but around 80,000 CPU hours were required for this study. Although incorporating GPR with reaction path search seems to reduce the cost of locating TSSs, exhaustive searches would likely remain prohibitively costly in many cases.

A major deficiency in Raven compared to the NEB-TS method[79] or Zimmerman's GSM-EV-ES method[82] is that Raven only gives a guess for the TSS. These other methods incorporate TS optimization algorithms, so they give an optimized TSS. Users will have to hope that Raven's TSS guess is adequate for finding their desired TSS. Jónsson *et al.* and Zimmerman both noted that incorporating TS optimization into their algorithms improved the reliability of their respective implementations, as the TSS optimization algorithm can benefit from knowledge of the desired reaction pathway. A similar improvement could be expected if TS optimization is added to Raven.

The parameter selection for the GPR model has not been tested. The value of the similarity falloff parameter, l, was chosen based on preliminarily good performance for just a few of these reactions. The default value of 10 Å may not be appropriate for larger molecules, where seemingly small changes in structure add up more quickly. Future studies should investigate the performance of Raven with respect to changes in the GPR parameters.

Raven will be made available on GitHub as a command line utility. In the future, Raven will also be included as an option on SEQCROW’s Transition State Structure tool. This will provide an easy-to-use interface to deploy Raven.

CHAPTER 7

CONCLUSIONS

I have described several toolkits to aid computational chemists in a variety of tasks. I have used AARON to assess the performance of DFT methods for predicting the regioselectivity of a C-H functionalization reaction. DFT methods were found to be inaccurate and often lacked predictive trends in selectivity. The latest version of AaronTools can set up computations for many popular quantum chemistry software packages. With AaronTools, users can calculate several parameters that are indicators of reactivity, produce simulated spectra, and generate molecular structures. SEQCROW provides a graphical interface to most of AaronTools, and can be used to produce high-quality graphics. Other bundle developers can utilize several of SEQCROW’s interfaces for their own purposes. Finally, Raven is an efficient implementation of the double-ended growing string method. The incorporation of a Gaussian process model for gradients of the potential energy surface reduces the number of computations required to estimate transition state structures. This enables more nodes to be added to the reaction pathway interpolation, increasing the flexibility of the pathway, and allowing for the discovery of multistep mechanisms.

I hope that these tools can help both veteran and novice computational chemists alike. The existence of these tools allows users to turn their focus to their science rather than worrying about the technical aspects of running different computations using different software packages. Veterans would not have to sink time into developing their own scripts. Novices do not have to spend time learning how to work on a command line interface to have access to the same features as veterans. Of course, if beginners want to learn the command line, AaronTools will still be there.

REFERENCES

(1) Sigman, M. S.; Harper, K. C.; Bess, E. N.; Milo, A. The Development of Multidimensional Analysis Tools for Asymmetric Catalysis and Beyond. *Accounts of Chemical Research* **2016**, *49* (6), 1292-1301. DOI: 10.1021/acs.accounts.6b00194.

(2) Harper, K. C.; Bess, E. N.; Sigman, M. S. Multidimensional steric parameters in the analysis of asymmetric catalytic reactions. *Nat Chem* **2012**, *4* (5), 366-374. DOI: 10.1038/nchem.1297.

(3) Tolman, C. A.; Seidel, W. C.; Gosser, L. W. Formation of three-coordinate nickel(0) complexes by phosphorus ligand dissociation from NiL4. *Journal of the American Chemical Society* **1974**, *96* (1), 53-60. DOI: 10.1021/ja00808a009.

(4) Guan, Y.; Ingman, V. M.; Rooks, B. J.; Wheeler, S. E. AARON: An Automated Reaction Optimizer for New Catalysts. *J Chem Theory Comput* **2018**, *14* (10), 5249-5261. DOI: 10.1021/acs.jctc.8b00578.

(5) Guan, Y.; Wheeler, S. E. Automated Quantum Mechanical Predictions of Enantioselectivity in a Rhodium-Catalyzed Asymmetric Hydrogenation. *Angew Chem Int Ed Engl* **2017**, *56* (31), 9101-9105. DOI: 10.1002/anie.201704663.

(6) Doney, A. C.; Rooks, B. J.; Lu, T.; Wheeler, S. E. Design of Organocatalysts for Asymmetric Propargylations through Computational Screening. *ACS Catalysis* **2016**, *6* (11), 7948-7955. DOI: 10.1021/acscatal.6b02366.

(7) Wheeler, S. E.; Seguin, T. J.; Guan, Y.; Doney, A. C. Noncovalent Interactions in Organocatalysis and the Prospect of Computational Catalyst Design. *Acc Chem Res* **2016**, *49* (5), 1061-1069. DOI: 10.1021/acs.accounts.6b00096.

(8) Rooks, B. J.; Haas, M. R.; Sepúlveda, D.; Lu, T.; Wheeler, S. E. Prospects for the Computational Design of Bipyridine N,N′-Dioxide Catalysts for Asymmetric Propargylation Reactions. *ACS Catalysis* **2014**, *5* (1), 272-280. DOI: 10.1021/cs5012553.

(9) Andreola, L. R.; Wheeler, S. E. Importance of favourable non-covalent contacts in the stereoselective synthesis of tetrasubstituted chromanones. *Organic Chemistry Frontiers* **2022**, *9* (11), 3027-3033. DOI: 10.1039/d2qo00090c.

(10) Vaganov, V. Y.; Fukazawa, Y.; Kondratyev, N. S.; Shipilovskikh, S. A.; Wheeler, S. E.; Rubtsov, A. E.; Malkov, A. V. Optimization of Catalyst Structure for Asymmetric Propargylation of Aldehydes with Allenyltrichlorosilane. *Advanced Synthesis & Catalysis* **2020**, *362* (23), 5467-5474. DOI: 10.1002/adsc.202000936.

(11) Kim, Y.; Park, Y.; Chang, S. Delineating Physical Organic Parameters in Site-Selective C-H Functionalization of Indoles. *ACS Cent Sci* **2018**, *4* (6), 768-775. DOI: 10.1021/acscentsci.8b00264.

(12) Robidas, R.; Legault, C. Y. CalcUS: An Open-Source Quantum Chemistry Web Platform. *Journal of Chemical Information and Modeling* **2022**, *62* (5), 1147-1153. DOI: 10.1021/acs.jcim.1c01502.

(13) Gorelsky, S. I.; Lapointe, D.; Fagnou, K. Analysis of the Concerted Metalation-Deprotonation Mechanism in Palladium-Catalyzed Direct Arylation Across a Broad Range of Aromatic Substrates. *Journal of the American Chemical Society* **2008**, *130* (33), 10848-10849. DOI: 10.1021/ja802533u.

(14) Park, C.-H.; Ryabova, V.; Seregin, I. V.; Sromek, A. W.; Gevorgyan, V. Palladium-Catalyzed Arylation and Heteroarylation of Indolizines. *Organic Letters* **2004**, *6* (7), 1159-1162. DOI: 10.1021/ol049866q.

(15) Wang, L.; Carrow, B. P. Oligothiophene Synthesis by a General C–H Activation Mechanism: Electrophilic Concerted Metalation–Deprotonation (eCMD). *ACS Catalysis* **2019**, *9* (8), 6821-6836. DOI: 10.1021/acscatal.9b01195.

(16) Ma, W.; Mei, R.; Tenti, G.; Ackermann, L. Ruthenium(II)-Catalyzed Oxidative C□H Alkenylations of Sulfonic Acids, Sulfonyl Chlorides and Sulfonamides. *Chemistry – A European Journal* **2014**, *20* (46), 15248-15251, https://doi.org/10.1002/chem.201404604. DOI: https://doi.org/10.1002/chem.201404604 (acccessed 2022/07/10).

(17) Rogge, T.; Oliveira, J. C. A.; Kuniyil, R.; Hu, L.; Ackermann, L. Reactivity-Controlling Factors in Carboxylate-Assisted C–H Activation under 4d and 3d Transition Metal Catalysis. *ACS Catalysis* **2020**, *10* (18), 10551-10558. DOI: 10.1021/acscatal.0c02808.

(18) Kim, Y.; Park, Y.; Chang, S. Delineating Physical Organic Parameters in Site-Selective C–H Functionalization of Indoles. *ACS Central Science* **2018**, *4* (6), 768-775. DOI: 10.1021/acscentsci.8b00264.

(19) Goerigk, L.; Hansen, A.; Bauer, C.; Ehrlich, S.; Najibi, A.; Grimme, S. A look at the density functional theory zoo with the advanced GMTKN55 database for general main group thermochemistry, kinetics and noncovalent interactions. *Physical Chemistry Chemical Physics* **2017**, *19* (48), 32184-32215, 10.1039/C7CP04913G. DOI: 10.1039/C7CP04913G.

(20) Dohm, S.; Hansen, A.; Steinmetz, M.; Grimme, S.; Checinski, M. P. Comprehensive Thermochemical Benchmark Set of Realistic Closed-Shell Metal Organic Reactions. *Journal of Chemical Theory and Computation* **2018**, *14* (5), 2596-2608. DOI: 10.1021/acs.jctc.7b01183.

(21) Goerigk, L.; Grimme, S. A thorough benchmark of density functional methods for general main group thermochemistry, kinetics, and noncovalent interactions. *Physical Chemistry Chemical Physics* **2011**, *13* (14), 6670-6688, 10.1039/C0CP02984J. DOI: 10.1039/C0CP02984J.

(22) Sperger, T.; Sanhueza, I. A.; Kalvet, I.; Schoenebeck, F. Computational Studies of Synthetically Relevant Homogeneous Organometallic Catalysis Involving Ni, Pd, Ir, and Rh: An Overview of Commonly Employed DFT Methods and Mechanistic Insights. *Chem Rev* **2015**, *115* (17), 9532-9586. DOI: 10.1021/acs.chemrev.5b00163.

(23) *Gaussian 09, Revision D.01*; Gaussian, Inc.: 2009.

(24) Grimme, S. Supramolecular binding thermodynamics by dispersion-corrected density functional theory. *Chem. Eur. J.* **2012**, *18* (32), 9955-9964. DOI: 10.1002/chem.201200497.

(25) Neese, F. Software update: the ORCA program system, version 4.0. *Wiley Interdisciplinary Reviews-Computational Molecular Science* **2018**, *8* (1). DOI: ARTN e1327
10.1002/wcms.1327.

(26) Rooks, B. J.; Haas, M. R.; Sepúlveda, D.; Lu, T.; Wheeler, S. E. Prospects for the Computational Design of BipyridineN,N′-Dioxide Catalysts for Asymmetric Propargylation Reactions. *ACS Catalysis* **2014**, *5* (1), 272-280. DOI: 10.1021/cs5012553.

(27) Ingman, V. M.; Schaefer, A. J.; Andreola, L. R.; Wheeler, S. E. QChASM: Quantum chemistry automation and structure manipulation. *WIREs Computational Molecular Science* **2021**, *11* (4), e1510, https://doi.org/10.1002/wcms.1510. DOI: https://doi.org/10.1002/wcms.1510 (acccessed 2022/06/17).

(28) Riplinger, C.; Neese, F. An efficient and near linear scaling pair natural orbital based local coupled cluster method. *The Journal of Chemical Physics* **2013**, *138* (3), 034106. DOI: 10.1063/1.4773581 (acccessed 2022/06/20).

(29) Parrish, R. M.; Parker, T. M.; Sherrill, C. D. Chemical Assignment of Symmetry-Adapted Perturbation Theory Interaction Energy Components: The Functional-Group SAPT Partition. *J Chem Theory Comput* **2014**, *10* (10), 4417-4431. DOI: 10.1021/ct500724p.

(30) Grimme, S. Semiempirical GGA-type density functional constructed with a long-range dispersion correction. *J. Comput. Chem.* **2006**, *27* (15), 1787-1799. DOI: 10.1002/jcc.20495.

(31) Grimme, S.; Antony, J.; Ehrlich, S.; Krieg, H. A consistent and accurate ab initio parametrization of density functional dispersion correction (DFT-D) for the 94 elements H-Pu. *J Chem Phys* **2010**, *132* (15), 154104. DOI: 10.1063/1.3382344.

(32) Grimme, S.; Ehrlich, S.; Goerigk, L. Effect of the damping function in dispersion corrected density functional theory. *J. Comput. Chem.* **2011**, *32* (7), 1456-1465. DOI: 10.1002/jcc.21759.

(33) Bilbrey, J. A.; Kazez, A. H.; Locklin, J.; Allen, W. D. Exact ligand cone angles. *Journal of Computational Chemistry* **2013**, *34* (14), 1189-1197. DOI: https://doi.org/10.1002/jcc.23217.

(34) Hillier, A. C.; Sommer, W. J.; Yong, B. S.; Petersen, J. L.; Cavallo, L.; Nolan, S. P. A Combined Experimental and Theoretical Study Examining the Binding ofN-Heterocyclic Carbenes (NHC) to the Cp*RuCl (Cp* = η5-C5Me5) Moiety: Insight into Stereoelectronic Differences between Unsaturated and Saturated NHC Ligands. *Organometallics* **2003**, *22* (21), 4322-4326. DOI: 10.1021/om034016k.

(35) Antinucci, G.; Dereli, B.; Vittoria, A.; Budzelaar, P. H. M.; Cipullo, R.; Goryunov, G. P.; Kulyabin, P. S.; Uborsky, D. V.; Cavallo, L.; Ehm, C.; et al. Selection of Low-Dimensional 3-D Geometric Descriptors for Accurate Enantioselectivity Prediction. *ACS Catalysis* **2022**, *12* (12), 6934-6945. DOI: 10.1021/acscatal.2c00976.

(36) Verloop, A. THE STERIMOL APPROACH: FURTHER DEVELOPMENT OF THE METHOD AND NEW APPLICATIONS. In *Pesticide Chemistry: Human Welfare and Environment*, Doyle, P., Fujita, T. Eds.; Pergamon, 1983; pp 339-344.

(37) Pino-Rios, R.; Yañez, O.; Inostroza, D.; Ruiz, L.; Cardenas, C.; Fuentealba, P.; Tiznado, W. Proposal of a simple and effective local reactivity descriptor through a topological analysis of an

orbital-weighted fukui function. *Journal of Computational Chemistry* **2017**, *38* (8), 481-488, https://doi.org/10.1002/jcc.24699. DOI: https://doi.org/10.1002/jcc.24699 (acccessed 2022/06/14).

(38) Lowe, D. *OPSIN: Open Parser for Systematic IUPAC nomenclature*. University of Cambridge Centre for Molecular Informatics, https://opsin.ch.cam.ac.uk/

(39) *RDKit: Open-source cheminformatics; http://www.rdkit.org*

(40) Silva, F. T.; Lins, S. L. S.; Simas, A. M. Stereoisomerism in Lanthanide Complexes: Enumeration, Chirality, Identification, Random Coordination Ratios. *Inorganic Chemistry* **2018**, *57* (17), 10557-10567. DOI: 10.1021/acs.inorgchem.8b01133.

(41) Houk, K. N.; Liu, F. Holy Grails for Computational Organic Chemistry and Biochemistry. *Acc Chem Res* **2017**, *50* (3), 539-543. DOI: 10.1021/acs.accounts.6b00532.

(42) Poree, C.; Schoenebeck, F. A Holy Grail in Chemistry: Computational Catalyst Design: Feasible or Fiction? *Acc Chem Res* **2017**, *50* (3), 605-608. DOI: 10.1021/acs.accounts.6b00606.

(43) Ryu, H.; Park, J.; Kim, H. K.; Park, J. Y.; Kim, S.-T.; Baik, M.-H. Pitfalls in Computational Modeling of Chemical Reactions and How To Avoid Them. *Organometallics* **2018**, *37* (19), 3228-3239. DOI: 10.1021/acs.organomet.8b00456.

