Exploring a Computationally Tractable Approach for Flexible Machine Learning in Organic Chemistry: Case Study in Nickel-Catalyzed Cross-Coupling

by

Jackson Warner Burns

in collaboration with

Himaghna Bhattacharjee*, Robert Martin°, Katerina Korch†, Dionisios Vlachos*, Osvaldo Gutierrez‡, and Donald Watson†

**University of Delaware, Department of Chemical Engineering*

†University of Delaware, Department of Chemistry and Biochemistry

°University of Maryland, Department of Chemistry and Biochemistry

‡Texas A&M University, Department of Chemistry and Biochemistry

A thesis submitted to the Faculty of the University of Delaware in partial fulfillment of the requirements for the degree of Honors Bachelor of Chemical Engineering with Distinction

2022

EXPLORING A COMPUTATIONALLY TRACTABLE APPROACH FOR FLEXIBLE MACHINE LEARNING IN ORGANIC CHEMISTRY: CASE STUDY IN NICKEL-CATALYZED CROSS-COUPLING

by

Jackson Warner Burns

Approved: ______________________________
Donald A. Watson, Ph.D.
Professor in charge of thesis on behalf of the Advisory Committee

Approved: ______________________________
Dionisios G. Vlachos, Ph.D.
Committee member from the Department of Chemical Engineering

Approved: ______________________________
Susan E. Groh, Ph.D.
Committee member from the Board of Senior Thesis Readers

Approved: ______________________________
Michael J. Chajes, Ph.D.
Dean, Honors College

Table of Contents

Acknowledgments

First and foremost, thank you Don. Your unending support both intellectually and professionally has propelled me to a level which I never thought possible. The freedom you gave me to explore my interests and your belief in me during the course of this project were invaluable. Additional thanks are due to my graduate mentors Feiyang, Kat, and Ola. Your patience in teaching me lab techniques, re-teaching me lab techniques, and providing day-to-day guidance made all of this possible.

To all the members of the Watson 'superlab' - thank you. The lab meetings, hallway greetings, and cupcake celebrations made coming in to work something to look forward to on even the most difficult days.

To my collaborators Dion, Osvaldo, Himaghna, and Robert, thank you for your incredible patience and enormous time investment on this work. Over the three or so years that this project spanned, I was able to develop skills across each of your domains thanks to your mentorship and guidance. Your collective passion for science drove this project forward.

Additional thanks to the one member of my thesis committee who has yet to be named - Professor Groh. From honors general chemistry to a senior thesis, the lessons you taught me in critical thinking and problem solving have carried me through all four years of my undergraduate education. As I move on to graduate study, they will continue to do so.

Last but certainly not least, thank you Mom and Dad. From encouraging me to engage with math and science throughout my K-12 education and giving me the support I needed to pursue my bachelor's degree, you have been my greatest supporters.

List of Figures

Chapter 1: Introduction

In the half century since UD's own Richard Heck discovered his eponymous reaction[1-6], the field of organometallic coupling reactions grew into one of the most critical chemical transformations practiced today. Numerous scientific reviews, multiple books, and a Nobel prize demonstrate the utility of the Heck reaction in modern chemistry[7]. Many of these advancements were guided by experimental study and the expertise of chemists alone, a time-proven process of empirical iteration.

Alongside the development of organometallic chemistry, computational chemistry began to flourish. Starting with relatively simple Fortran programs written in the 1970's[8], medicinal chemists developed metrics which could be correlated to the activity of drugs introduced to the human body. These Quantitative Structure-Activity Relationship (QSAR) studies would prove remarkably useful for the creation of novel therapeutics in the pharmaceutical space[9]. So too would they prove useful as Moore's law began to reach the limitations of physics in the last decade and world class computing hardware became commercially available. With the publication of the first Neural Network used in the chemical space a resurgence of interest in Machine Learning (ML) took place, drawing on the wealth of knowledge from computational chemists and QSAR to teach computers to understand chemistry.

Only in the last few years has ML made its way into organometallic coupling. While the field had made avid use of quantum chemical simulations to model complex transition states for decades, the complexity of the structures at hand and the scale of the systems under study made ML difficult. Doyle, Sigman, and Denmark have made significant strides in applying ML[10-12] by designing novel descriptors for this type of chemistry, overcoming challenges of data sparsity with the clever use of ML algorithms other than those that started the trend.

Current strategies rely on the use of hundreds of thousands of hours of computational time, brute force being required to counter the fact that running organometallic reactions to retrieve data for an ML algorithm is often prohibitively expensive. The premise of this work is to reduce the reliance on expensive simulations by improving the design of the data which the ML algorithm uses to make predictions. This would decrease the required computational time required for study, enabling rapid prototyping and larger studies. Following the implementation of this approach, this study will then seek to experimentally verify the result in a wet lab environment.

Chapter 2: Background

Part 1: Organometallic Cross-Coupling

The formation of bonds between carbon atoms has been a primary focus of organic chemistry for more than a hundred years. These bonds make up the backbone of nearly all organic chemicals one may encounter, and everything from human metabolism to plant photosynthesis is focused on the creation or scission of carbon bonds. While nature has developed enzymes to achieve its transformations, human chemists were limited to a handful of ‘workhorse’ carbon-carbon bond forming reactions prior to the discovery of cross-coupling. Because of their capacity to form carbon-carbon bonds under milder conditions than previous tools, cross-coupling reactions are of immense importance.