(44) Ahn, S.; Hong, M.; Sundararajan, M.; Ess, D. H.; Baik, M. H. Design and Optimization of Catalysts Based on Mechanistic Insights Derived from Quantum Chemical Reaction Modeling. *Chem Rev* **2019**, *119* (11), 6509-6560. DOI: 10.1021/acs.chemrev.9b00073.

(45) *GaussView*; Semichem Inc.: Shawnee Mission, KS, 2016.

(46) Hanwell, M. D.; Curtis, D. E.; Lonie, D. C.; Vandermeersch, T.; Zurek, E.; Hutchison, G. R. Avogadro: an advanced semantic chemical editor, visualization, and analysis platform. *J Cheminform* **2012**, *4* (1), 17. DOI: 10.1186/1758-2946-4-17.

(47) *IQMol*; Australia National University, 2019.

(48) Pettersen, E. F.; Goddard, T. D.; Huang, C. C.; Meng, E. C.; Couch, G. S.; Croll, T. I.; Morris, J. H.; Ferrin, T. E. UCSF ChimeraX: Structure visualization for researchers, educators, and developers. *Protein Sci* **2020**. DOI: 10.1002/pro.3943.

(49) *Gaussian 16 Rev. C.01*; Wallingford, CT, 2016.

(50) Epifanovsky, E.; Gilbert, A. T. B.; Feng, X.; Lee, J.; Mao, Y.; Mardirossian, N.; Pokhilko, P.; White, A. F.; Coons, M. P.; Dempwolff, A. L.; et al. Software for the frontiers of quantum chemistry: An overview of developments in the Q-Chem 5 package. *The Journal of Chemical Physics* **2021**, *155* (8), 084801. DOI: 10.1063/5.0055522.

(51) Turney, J. M.; Simmonett, A. C.; Parrish, R. M.; Hohenstein, E. G.; Evangelista, F. A.; Fermann, J. T.; Mintz, B. J.; Burns, L. A.; Wilke, J. J.; Abrams, M. L.; et al. PSI4: an open-source ab initio electronic structure program. *Wiley Interdisciplinary Reviews-Computational Molecular Science* **2012**, *2* (4), 556-565. DOI: 10.1002/wcms.93.

(52) Ingman, V. M.; Schaefer, A. J.; Andreola, L. R.; Wheeler, S. E. QChASM: Quantum chemistry automation and structure manipulation. *WIREs Computational Molecular Science* **2020**, *11* (4), e1510. DOI: 10.1002/wcms.1510.

(53) https://github.com/QChASM/SEQCROW (accessed April 6, 2021).

(54) Wang, M.; Khan, S.; Miliordos, E.; Chen, M. Enantioselective Allenylation of Aldehydes via Brønsted Acid Catalysis. *Advanced Synthesis & Catalysis* **2018**, *360* (23), 4634-4639. DOI: 10.1002/adsc.201801080.

(55) Bannwarth, C.; Caldeweyher, E.; Ehlert, S.; Hansen, A.; Pracht, P.; Seibert, J.; Spicher, S.; Grimme, S. Extended tight-binding quantum chemistry methods. *WIREs Computational*

Molecular Science **2021**, *11* (2), e1493, https://doi.org/10.1002/wcms.1493. DOI: https://doi.org/10.1002/wcms.1493 (acccessed 2022/06/14).

(56) *Amber*; 2022. ambermd.org.

(57) Jeziorski, B.; Moszynski, R.; Szalewicz, K. Perturbation-Theory Approach to Intermolecular Potential-Energy Surfaces of Van-Der-Waals Complexes. *Chem. Rev.* **1994**, *94* (7), 1887-1930. DOI: DOI 10.1021/cr00031a008.

(58) Szalewicz, K. Symmetry-adapted perturbation theory of intermolecular forces. *Wiley Interdisciplinary Reviews-Computational Molecular Science* **2012**, *2* (2), 254-272. DOI: 10.1002/wcms.86.

(59) Hohenstein, E. G.; Sherrill, C. D. Density fitting of intramonomer correlation effects in symmetry-adapted perturbation theory. *J. Chem. Phys.* **2010**, *133* (1), 014101. DOI: 10.1063/1.3451077.

(60) Hohenstein, E. G.; Sherrill, C. D. Density fitting and Cholesky decomposition approximations in symmetry-adapted perturbation theory: Implementation and application to probe the nature of pi-pi interactions in linear acenes. *J. Chem. Phys.* **2010**, *132* (18), 184111. DOI: Artn 184111

10.1063/1.3426316.

(61) *AaronJr: Automate Any Reaction or Optimization, Normally Just Right*; University of Georgia: 2021.

(62) Ribeiro, R. F.; Marenich, A. V.; Cramer, C. J.; Truhlar, D. G. Use of solution-phase vibrational frequencies in continuum models for the free energy of solvation. *J. Phys. Chem. B* **2011**, *115* (49), 14556-14562. DOI: 10.1021/jp205508z.

(63) NIST Computational Chemistry Comparison and Benchmark Database, NIST Standard Reference Database Number 101 Release 21, August 2020, Editor: Russell D. Johnson III. http://cccbdb.nist.gov/

(64) Alecu, I. M.; Zheng, J.; Zhao, Y.; Truhlar, D. G. Computational Thermochemistry: Scale Factor Databases and Scale Factors for Vibrational Frequencies Obtained from Electronic Model Chemistries. *Journal of Chemical Theory and Computation* **2010**, *6* (9), 2872-2887. DOI: 10.1021/ct100326h.

(65) Kanchanakungwankul, S.; Zheng, J.; Alecu, I. M.; Lynch, B. J.; Zhao, Y.; Truhlar, D. G. *Database of Frequency Scale Factors for Electronic Model Chemistries*. 2018. https://comp.chem.umn.edu/freqscale/ (accessed 4/6/2021).

(66) Falivene, L.; Cao, Z.; Petta, A.; Serra, L.; Poater, A.; Oliva, R.; Scarano, V.; Cavallo, L. Towards the online computer-aided design of catalytic pockets. *Nat Chem* **2019**, *11* (10), 872-879. DOI: 10.1038/s41557-019-0319-5.

(67) Durand, D. J.; Fey, N. Computational Ligand Descriptors for Catalyst Design. *Chem Rev* **2019**, *119* (11), 6561-6594. DOI: 10.1021/acs.chemrev.8b00588.

(68) Durand, D. J.; Fey, N. Building a Toolbox for the Analysis and Prediction of Ligand and Catalyst Effects in Organometallic Catalysis. *Acc Chem Res* **2021**, *54* (4), 837-848. DOI: 10.1021/acs.accounts.0c00807.

(69) Verloop, A. In *Drug Design*, Ariens, E. J. Ed.; Vol. III; Academic Press, 1976.

(70) Cavallo, L.; Correa, A.; Costabile, C.; Jacobsen, H. Steric and electronic effects in the bonding of N-heterocyclic ligands to transition metals. *Journal of Organometallic Chemistry* **2005**, *690* (24-25), 5407-5413. DOI: 10.1016/j.jorganchem.2005.07.012.

(71) Poater, A.; Ragone, F.; Giudice, S.; Costabile, C.; Dorta, R.; Nolan, S. P.; Cavallo, L. Thermodynamics of N-Heterocyclic Carbene Dimerization: The Balance of Sterics and Electronics. *Organometallics* **2008**, *27* (12), 2679-2681. DOI: 10.1021/om8001119.

(72) Poater, A.; Cosenza, B.; Correa, A.; Giudice, S.; Ragone, F.; Scarano, V.; Cavallo, L. SambVca: A Web Application for the Calculation of the Buried Volume of N-Heterocyclic Carbene Ligands. *European Journal of Inorganic Chemistry* **2009**, *2009* (13), 1759-1766. DOI: 10.1002/ejic.200801160.

(73) *Mol2Mol*; University of Debrecen: Debrecen, Hungary, 2011. https://www.gunda.hu/mol2mol.

(74) *Sterimol.py*; 2017. https://github.com/bobbypaton/Sterimol.

(75) Virtanen, P.; Gommers, R.; Oliphant, T. E.; Haberland, M.; Reddy, T.; Cournapeau, D.; Burovski, E.; Peterson, P.; Weckesser, W.; Bright, J.; et al. SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nat Methods* **2020**, *17* (3), 261-272. DOI: 10.1038/s41592-019-0686-2.

(76) E, W.; Ren, W.; Vanden-Eijnden, E. String method for the study of rare events. *Physical Review B* **2002**, *66* (5), 052301. DOI: 10.1103/PhysRevB.66.052301.

(77) Peters, B.; Heyden, A.; Bell, A. T.; Chakraborty, A. A growing string method for determining transition states: Comparison to the nudged elastic band and string methods. *The Journal of Chemical Physics* **2004**, *120* (17), 7877-7886. DOI: 10.1063/1.1691018 (acccessed 2022/06/16).

(78) Denzel, A.; Haasdonk, B.; Kästner, J. Gaussian Process Regression for Minimum Energy Path Optimization and Transition State Search. *The Journal of Physical Chemistry A* **2019**, *123* (44), 9600-9611. DOI: 10.1021/acs.jpca.9b08239.

(79) Ásgeirsson, V.; Birgisson, B. O.; Bjornsson, R.; Becker, U.; Neese, F.; Riplinger, C.; Jónsson, H. Nudged Elastic Band Method for Molecular Reactions Using Energy-Weighted Springs Combined with Eigenvector Following. *Journal of Chemical Theory and Computation* **2021**, *17* (8), 4929-4945. DOI: 10.1021/acs.jctc.1c00462.

(80) Kabsch, W. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A* **1976**, *32* (5), 922-923, https://doi.org/10.1107/S0567739476001873. DOI: https://doi.org/10.1107/S0567739476001873 (acccessed 2022/06/16).

(81) Birkholz, A. B.; Schlegel, H. B. Using bonding to guide transition state optimization. *J Comput Chem* **2015**, *36* (15), 1157-1166. DOI: 10.1002/jcc.23910.

(82) Zimmerman, P. Reliable Transition State Searches Integrated with the Growing String Method. *J Chem Theory Comput* **2013**, *9* (7), 3043-3050. DOI: 10.1021/ct400319w.

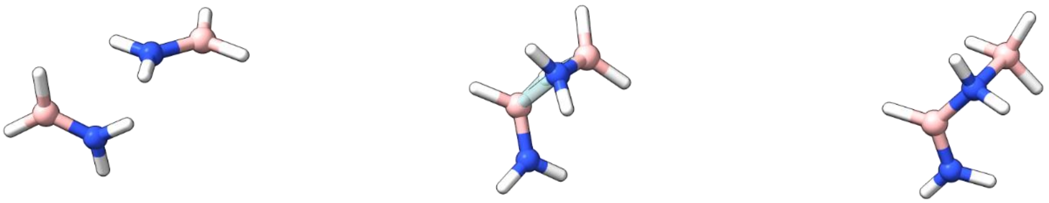
(83) Zimmerman, P. M. Navigating molecular space for reaction mechanisms: an efficient, automated procedure. *Molecular Simulation* **2015**, *41* (1-3), 43-54. DOI: 10.1080/08927022.2014.894999.

(84) Pendleton, I. M.; Pérez-Temprano, M. H.; Sanford, M. S.; Zimmerman, P. M. Experimental and Computational Assessment of Reactivity and Mechanism in C(sp3)–N Bond-Forming Reductive Elimination from Palladium(IV). *Journal of the American Chemical Society* **2016**, *138* (18), 6049-6060. DOI: 10.1021/jacs.6b02714.

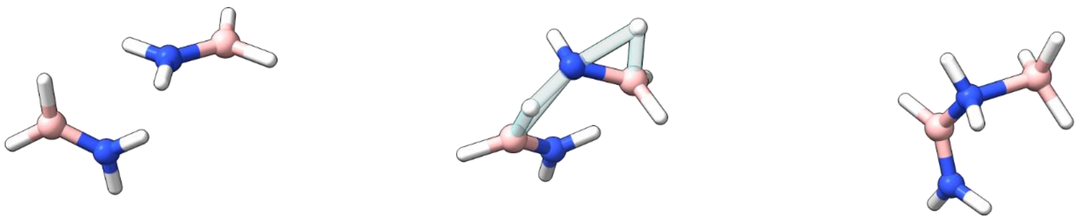
APPENDIX A

RAVEN BENCHMARKING SET

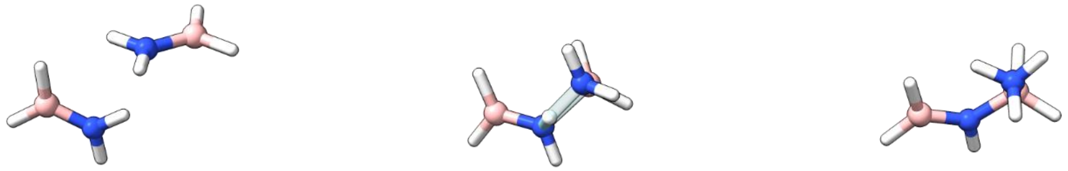
The following structures are the reactant (left), transition state structure (middle), and product (right) from the set used to benchmark Raven. If a reaction is marked as “failed”, the TSS shown is Raven’s highest energy guess. Otherwise, it is the TSS optimized from Raven’s guess. Select atoms are labeled if the reactant and product are the same molecules.



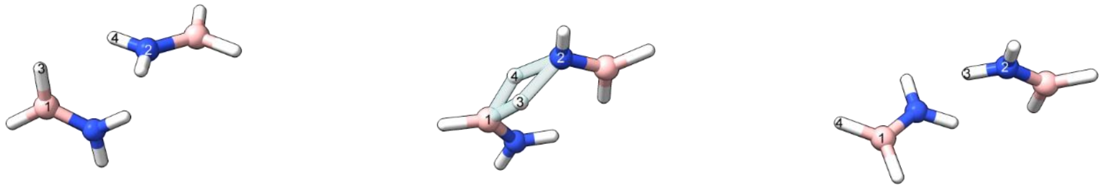
reaction 1



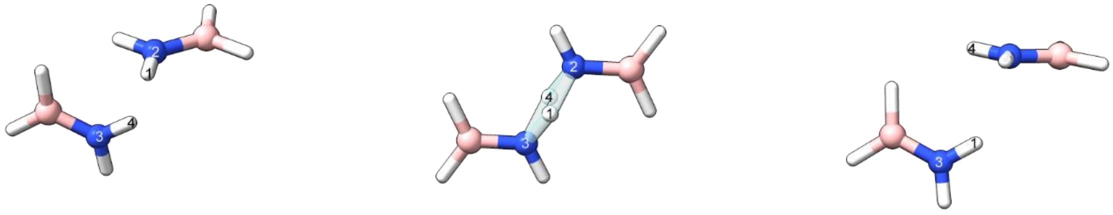
reaction 2



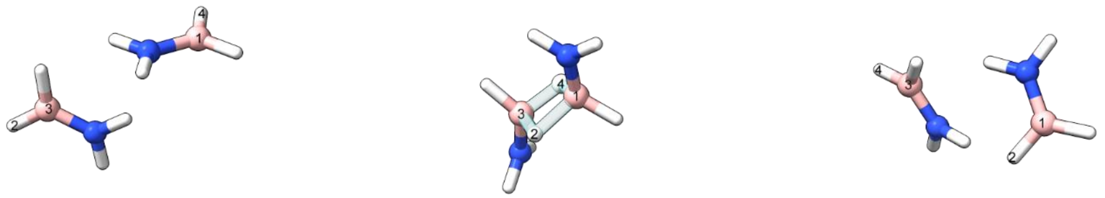
reaction 3



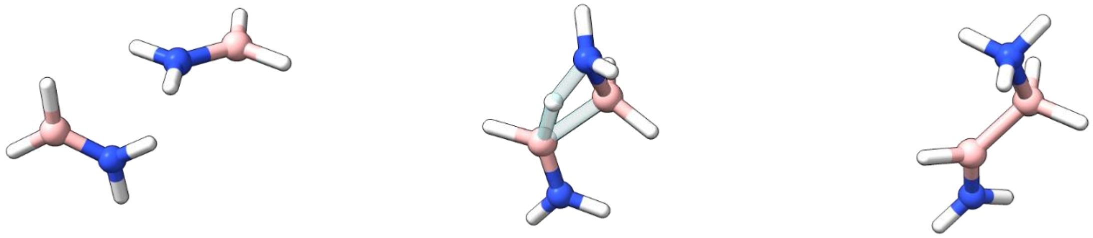
reaction 4



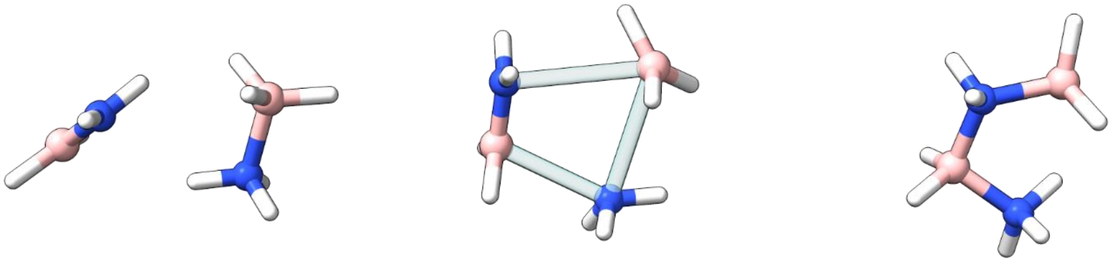
reaction 5



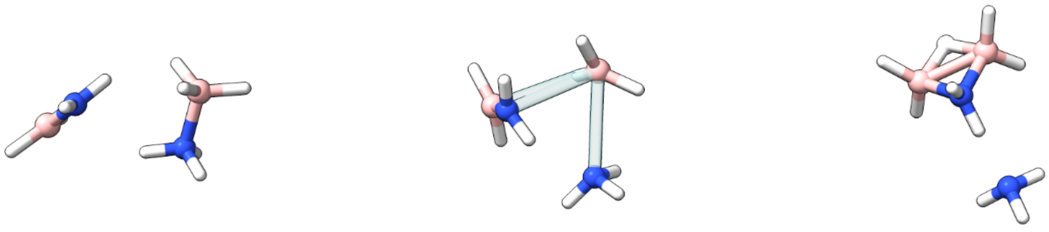
reaction 6: failed



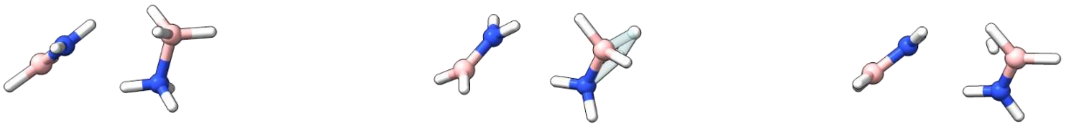
reaction 7



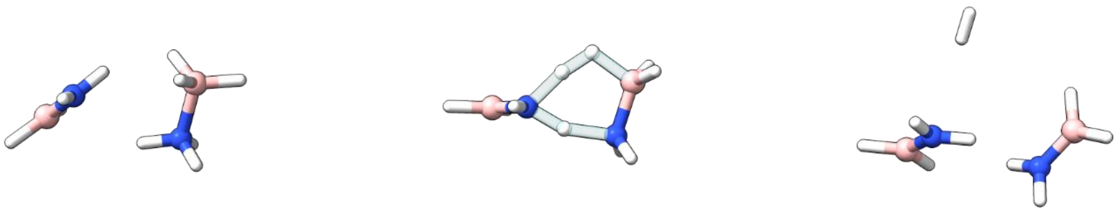
reaction 8: failed



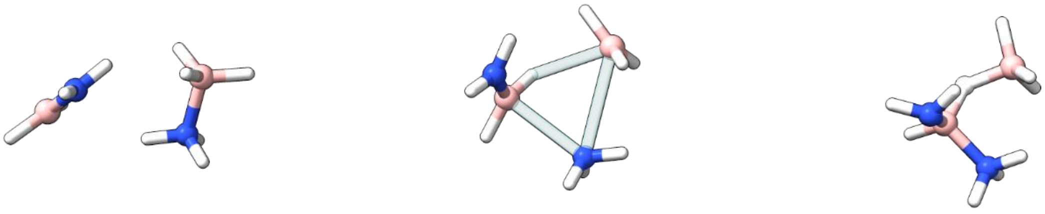
reaction 9: failed



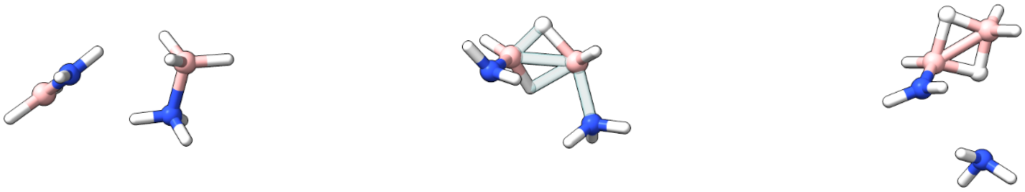
reaction 10



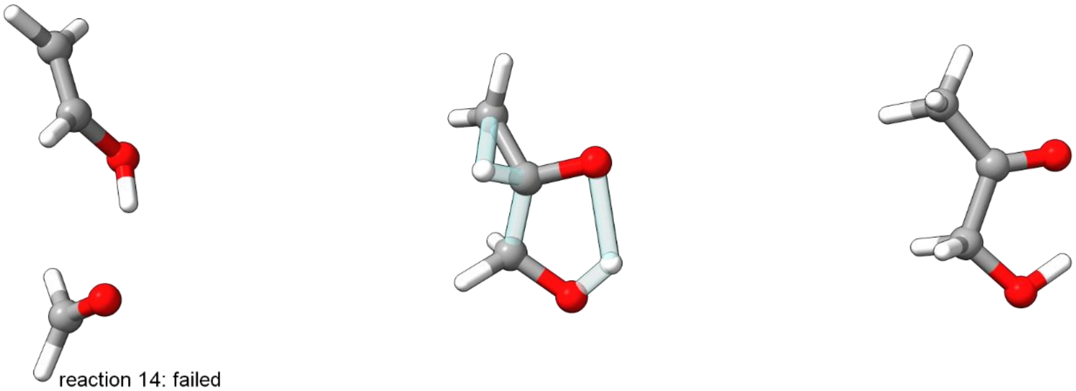
reaction 11



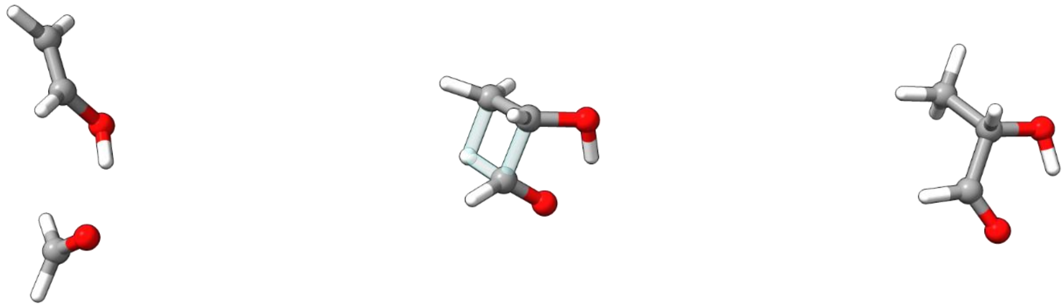
reaction 12



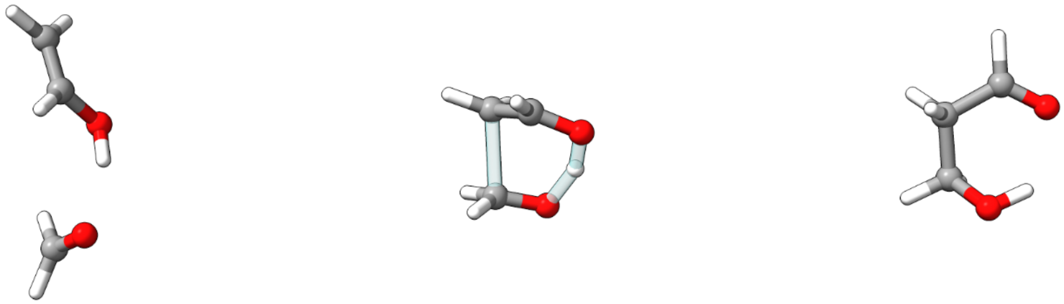
reaction 13: failed



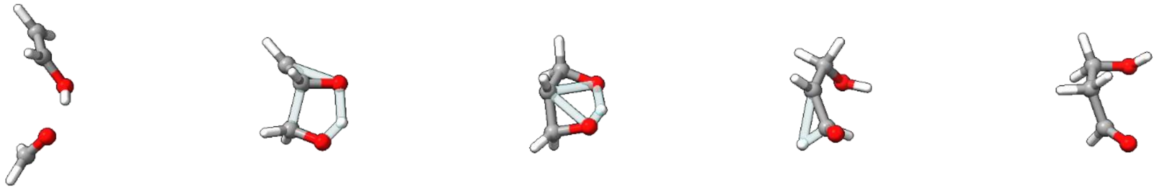
reaction 14: failed



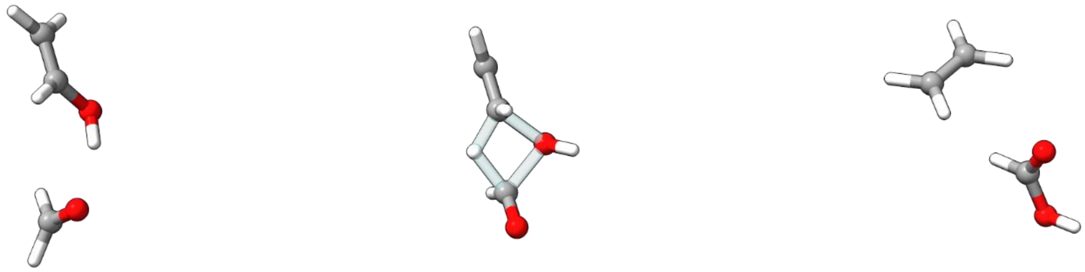
reaction 15



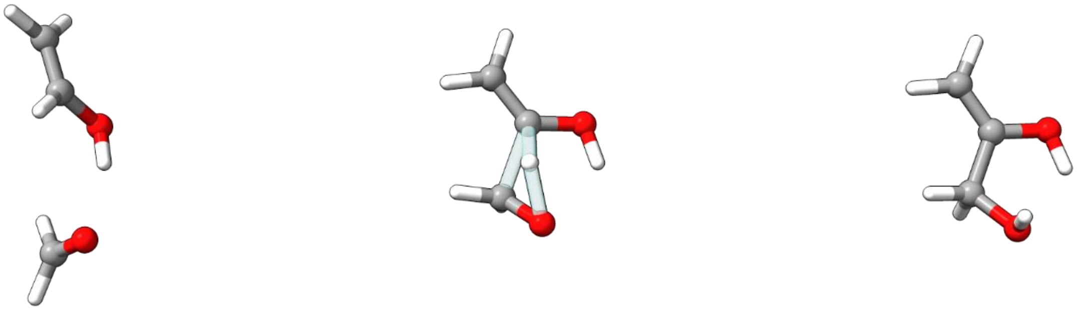
reaction 16



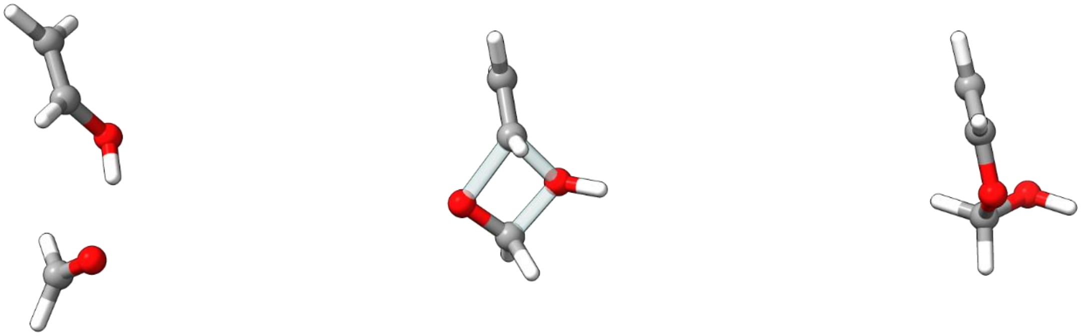
reaction 17



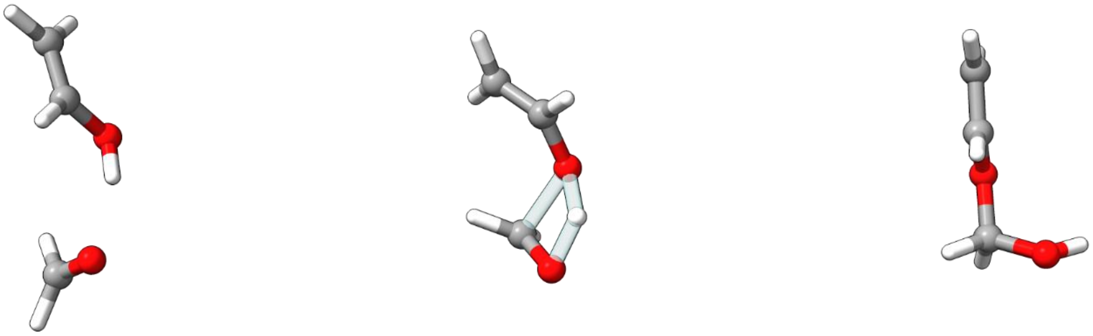
reaction 18



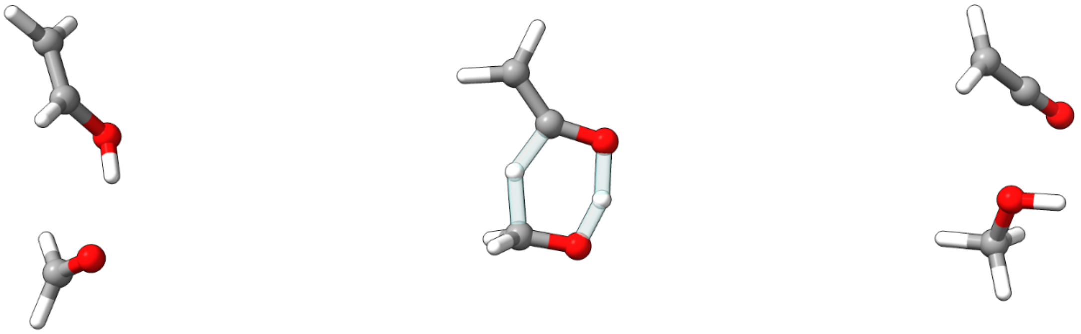
reaction 19



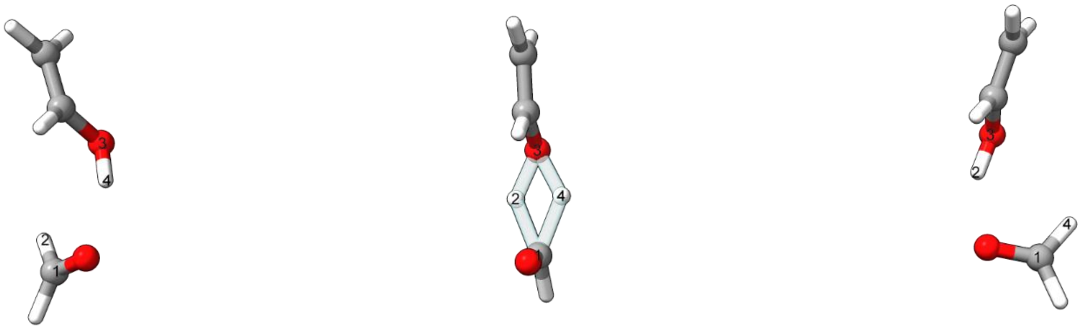
reaction 20