Prior to Heck’s discovery, the primary organometallic method to form carbon-carbon bonds was via the use of Grignard reagents[13]. This class of chemicals is still in use today for reactions more tolerant of the harsh reagent, though it lacks the ability to react stereoselectively (preserving the 3D configuration) or in the presence of some functional groups. With his series of sole-author papers[1-5] in 1968 and a final cornerstone paper[6] in 1972, Heck laid the foundation for palladium-catalyzed cross-coupling. In the years since that discovery, his work has been expanded to include inter- and intra-molecular cross-coupling, tolerance of asymmetric substrates, enantioselective control of products, and addition of phosphine ligands to promote and regulate reactivity[14]. These advances have enabled chemists to achieve late-stage, complex functionalization on chemical substrates, making the Heck reaction the modern workhorse of organic synthesis.

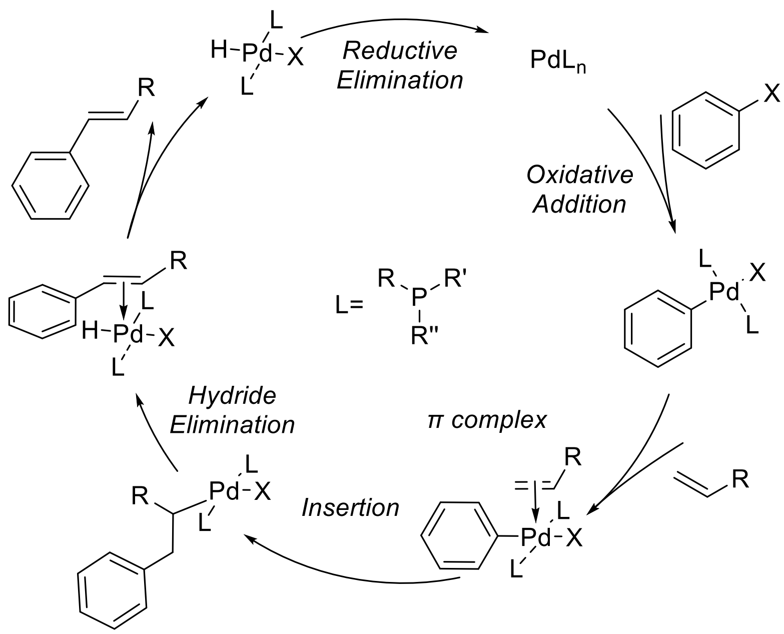


Figure 1. Catalytic cycle for Heck-type cross-coupling reactions

As a case study for the alternative approach to machine learning described previously, a model Heck reaction was identified by my collaborator Katerina Korch from current literature: Jamison's 2014 paper titled *Nickel-Catalyzed Mizoroki–Heck Reaction of Aryl Sulfonates and Chlorides with Electronically Unbiased Terminal Olefins: High Selectivity for Branched Products*[15]. In this study, the authors optimized reaction conditions and choice of phosphorus-based ligand to promote selectivity for the branched over the linear product in an electronically challenging substrate, an open problem in the field of nickel-catalyzed Heck cross-coupling since its inception.

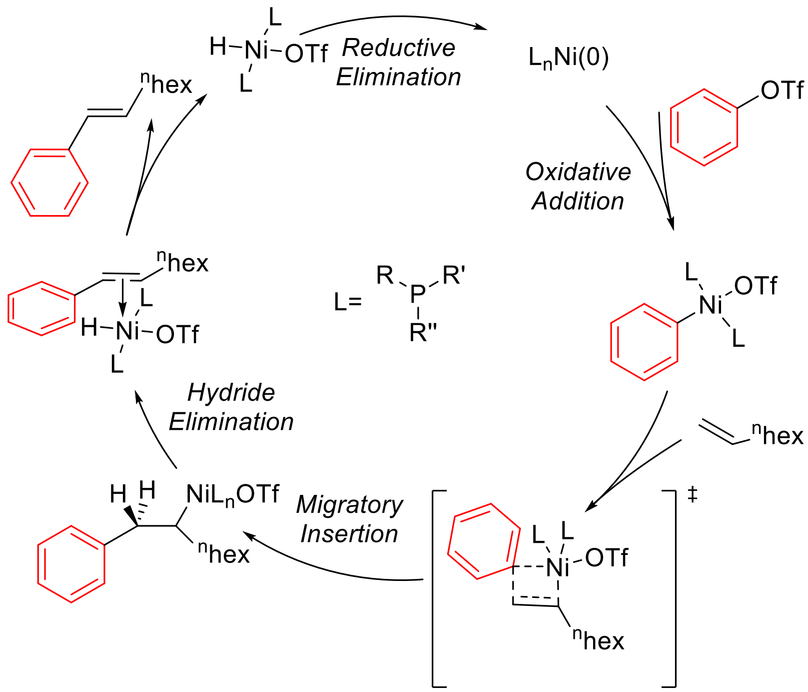


Figure 2. Catalytic cycle of model reaction

This model reaction was selected for additional practical reasons, as well. The end result of the initial study was a complete optimization for branch product over linear product, but not vice versa. Optimizing in the opposite direction would be a useful addition to the science and leaves a clear gap for investigation. Additionally, the outcome of the reaction could be represented as a single scalar value — the ratio of the two possible products — critical for presenting the data to an ML algorithm.