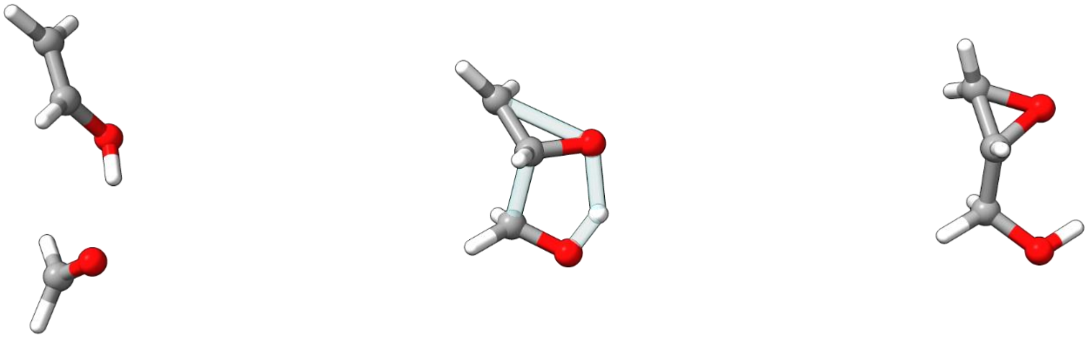
reaction 21



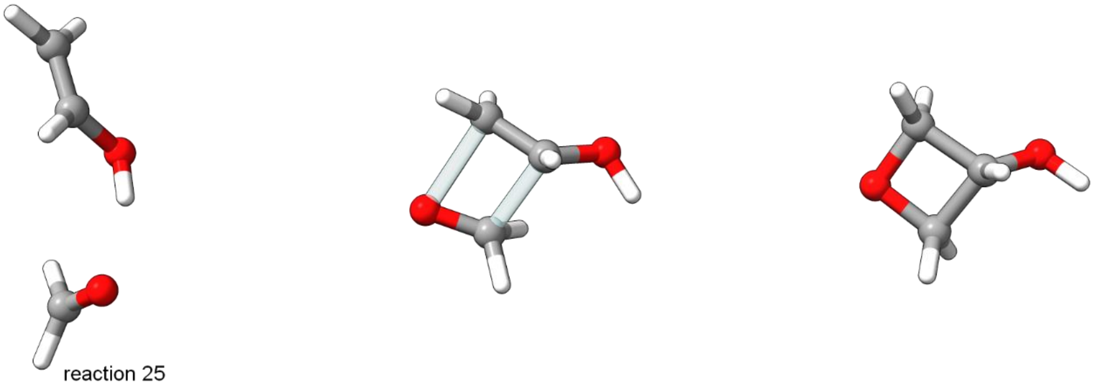
reaction 22



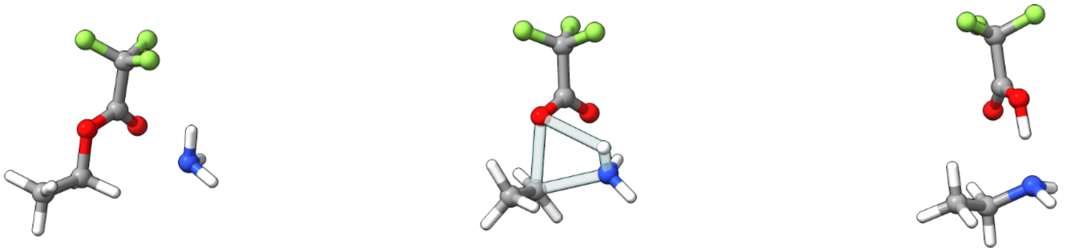
reaction 23



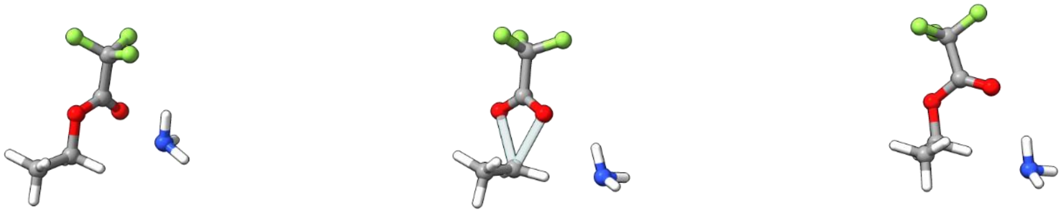
reaction 24



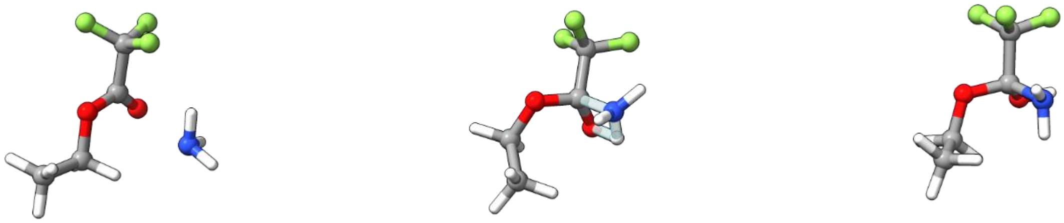
reaction 25



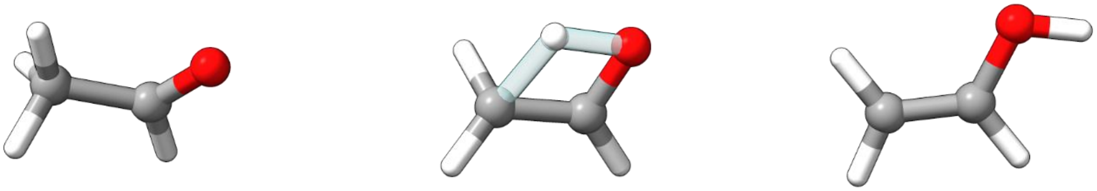
reaction 26: failed



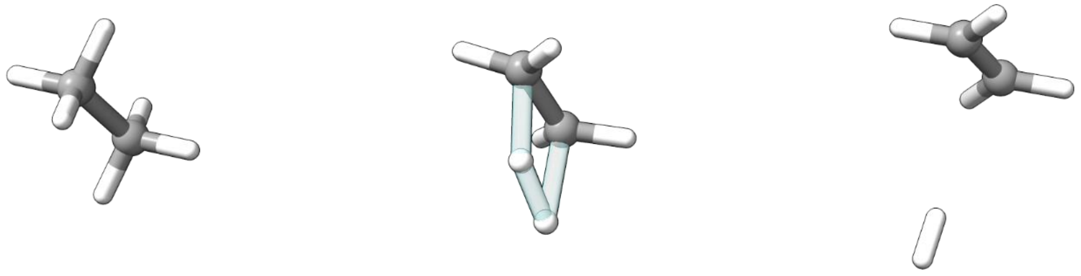
reaction 27



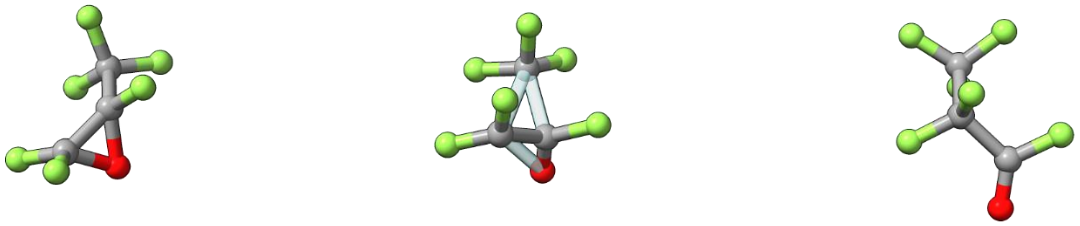
reaction 28



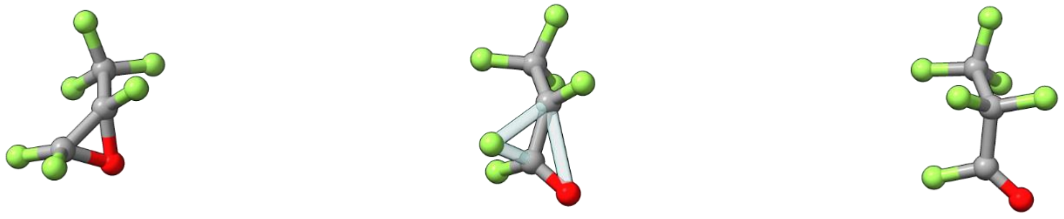
reaction 29



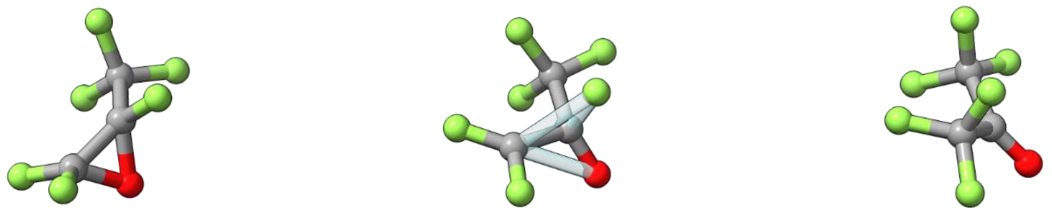
reaction 30



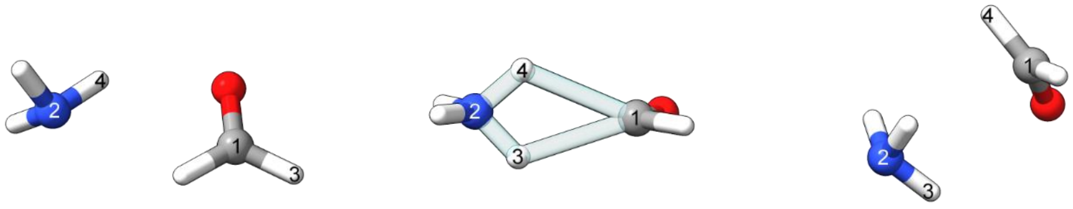
reaction 31



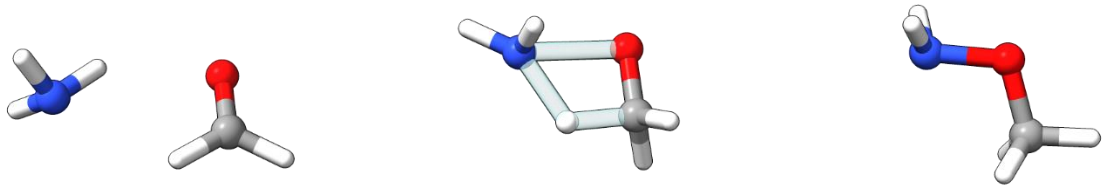
reaction 32



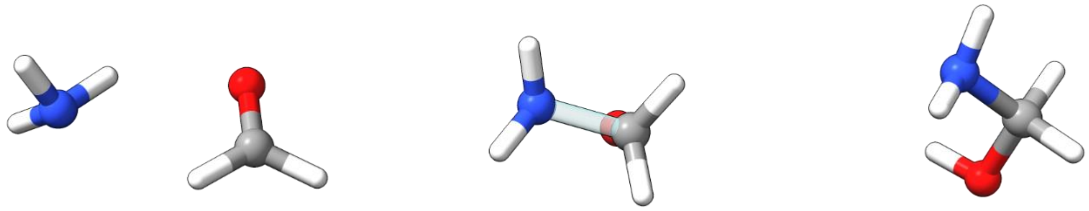
reaction 33



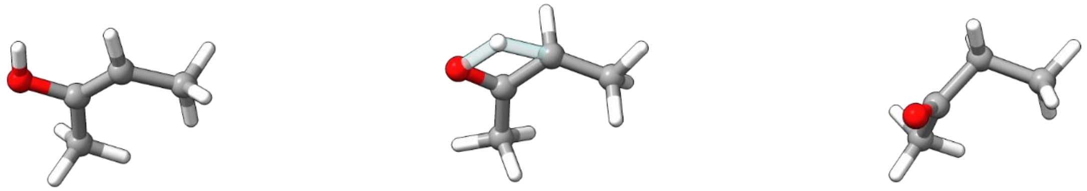
reaction 34



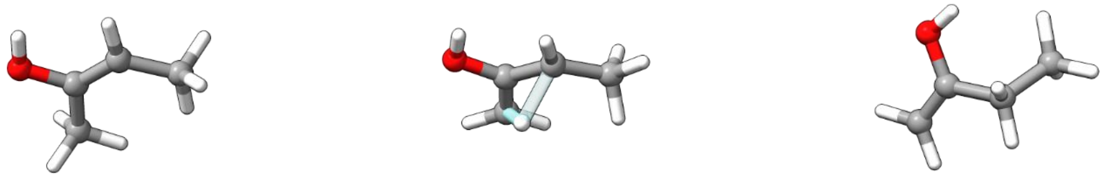
reaction 35



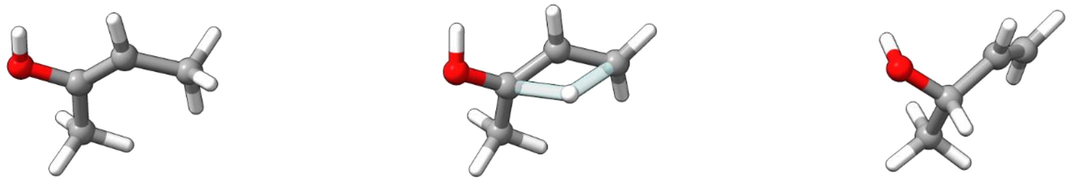
reaction 36



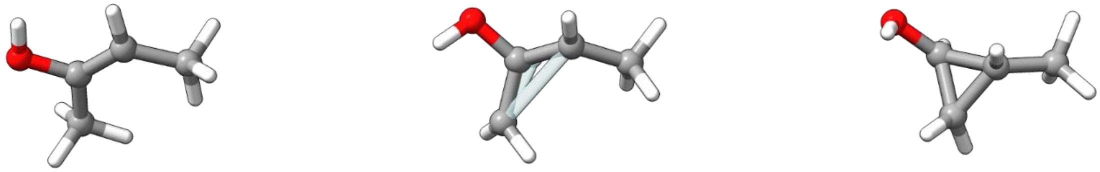
reaction 37



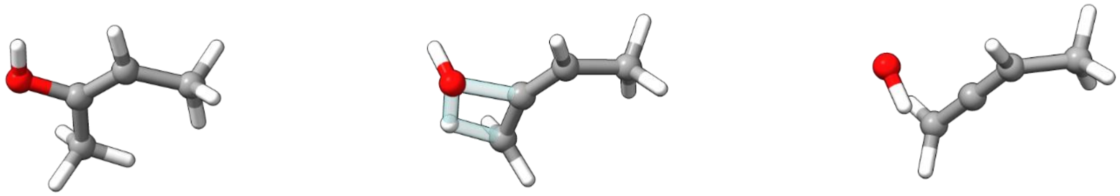
reaction 38



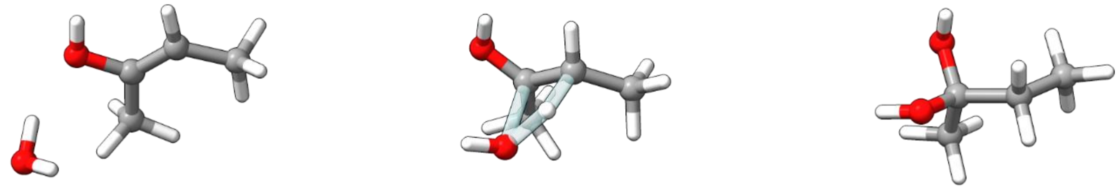
reaction 39: failed



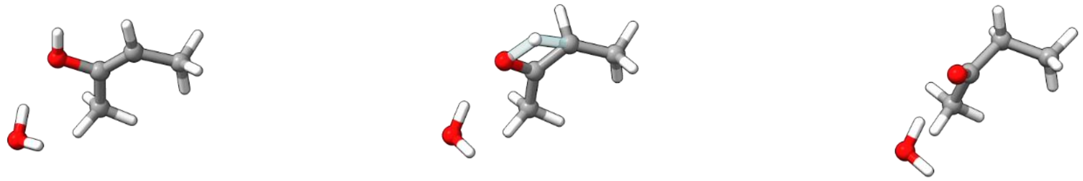
reaction 40: failed