Part 2: Computational Chemistry

In the world of small molecules, chemical simulations are useful to visualize the geometry of transition states, probe mechanisms for optimization, and predict properties of real or hypothetical compounds. Two techniques in particular dominate the space: quantum chemical simulations such as Density Functional Theory (DFT)[16], and Quantitative Structure-Activity Relationship (QSAR) studies.

DFT refers to software, such as Gaussian[17], which solve quantum mechanical equations representing molecular structures to identify what the minimum energy configuration is in space. In the study of organometallic cross-coupling reactions, it has been used since at least the 1980's to solve for the structure of otherwise unobservable intermediates[18], providing valuable mechanistic insight. Using the final structure, additional quantities like partial charges and the HOMO-LUMO gap can be calculated to describe the electronic structure of the molecule and enable further comparison.

QSAR refers generally to the correlation of molecular descriptors to chemical properties[9], most often with pharmaceutical activity. Go and coworkers[19], for example, used the sterimol parameters originally proposed by Verloop[8] in 1976 to build a multi-linear regression capable of predicting the ability of novel indoloacetamides to inhibit a human enzyme.

$$pIC50 = f(cLogP, HOMO, LUMO, \ldots)$$

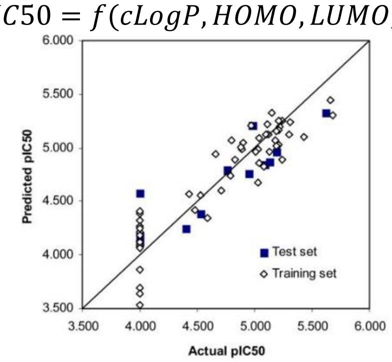


Figure 3. QSAR Parameterization Equation and Resulting Regression

Sterimol parameters, which are used later on as a descriptor for the model Heck-type cross-coupling system, consist of a set of three scalar values. Each scalar is taken from a distribution reflecting three-dimensional space occupied by a given molecule. Since the descriptor was revitalized and coded in a more modern language than the original implementation, it has been shown be massively useful in predicting the properties of compounds across a range of disciplines[20].

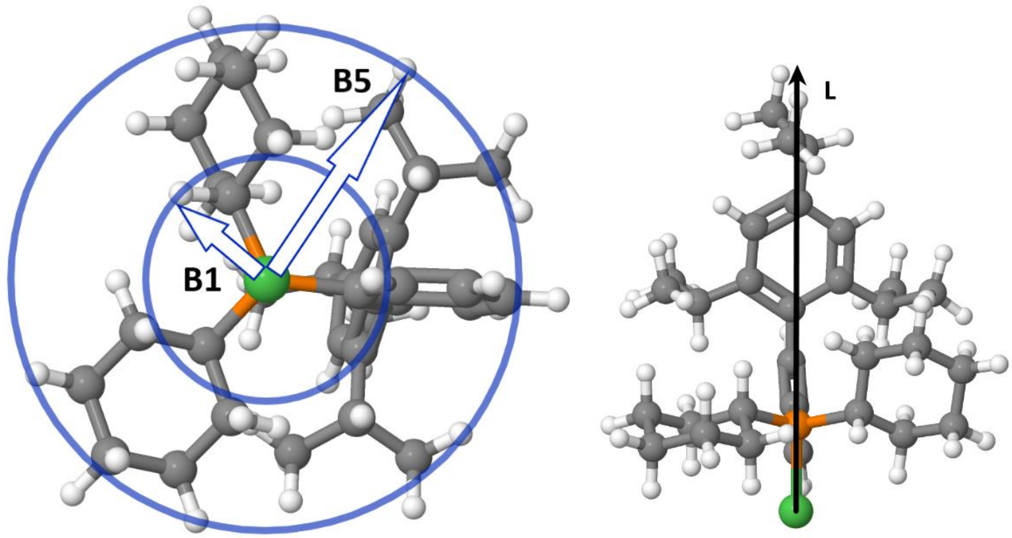


Figure 4. Sterimol parameters for phosphine complexes

In the field of organometallic cross-coupling, descriptors such as the buried volume have been developed to correlate the structural configuration of phosphine ligands to the results of the corresponding cross-coupling reactions. Luigi Cavallo's buried volume is one such parameter, and it too has been shown to predict catalyst reactivity across a range of systems[21]. Cone angle, another parameter which will be used later on, describes the angle formed by the three substituents of a phosphine[22] and has been shown to be a highly effective predictor as well[23, 24].

Another subfield of computational chemistry in the small-molecules space is in molecular mechanics (MM), often used in the study of macromolecules but with some applications to the behavior of small molecules in solution[25]. Relative to DFT simulations, MM can be used to solve systems rapidly because it uses only physical space constraints. Every atom in a molecule is treated as a hard sphere, and each ‘sphere’ in the molecule is solved with simple kinematics equations to minimize overlap and maximize distance, reaching the lowest energy structure via the “forcefield” around the atoms. MM is often coupled with DFT to identify an ensemble of different 3D configurations using the simple physics simulation and then following it up with DFT to identify a collection of minimum energy structures.

As a whole, by the early 21st century there was a wealth of computational chemistry research and software in existence to accelerate studies in chemistry. An unintended but welcome side effect of the development of DFT and QSAR techniques manifested itself alongside the re-emergence of another field: machine learning.

Part 3: Machine Learning for Small Molecules

Broadly speaking, machine learning is the act of training a computer on some input data **X** such that it can predict an outcome **y**. If you were a real estate agent, this could be the address, square footage, and color being used to predict the market value of a home. The power in ML is that, unlike a human real estate agent, your computer can handle as many homes as you can describe, limited only by the amount of memory in the computer. Additionally, the input **X** can be far more than three values - studies in cross-coupling can use more than 100 descriptors to describe a single reaction[12].

With commercial computing hardware achieving better performance and lower costs year-over-year, chemists realized that they could leverage this technology as well. Now, borrowing from the world of computational chemistry, molecules and reactions are described to the computer in terms of HOMO-LUMO gap, sterimol parameters, and buried volume. With sufficient data and correct configurations, ML can be used to predict yield, selectivity, or individual chemical properties of interest.

An additional advantage, and one whose use has exploded in recent years, is the advent of virtual screening. Accurate models can supplant experimentation in protein docking[26] and pharmaceutical development[27], saving both time and money otherwise spent at a bench. With a well-trained model in hand, a chemist need only generate some descriptors for a hypothetical chemical compound - a quick task with modern computers - and query the model. At scale, entire libraries of 'virtual' molecules, in this case phosphine ligands, can be created and evaluated instantly.

Implementation of these broad ideas is the final barrier to making ML-guided chemistry a more ubiquitous tool. As is stands, retrieving molecular descriptors requires writing original code to use multiple of the many fragmented descriptor-generation tools available. Additional challenges are faced in the process of implementing the ML models themselves. In the case of cross-coupling reactions, each new data point can represent many hundreds of dollars invested in reagents and time. Data-greedy ML algorithms are forced to operate in a definitionally data-sparse environment, requiring extensive fine-tuning on the part of the programmer.

Chapter 3: Experimental Investigation

Part 1: Reaction Identification

As discussed briefly above, a case study for the new ML approach was identified based on a number of criteria. The first was scientific merit, in that the study was facing a challenging and still open problem within chemistry. The model Heck reaction[15], identified by my collaborator Katerina Korch, required optimization of reaction conditions and choice of phosphorus-based ligand to promote selectivity for the branched over the linear product in an electronically challenging substrate, a persistent problem in the field of nickel-catalyzed Heck cross-coupling since its inception.

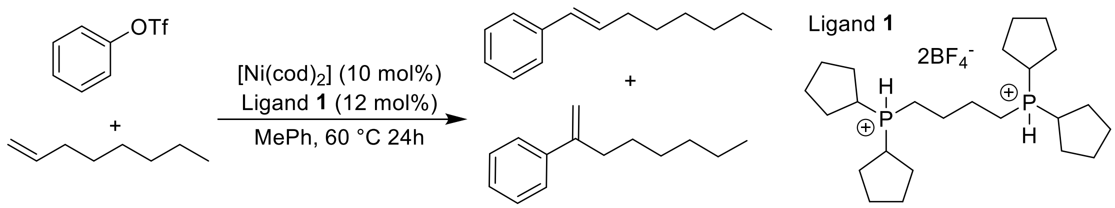


Figure 5. Model reaction system

Historically, it has been difficult to achieve any selectivity in this class of reactions when the substrate (*Figure 5*) is unfunctionalized. This is because the lack of standout electronic features shifts the burden of modulating reactivity exclusively on to the transition metal complex.

The authors achieved near-perfect selectivity for the branched product over linear product, but not vice versa. Both of these products would be of immense use as intermediates in product synthesis, and no other existing solutions can access the linear product exclusively via this reaction pathway. An additional strength of this reaction is the ease with which the outcome can be presented to a computer. The ratio of the yields of the two products reflects the ‘preference’ between the two outcomes on a continuous scale, accurately reflecting the behavior of the system.

Some final technical qualifiers for the reaction to be studied were also passed. The technology with which the Watson lab studies reaction optimization is High Throughput Experimentation (HTE), in which reactions are scaled down to a few millimoles of material and a single control variable is examined in hundreds of different "mini-reactions" at the same time. The control variable in this experiment is the phosphine ligand, which can easily be varied across reactions at this scale. Finally, the results of the reaction can be quantified without need for a full workup and isolation - that is, in a high throughput fashion.

Part 1.1: Development of Crow

To that final point, in the course of this project the challenge of analyzing large volumes of data generated via HTE was a recurring theme. This led to the spawning of a secondary aim of the overall endeavor: to design a software tool to enable HTE data analysis on a large scale. The UD HTE facility is equipped with a gas chromatogram configured to accept multi-well HTE plates; this relatively simple hardware configuration is sufficient to perform HTE, but the corresponding software is only sold packaged with custom-built HTE equipment. This carries a heavy price tag, removing it from the realm of possibility for many academic institutions, and prevents carrying forward existing equipment, which is needlessly wasteful.