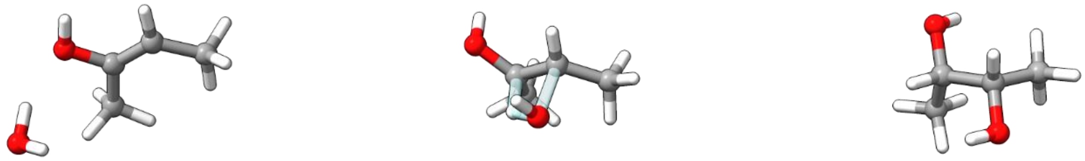
reaction 41



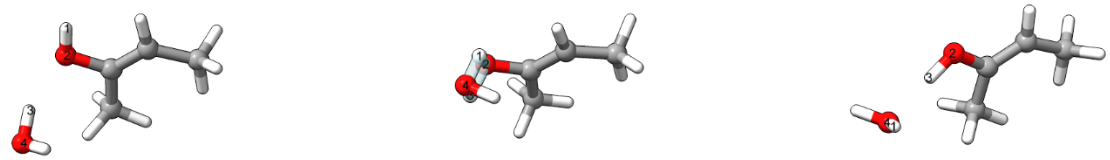
reaction 42



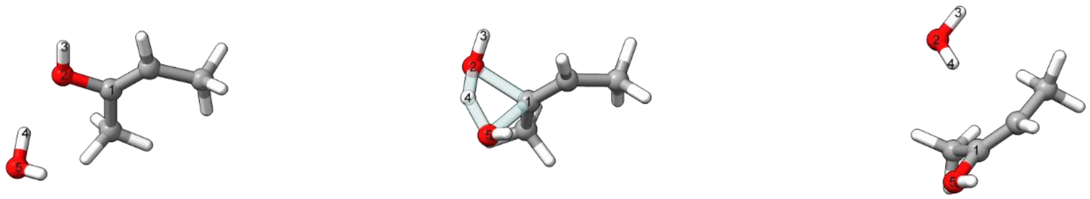
reaction 43: failed



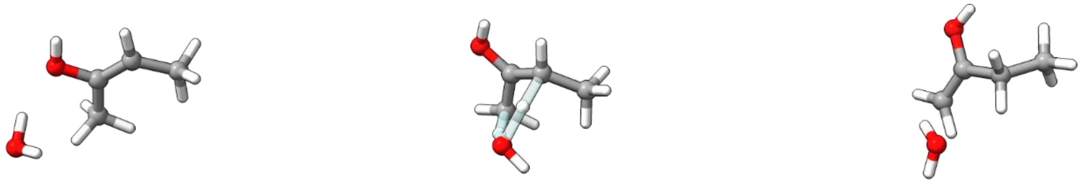
reaction 44



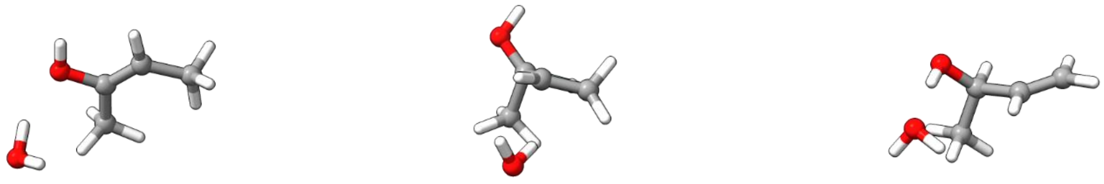
reaction 45



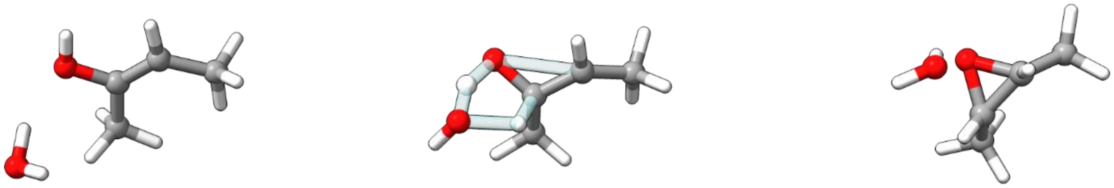
reaction 46



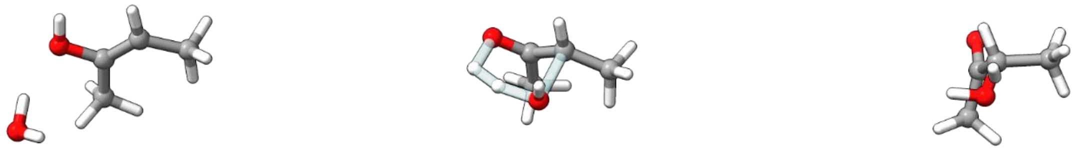
reaction 47



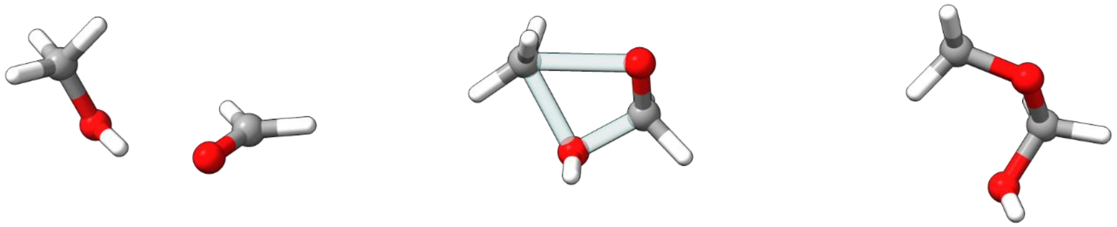
reaction 48: failed



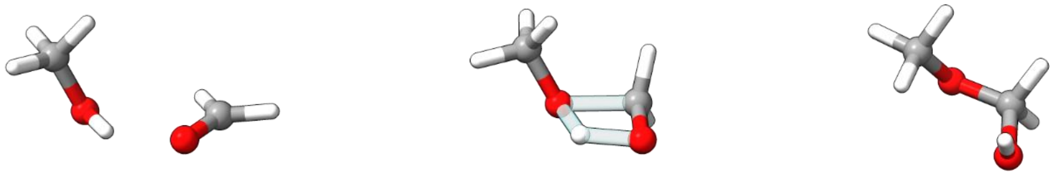
reaction 49: failed



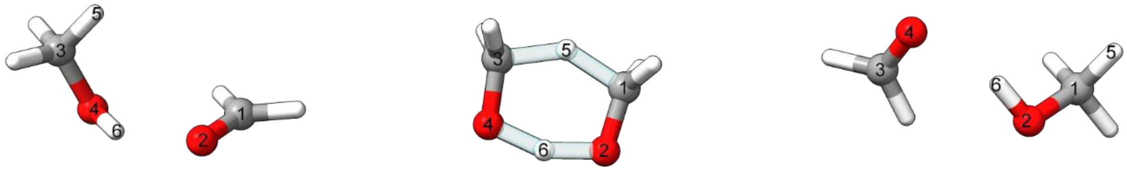
reaction 50



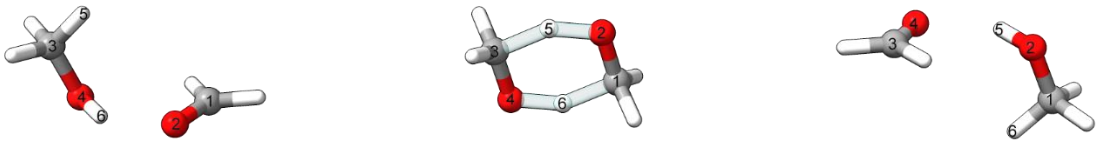
reaction 51



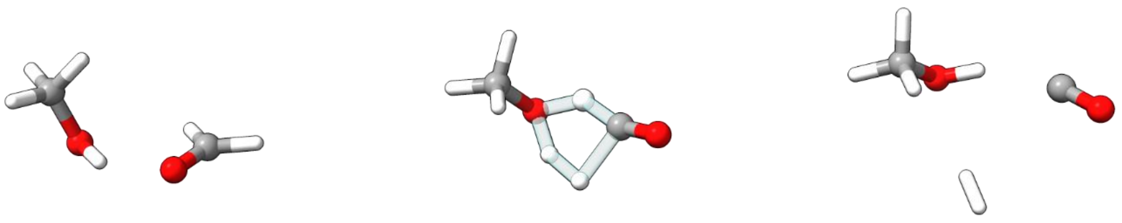
reaction 52



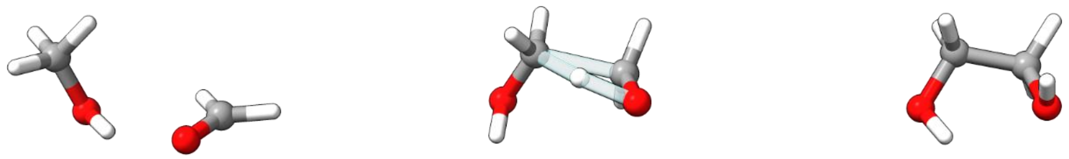
reaction 53



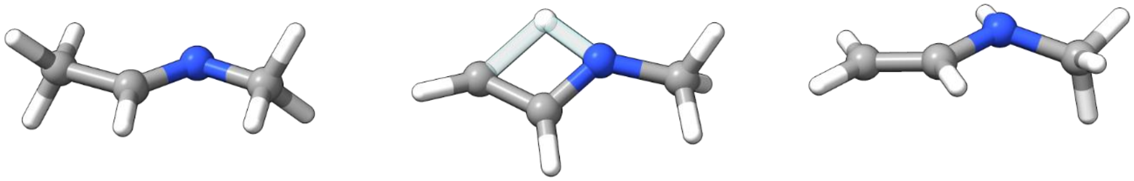
reaction 54



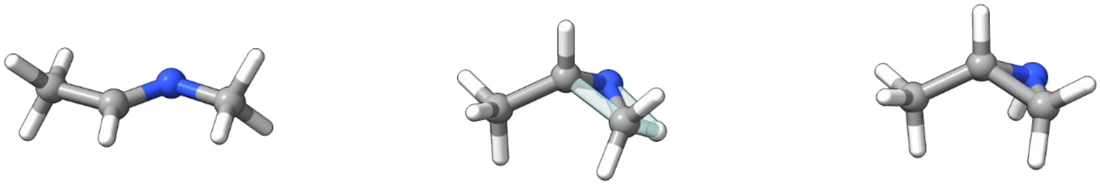
reaction 55: failed



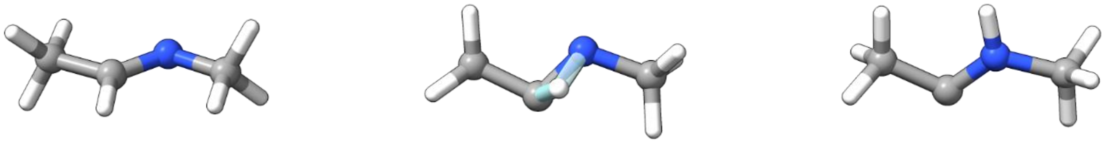
reaction 56



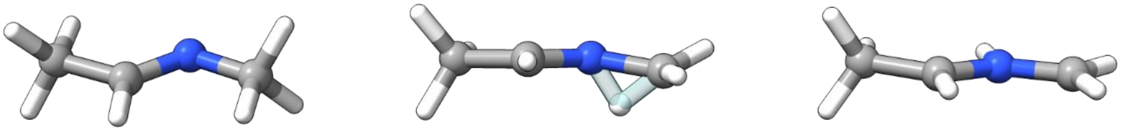
reaction 57



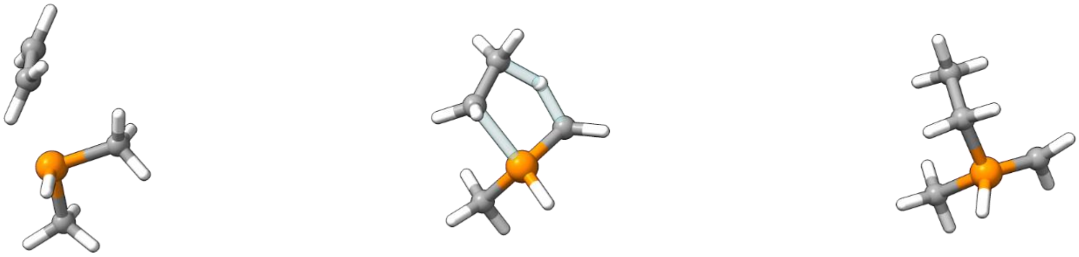
reaction 58



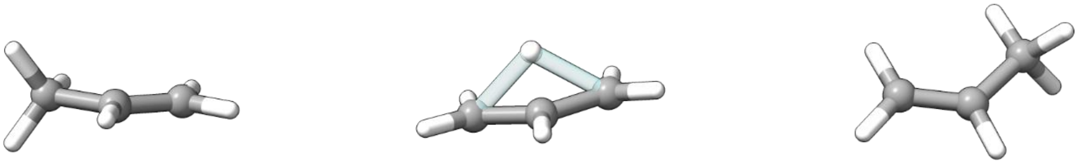
reaction 59



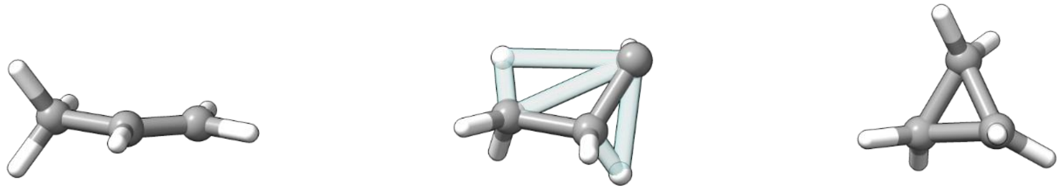