Over time this concept evolved into a tool now known as *Crow* (*Crow Really Outta' Work*), a free and open-source Python package that can readily retrieve, parse, and present data from nearly any instrument capable of HTE[28] without the expense and limitations of existing commercial solutions. I initially wrote the program in MATLAB and then coded into Python with the financial support of the University of Delaware Undergraduate Research Program. *Crow* provides three key functionalities: Pre-Pull, Pull, and Present. Pre-Pull is a diagnostic tool to identify which products are present in each well of an HTE screen, simplifying data retrieval. Pull then provides the functionality to do said retrieval. Users specify an expected retention time for a given products, and *Crow* programmatically parses data files from the chromatograms, returning tabulated peak areas, masses, or whatever arbitrary value the instrument produces. Using these values, users then turn to Present, which transforms tabulated yield or selectivity data into graphics for easy interpretation, such as the example shown in *Figure 6*.

Crow is now in use at the University of Delaware High Throughput Experimentation Facility and has accrued hundreds of downloads online. Owing to its implementation in Python, leveraging other open-source tools, it is compatible with all operating systems and customizable for any hardware configuration. The base tool itself is also flexible and easily extended, so I hope that it will continue to evolve over the coming years.

Part 2: Data Retrieval via High Through Experimentation

To begin studying the reaction at hand, more data was needed to evaluate performance across a range of ligands. This step in the project was completed primarily by Katerina Korch, a collaborator with extensive expertise in the world of chemistry and High Throughput Experimentation, and Austin Kelly, previous director of the UD HTE Facility. Leveraging the ligand library at the facility, a screening on dozens of possible phosphine ligands was performed with many replicates to ensure data consistency. Each replicate took place in a 96-well HTE plate, such as the one pictured in *Figure 6*, to generate data as shown in the included example graphic.

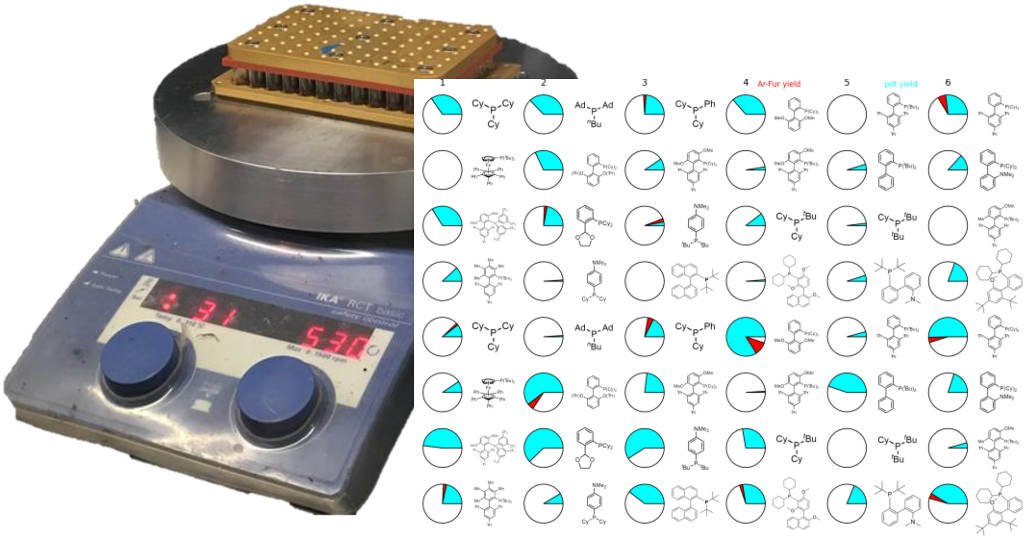


Figure 6. HTE plate used in experimental studies with simulated run result

Ligands selected for evaluation were hand-picked to cover the broadest range of chemical space possible, including tri-alkyl, di-alkyl aryl, alkyl di-aryl, and tri-aryl phosphines. A variety of functional groups were also included, as well as a number of common phosphine ligand motifs such as monodentate and bidentate ligands, as well as the ubiquitous Buchwald-type[29]. Selected results of the initial screening are shown in the graphic below, generated using *Crow*. A complete list of ligands evaluated and a table of all HTE results is included in the supporting information.

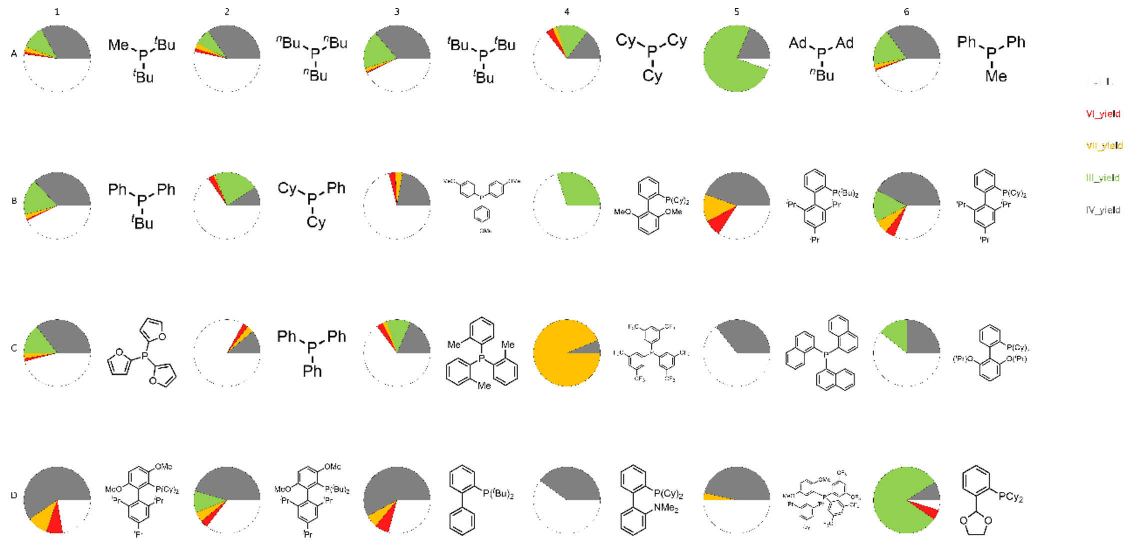


Figure 7. Select HTE screening results

Each pie chart in *Figure 7* represents one individual well in the reaction plate, in each of which is contained a unique phosphine ligand pictured next to the pie chart. The 5 colors present indicate the yields of each of the products for the given ligand. A critical piece of information gained from this study, beyond just the data itself, was the sheer variety of different behaviors witnessed. Nearly every possible combination of the 5 products is seen, and in wildly inconsistent ratios. For example C4, tris(3,5-perfluoromethylphenyl)phosphine, produced almost exclusively product **VII**, whereas A5, diadamantylbutylphosphine, produced mainly compound **III**. Neither of these behaviors had been observed in the initial study, and unfortunately for much of the machine learning modeling phase of the project only the two anticipated major products were quantified. This will be discussed in more detail later on.

The distribution of the data, however, is both promising and potentially hazardous. Some ligands included in the HTE screen demonstrated near perfect selectivity for the same isomer the original study had optimized for, and a handful reached excellent selectivity in the opposite direction. Nearly all of the data points sat somewhere in the middle, not deviating from more than 2:1 selectivity in either direction. When this data is fed to a machine learning algorithm, it is more than likely that the training data may not contain any previous examples of highly selective ligands. This would force the model to attempt to extrapolate, a dubious task.

Not pictured in this result, but included in the Appendix, is the overall yield of these reactions, which was often in the single-digit percentages. This would prove to be a nuisance in subsequent analysis, as these ligands could technically produce extremely high selectivity results (huge preference for one product over the other) where in reality they yielded a minute amount of one product and a nearly undetectable amount of another. A significant number of results, however, are promising. Many of the ligands in this initial screen showed high selectivity for the linear (orange) product over the branched (red) product, even if in low yields.

Chapter 4: Implementing Machine Learning

Part 1: Parameterization

At this point in the process, a set of approximately 100 diverse ligands have been run in the model reaction, and the resulting list **y -** the selectivity values **-** is in hand. Now, each ligand must be parameterized as a list of descriptors to form the **X** descriptor matrix.

Collaborators Robert Martin and Dr. Gutierrez, experienced computational chemists, identified a list of parameters reflecting the electronic behavior of the molecules. The Watson lab also selected a set of steric parameters, which reflect the 3-dimensional configuration (conformation) of the molecule in space. Decision making on both sides was guided by *a priori* knowledge of the reaction system, as well as literature precedent from similar studies. See the table of molecular descriptors for a complete list.

Name	Description[reference]
Phosphorus-Carbon Bond Length	Distance between central phosphorus and each of the three nearest carbons
Carbon-Phosphorus-Carbon Bond Angle	Angle formed by central phosphorus and 2 of the 3 nearest carbons
Cone Angle	Angle formed by the central phosphorus and all 3 nearest carbons as if shaped in a cone[30, 31]
Buried Volume	For a sphere drawn around a point x distance from the central phosphorus, the % of that sphere occupied by the ligand[21]
Sterimol Parameters	Scalars representing a distribution of molecular conformations[8, 20]
Phosphorus Atom Mulliken Charge	Population-adjusted partial charges on individual atoms resulting from asymmetric electron distribution[32]
Phosphorus Atom APT Charge	Dipole-adjusted Atomic Polar Tensor charge[33]
PNMR Shielding Constant	Predicted NMR frequency of phosphorus atom[34]
CNMR Shielding Constant	Predicted NMR frequency of carbons atoms closest to the central phosphorus[35]
Phosphorus-Carbon Bond Stretch Frequency	Predicted IR Spectral Vibrations of Phosphorus-Carbon Bond[36]
HOMO Eigenvalue	Mathematical reduction of matrix representing highest occupied molecular orbital of the molcule[37]
LUMO Eigenvalue	Lowest Unoccupied Molecular Orbital alternative of HOMO eigenvalue[37]
HOMO-LUMO Gap	Energy difference between HOMO and LUMO[38]

Figure 8. Table of molecular descriptors

The exact methods by which these values are calculated are covered in a subsequent section. Before doing so, the literature precedent must be explored to frame the alternative approach used in this case study.