reaction 60



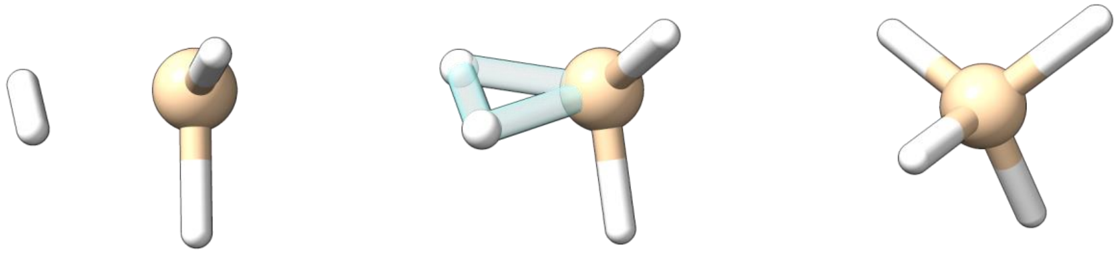
reaction 61



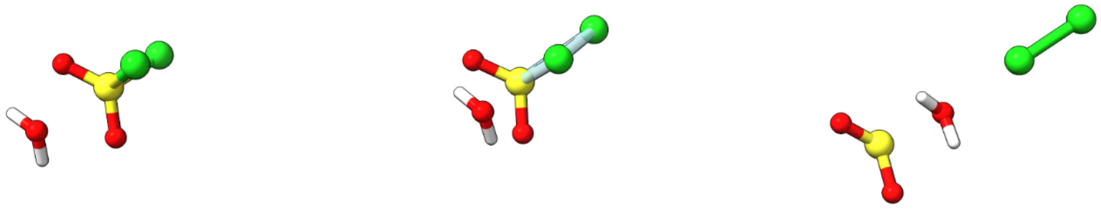
reaction 62



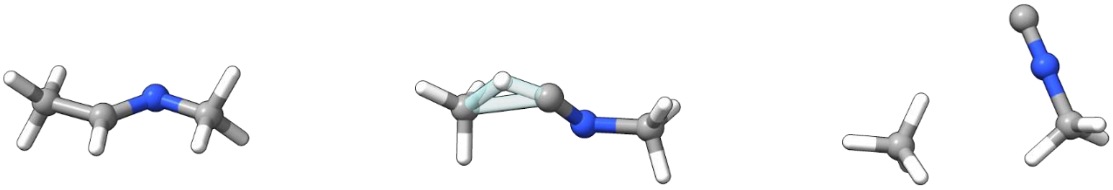
reaction 63: failed



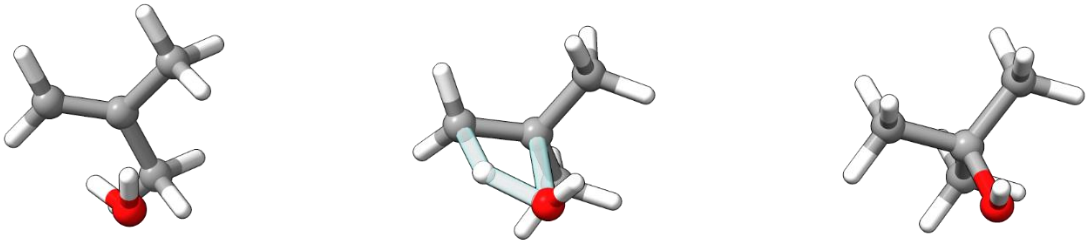
reaction 64



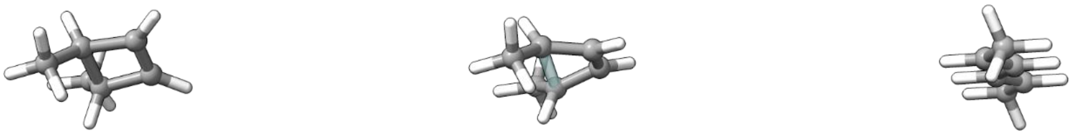
reaction 65



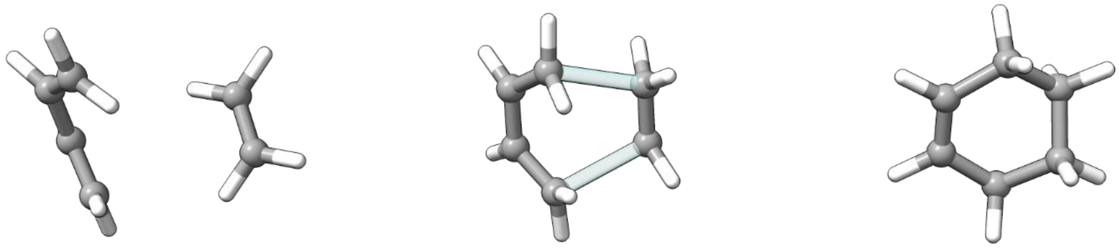
reaction 66



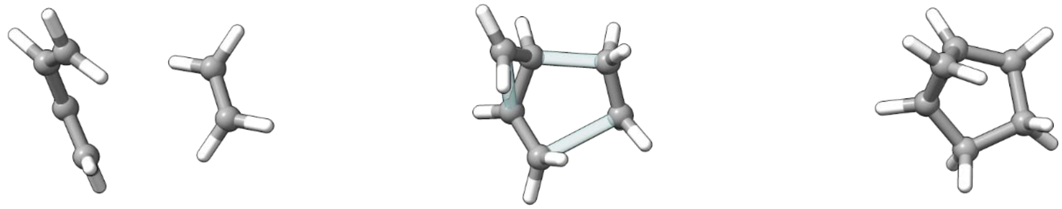
reaction 67



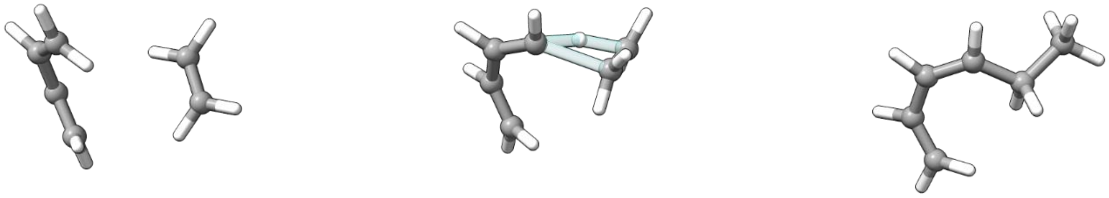
reaction 68



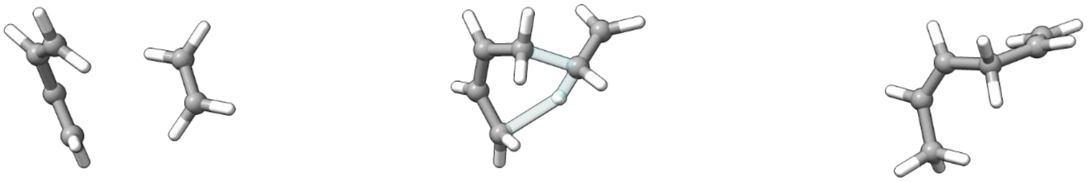
reaction 69



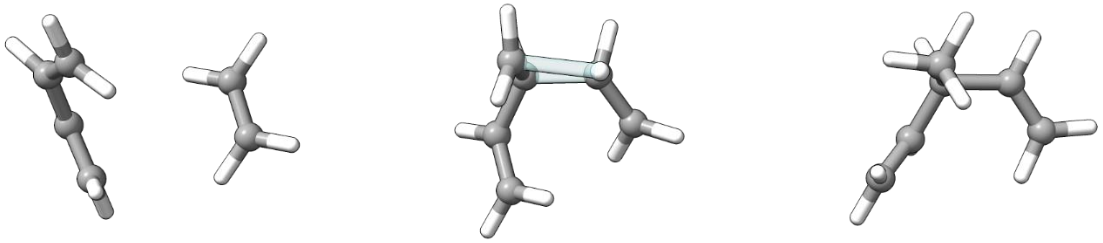
reaction 70



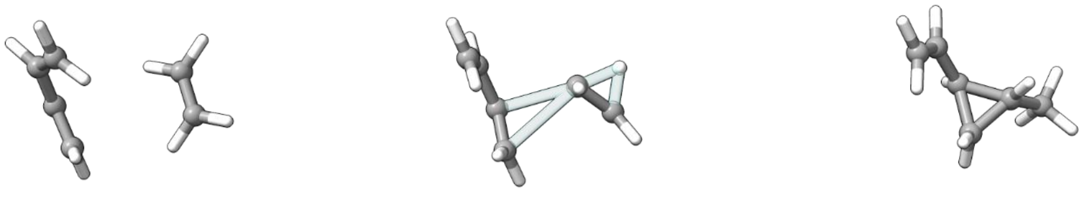
reaction 71



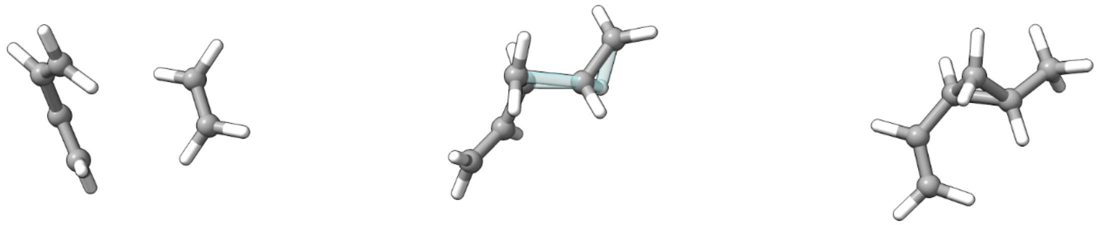
reaction 72



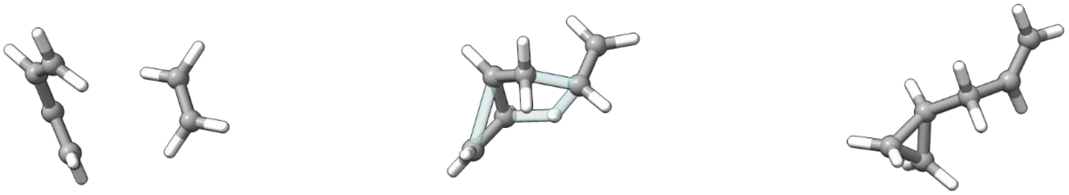
reaction 73



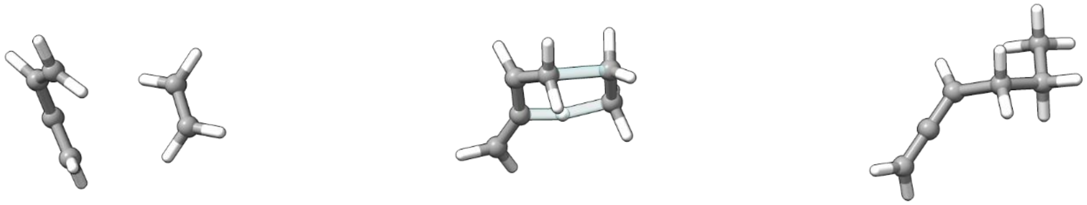
reaction 74: failed



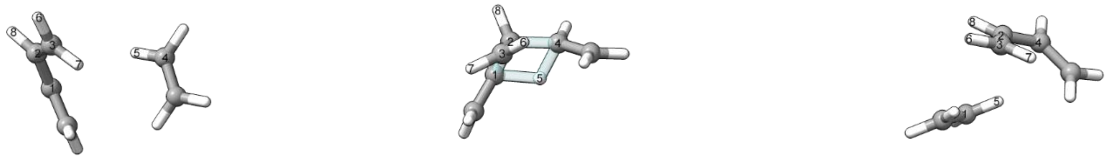
reaction 75



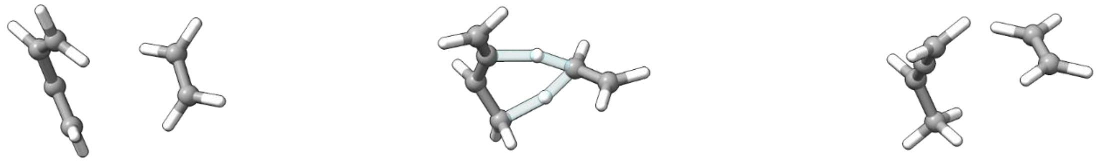
reaction 76: failed



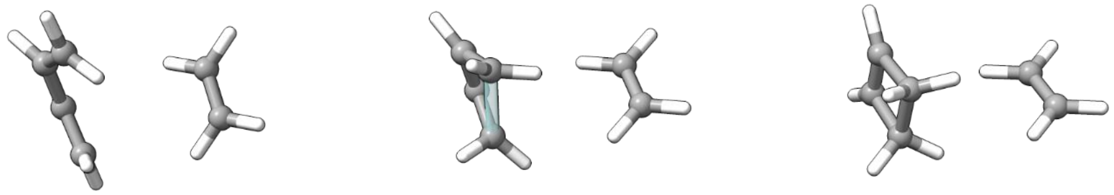
reaction 77



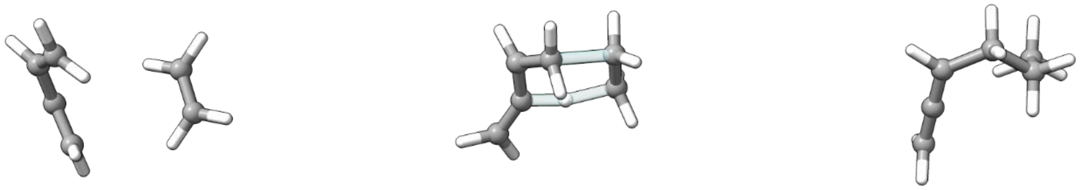
reaction 78



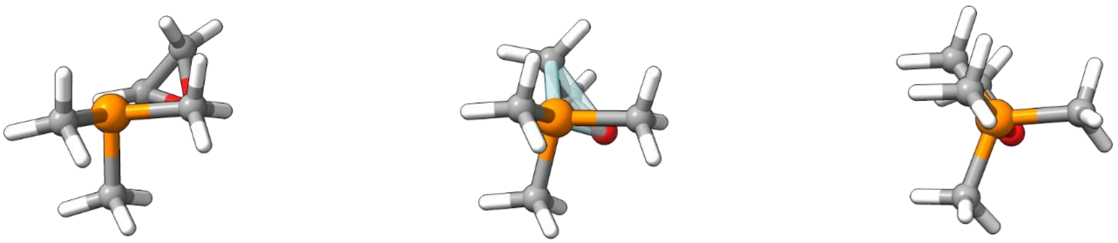
reaction 79



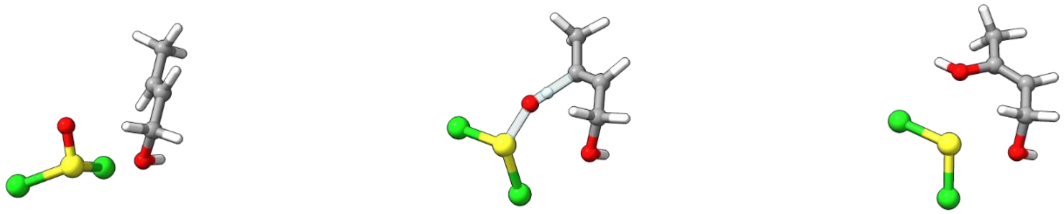
reaction 80



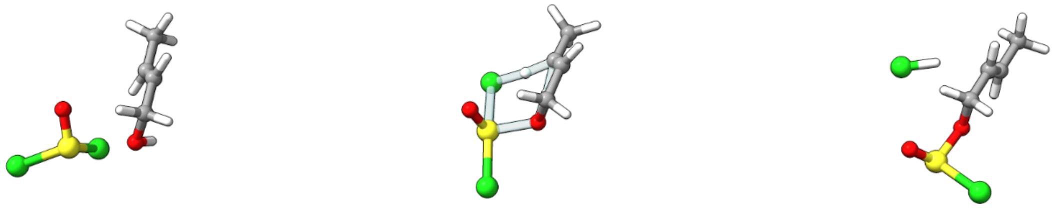
reaction 81



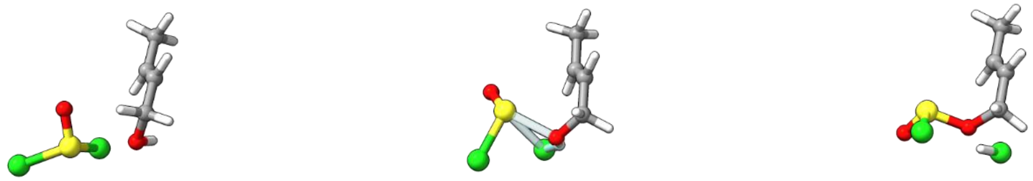
reaction 82



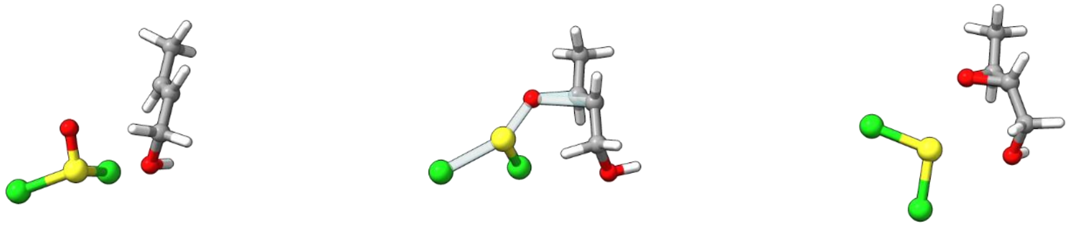
reaction 83: failed



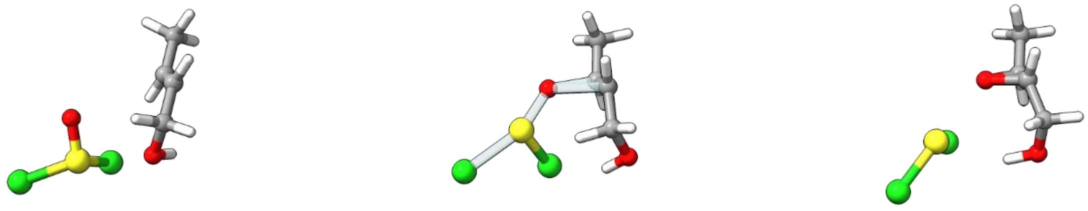
reaction 84: failed



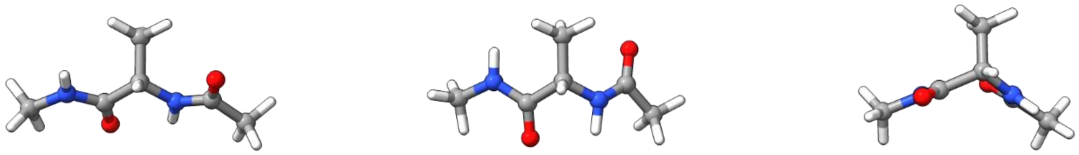
reaction 85



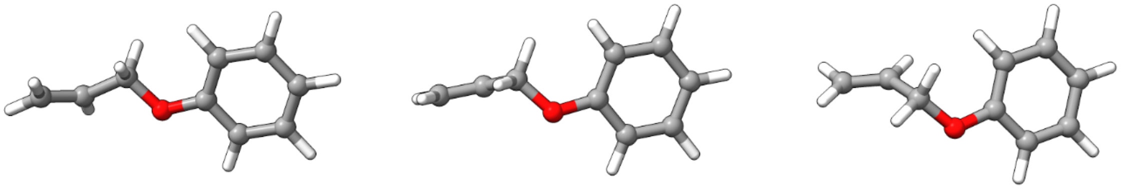
reaction 86



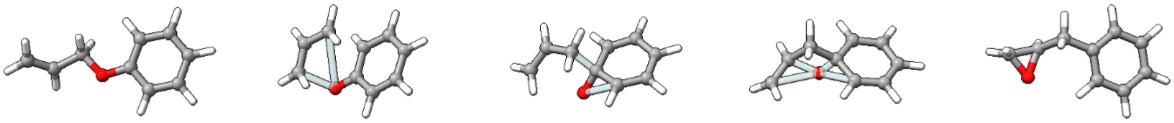
reaction 87



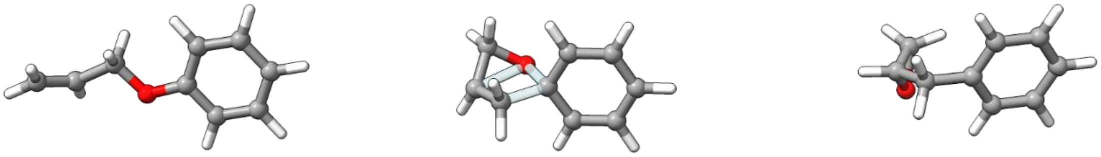
reaction 88



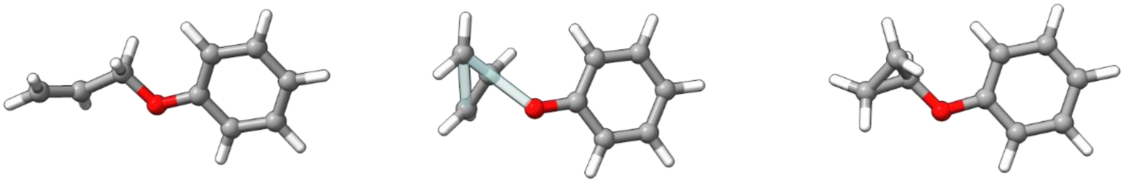
reaction 89



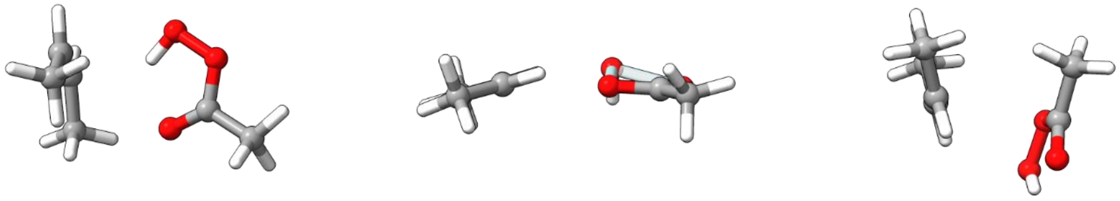
reaction 90



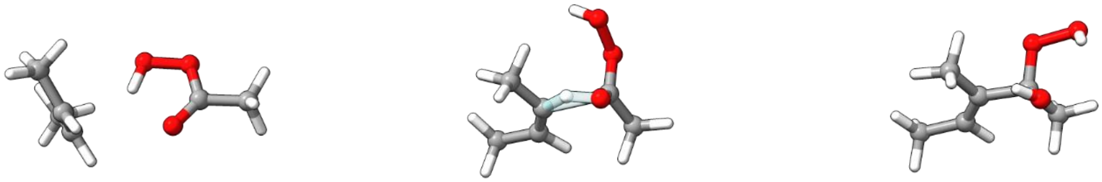
reaction 91



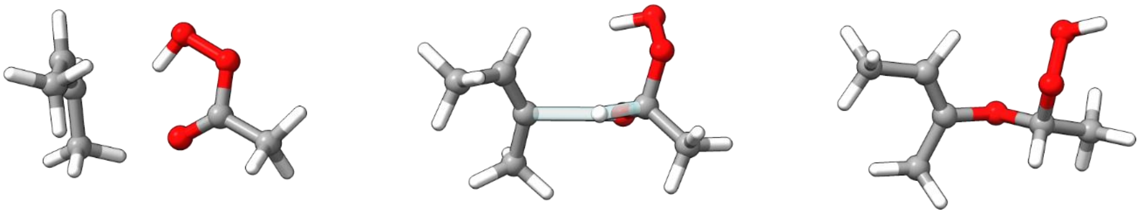
reaction 92



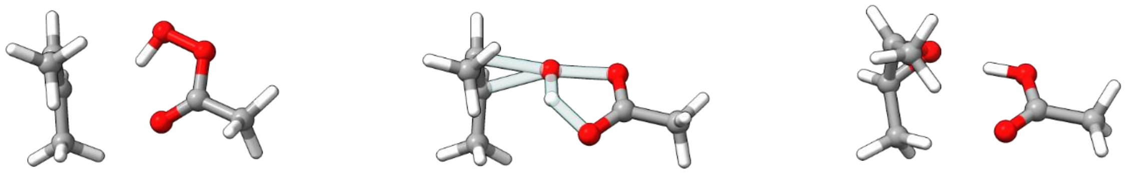
reaction 93: failed



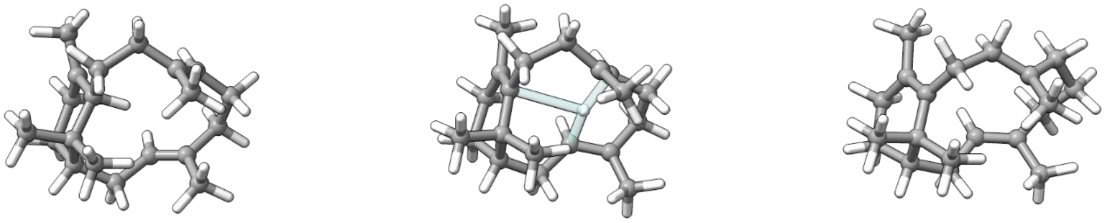
reaction 94: failed



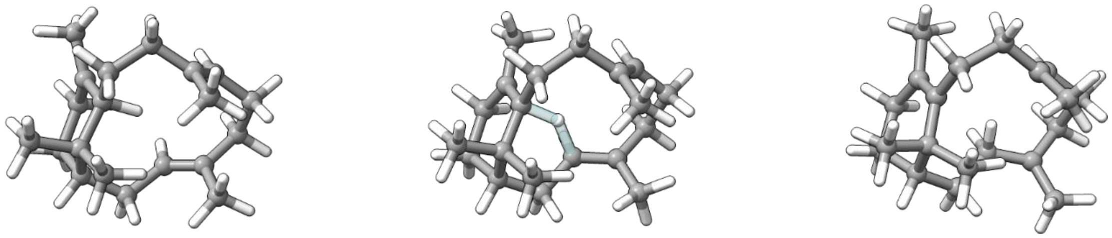
reaction 95: failed



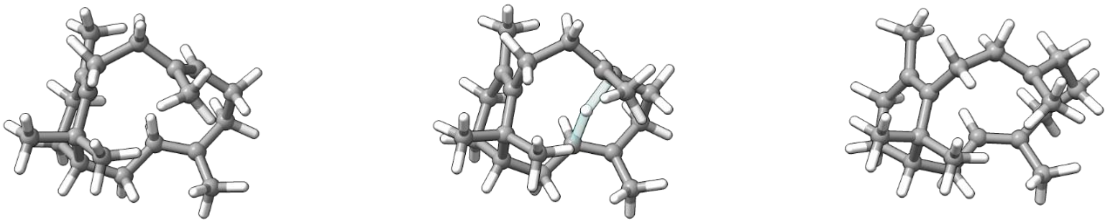
reaction 96



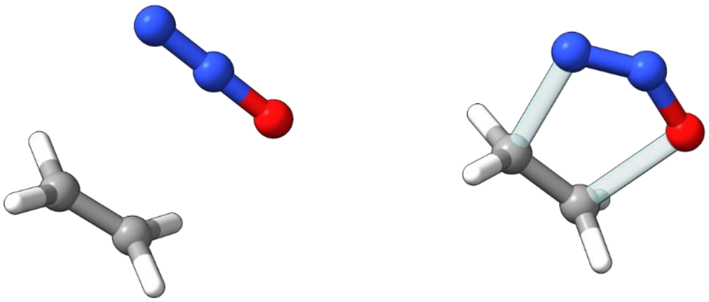
reaction 97



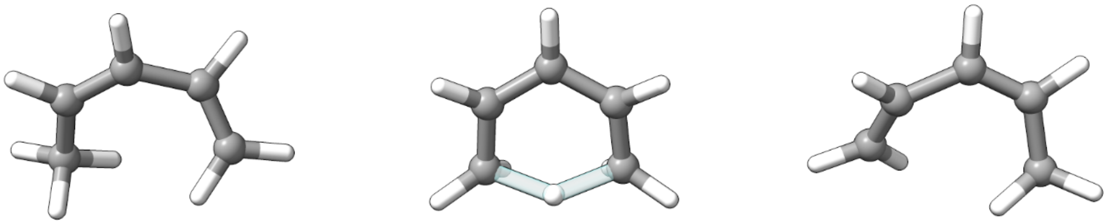
reaction 98



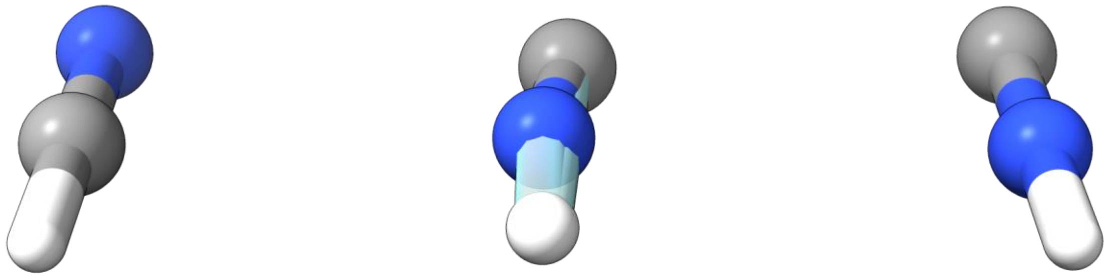
reaction 99



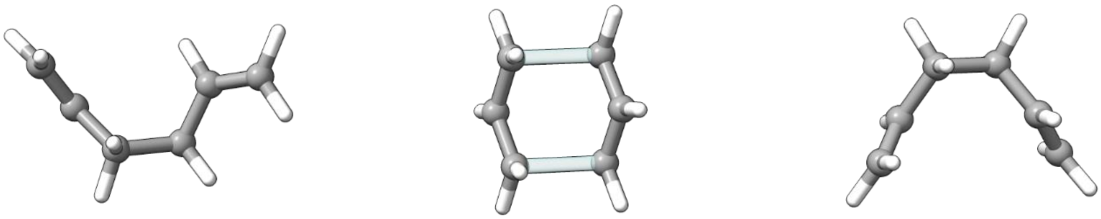