Part 2: Computationally Inexpensive Approach

All of the studies performed by contemporaries in this field rely exclusively on DFT to generate the parameters shown in the molecular descriptors table[10-12]. One example of note is *kraken*, the name given to the collection of DFT-derived molecular descriptors calculated by Aspuru-Guzik and coworkers[39]. This collection includes hundreds of existing phosphines used in transition-metal catalyzed chemistry as well as many thousands of ‘virtual’ phosphines the authors theorize might be useful. The results are also available online, meaning that researchers who can find their ligands in the dataset can access and use the descriptors to train their own machine learning models. While impressive, this work required the combined resources of seven institutions, both private and public, to generate the data. We estimate that the total computational cost of this study was in excess of half a million computational hours. This makes both reproducibility and translational application of the principles practically impossible.

The solution explored in this case study moves away from this excessive use of DFT and attempts to integrate lower-order molecular simulations to reduce total computational cost. As demonstrated in the diagram below, each phosphine evaluated in the initial HTE screen is subdivided into three components. Each of these is composed of the central phosphorus atom and one of the three ‘substituents’ directly affixed to it with the remaining two groups substituted for hydrogen atoms.

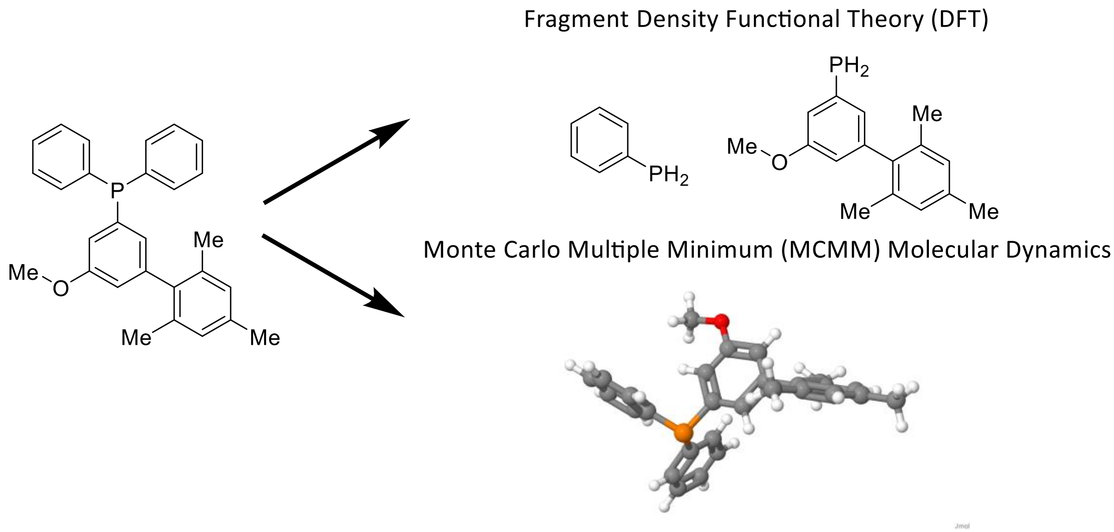


Figure 9. Proposed alternative descriptor generation technique

The electronic parameters required to describe this system can only be retrieved using quantum chemical simulations such as DFT, but each complete ligand can take multiple weeks of computation to solve. These ‘fragments’ can be simulated via DFT in a matter of hours owing to the dramatically reduced scale of the system. This relies on the assumption that said parameters are independent of other ‘fragments’ attached to the central phosphorus, built on the limited spin-spin coupling observed across the phosphorus atom. Monte-Carlo Multi-Minimum (MCMM) simulations, an extension of the molecular mechanics methods mentioned above, can then be used to solve for an ensemble of low-energy conformations of a given molecule[40]. This approach uses a simple molecular force field to guide rotation of bonds and orientation of atoms to reduce the total energy of a molecule. The resulting structures are used to calculate the remaining steric parameters. In summary, the main thrust of the hypothesis is two-fold: it is assumed that the steric parameters describing phosphines are dominated by simple physics constraints, and that the DFT parameters are independent of position relative to other species in the phosphine.

Generation and retrieval of the electronic parameters was performed manually by our collaborators at the University of Maryland. Geometry optimizations were performed at B3LYP-D3/def2SVP-cpcm(dichloroethane) level of theory. Owing to the large number of structures generated by the MCMM simulations used for the steric descriptors, retrieval of said values was automated using shell scripting and Python[41]. The scripts used in this study are included in the supporting information. In similar fashion to *Crow*, this phase of the project also sparked the creation of a software tool.

Part 2.1: Development of py2sambvca

Buried volume[21] is a chemical descriptor of immense importance, but *Sambvca* — the tool used to calculate it — is inaccessible to Python programmers. It can only be accessed manually via a web browser or by compiling and executing Fortran code from the command line. For ML applications it is critical to generate buried volume values dynamically and without human intervention, especially on datasets of this size where the human work input would take many hours.

py2sambvca[42] is an easy-to-use, thin client Python interface to the original *Sambvca* program. It provides access to all the key functionalities of the original Fortran functions using Pythonic programming conventions. For example, users can simply call the function 'get_buried_volume' to retrieve the corresponding value from *Sambvca* without having to deal with the different data types in the two programs or the tedious process of building the input file that *Sambvca* requires. Since its release, *py2sambvca* has been used in multiple settings across chemistry and attracted code contributions from the cheminformatics community.

Part 3: Modeling Results

With the assistance of our collaborator Himaghna Bhattacharjee, a multitude of ML models were implemented. All programs were written in Python[41] using the sklearn package[43].