reaction 100



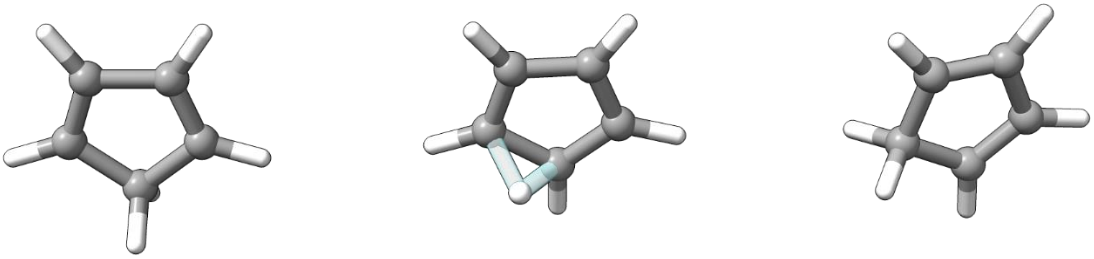
reaction 101



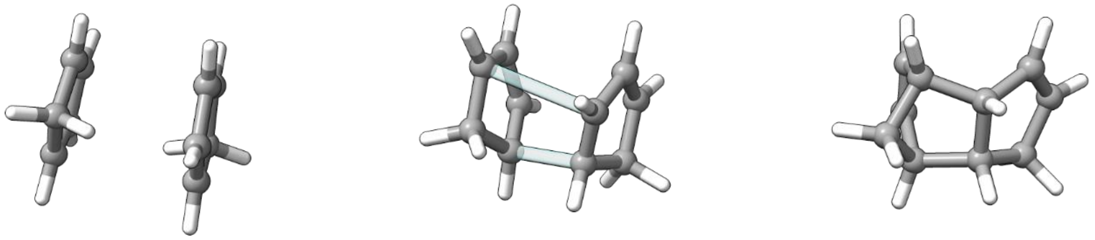
reaction 102: failed



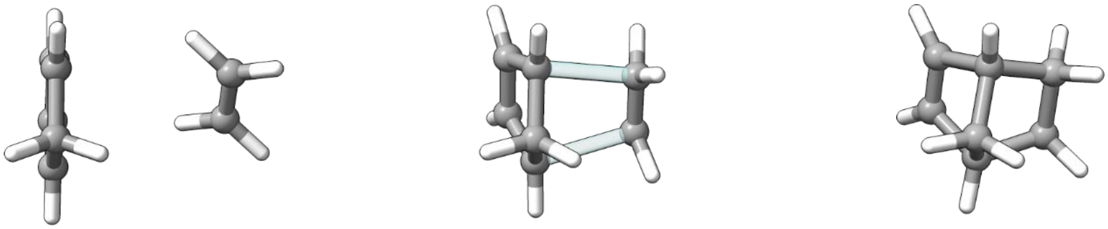
reaction 103



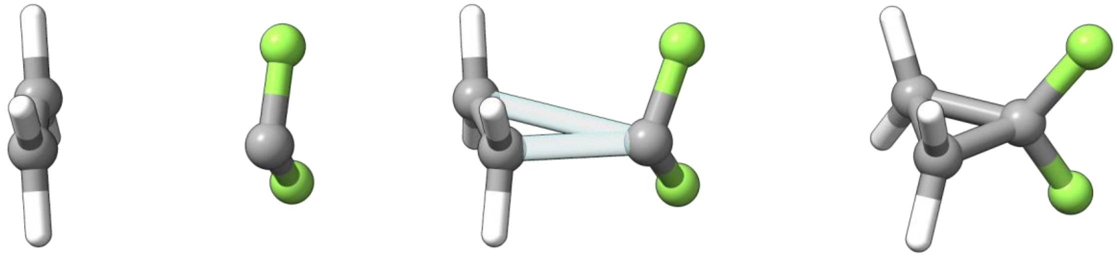
reaction 104



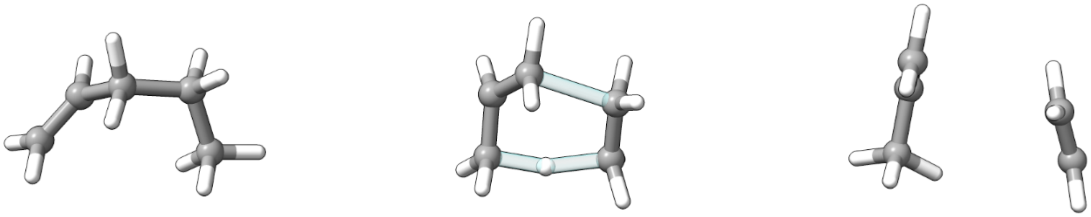
reaction 105



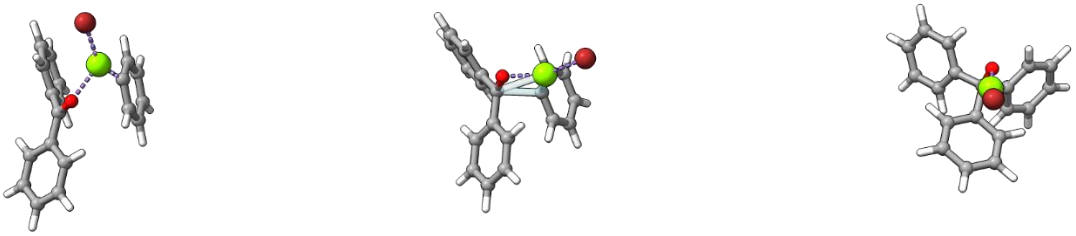
reaction 106



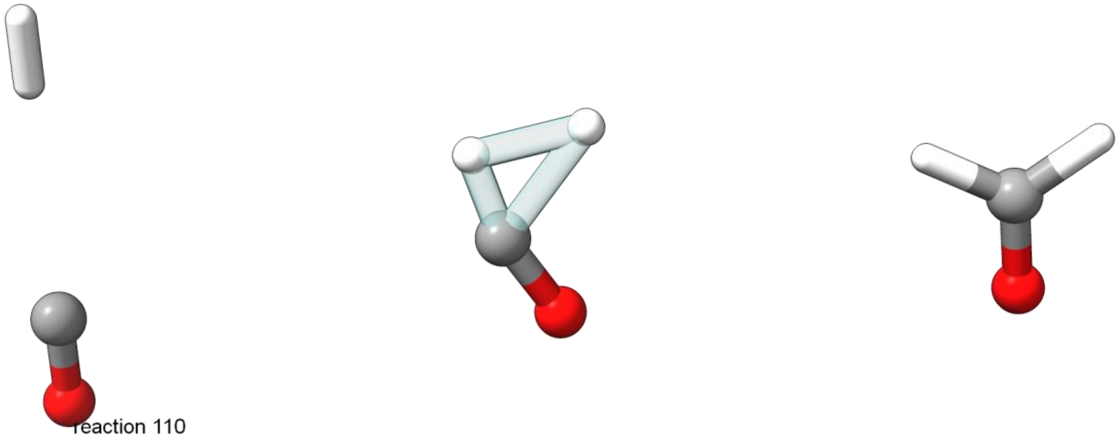
reaction 107



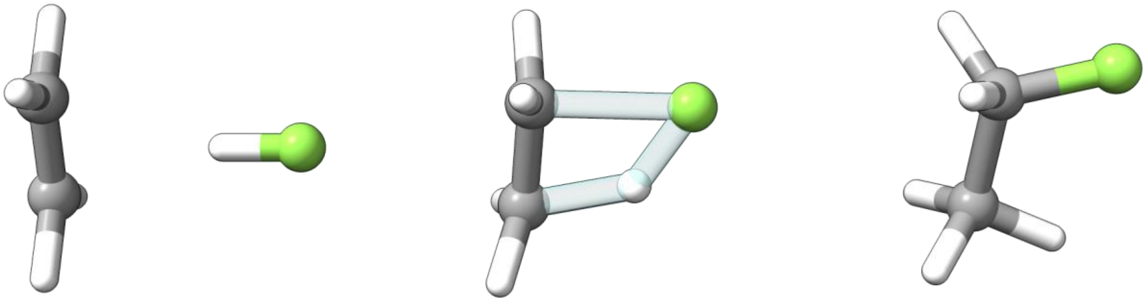
reaction 108



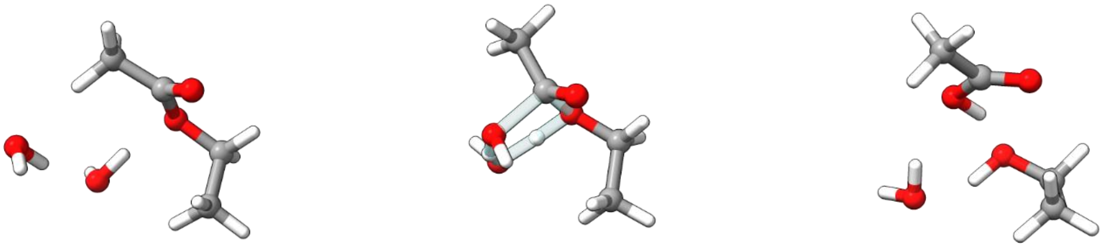
reaction 109



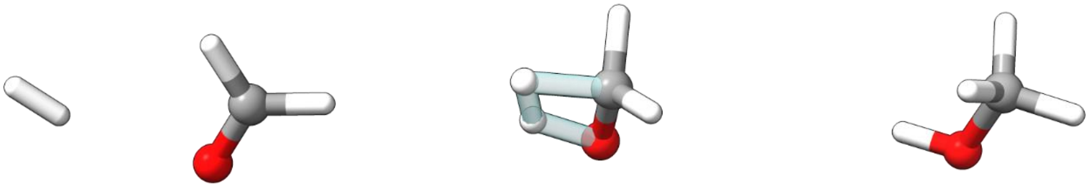
reaction 110



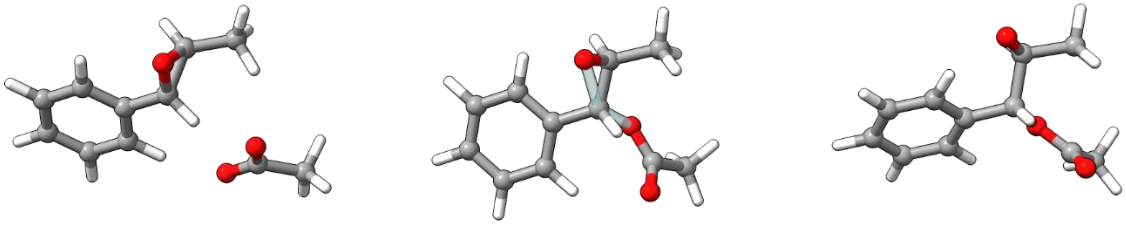
reaction 111



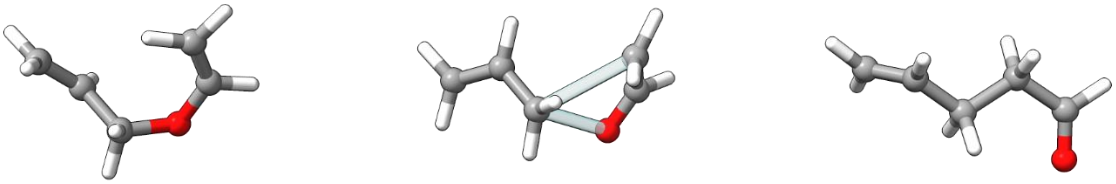
reaction 112



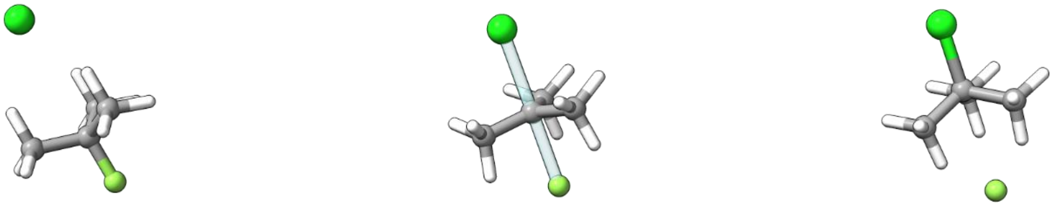
reaction 113



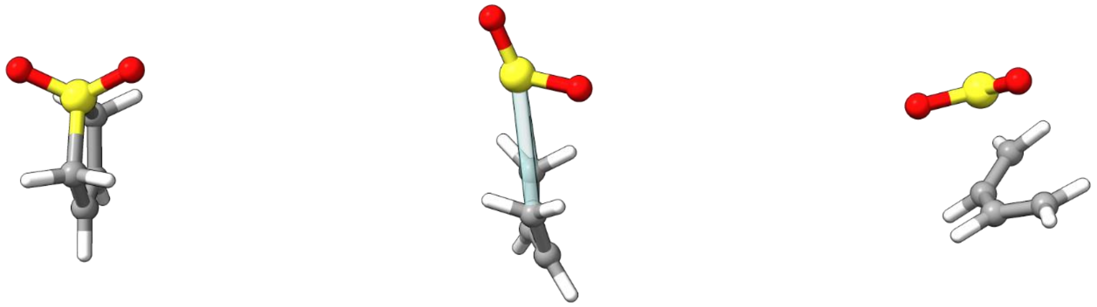
reaction 114



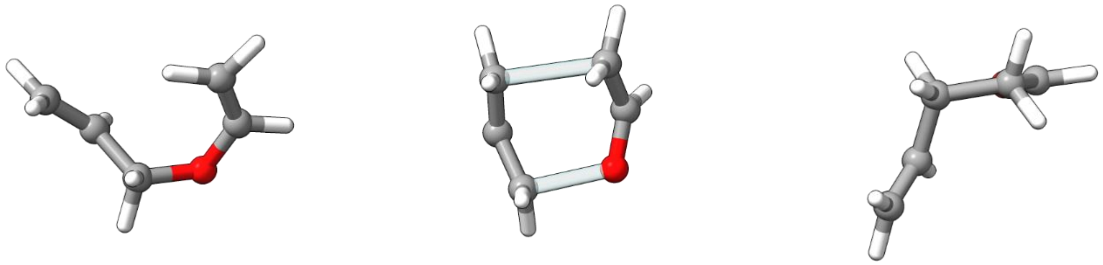
reaction 115: failed



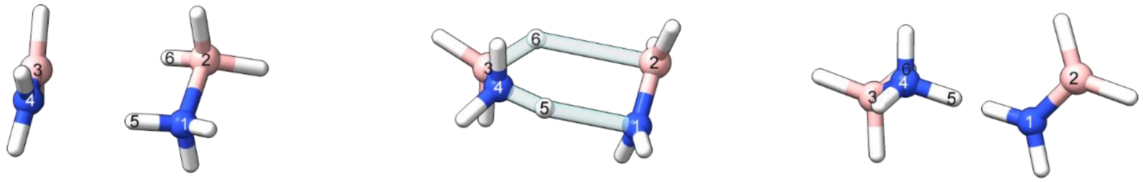
reaction 116



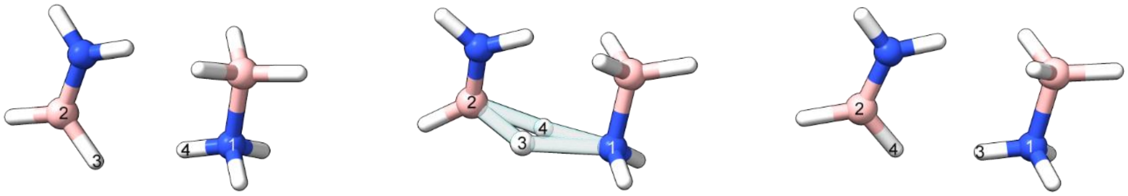
reaction 117: failed



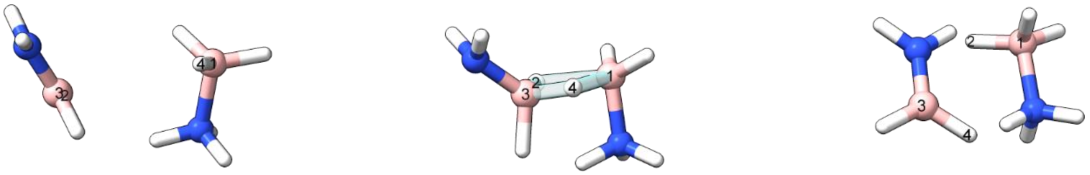
reaction 118



reaction 119: failed



reaction 120: failed



reaction 121