Out of the many thousands of ML models recorded in literature, Varnek and Baskin have identified a subset of known use in cheminformatics-type studies[44] and there exist many individual examples specific to this discipline (referenced by name). During the course of the project, the models evaluated included neural networks[45, 46], random forests[12], Bayesian regression[47], and kernel-based methods. Each model is evaluated based on its mean absolute error (MAE) and its prediction error plot (PEP). MAE is calculated using the formula shown in equation 1, where *y* is a predicted value, *x* is a true value, and *n* is the total number of values in the set.

$$\boldsymbol{Equation\ 1}.\ MAE = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

Smaller values indicate 'better' performance, meaning that on average a model with a lower MAE predicts with less error than a model with a higher MAE. The PEP is essentially a visualization of the MAE, showing precisely how the model performed on the input data. It can also reveal aspects of model performance which would otherwise be hidden by the scalar MAE value. For example, the model behind *Figure 10* would have a low MAE because of the tight clustering of points around 0.5-0.6 on the ground truth axis, which would initially seem to be a promising result.

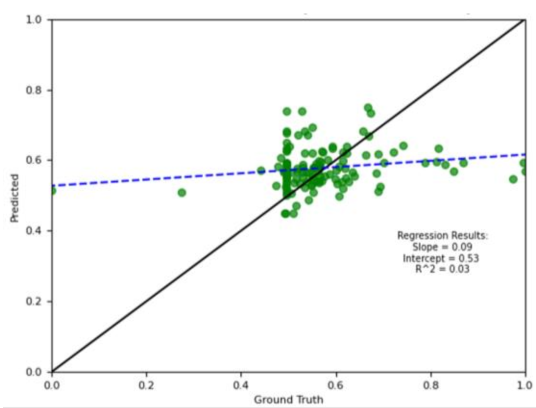


Figure 10. Example Prediction Error Plot (PEP)

Upon inspection of the PEP, however, it is obvious that the model is predicting only values in this range. Data points that lie at the extremes of the horizontal axis are predicted entirely incorrectly by the model. The linear regression line, too, strays far from the central black line, indicating the model is highly inaccurate.

Neural networks were among the first category of machine learning algorithms tested on this dataset, primarily for their wide acclaim in the world of ML. The approach works by mimicking the configuration of biological neural systems, being made up of interconnected neurons. The input to the neural network, in this case a list of chemical descriptors, is broken up and fed to an input layer of neurons. Each of these then passes the input to one or more neurons in a second layer, which then perform some mathematical calculation on their input(s). This process is repeated with subsequent layers of different sizes and levels of interconnectedness until the final output layer is reached, which represents the predict result for the input. The training aspect of neural networks refers to the specific functions used in the middle-layer neurons, each of which has configurable parameters similar to a weight in a linear regression technique.

Unfortunately, neural networks of all varieties were unsuccessful on this dataset, never learning any of the trends. *Figure 11* is a single result for a network configuration that was representative of the typical behavior.

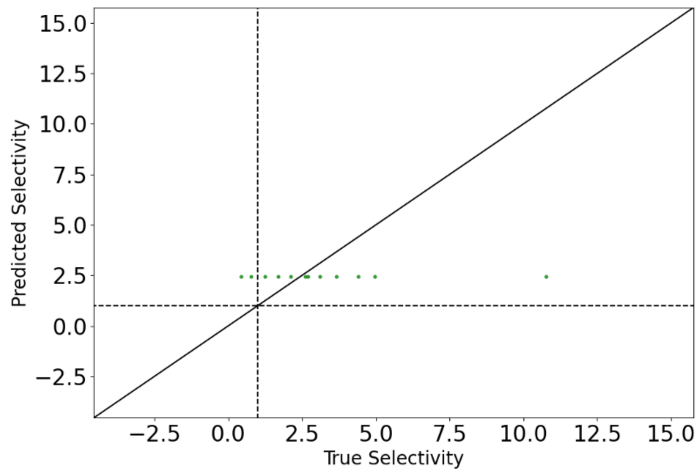


Figure 11. Neural Network PEP

As shown, the network consistently predicted the exact same selectivity value of 2.5 regardless of the input. MAE is a meaningless metric in this circumstance, since it is obvious from the PEP that the network is only guessing every time it is queried.

This failure is likely due to the size of the dataset and how it interacts with the architecture of the network. The aforementioned values of the neuron parameters must be optimized using backpropagation. This technique involves repeat analysis of how the weights of the nodes correspond to the error in the output, and without sufficient samples the network is unable to optimize to a sufficient degree. Given that extending the dataset at hand is prohibitively expensive, this approach was abandoned in favor of the others.

Bayesian regression and kernel-based methods suffered similar issues to the neural networks in terms of the dataset size preventing sufficient optimization. It was also at this point in the study that the outcome of the reaction was re-evaluated to ensure the data was not somehow interfering with the modeling efforts. As mentioned previously, initial modeling attempts assumed that only two products were forming, but in reality (and as demonstrated in

Figure 7) this was not the case. The original paper reported only two primary compounds in the reaction products and wrote off other reactivity as statistically insignificant; the addition of new ligands to the dataset had made alternative mechanistic pathways possible and led to a slew of byproducts being formed. The mechanistic pathways by which these structures can be reached are discussed in *Figure 12*.

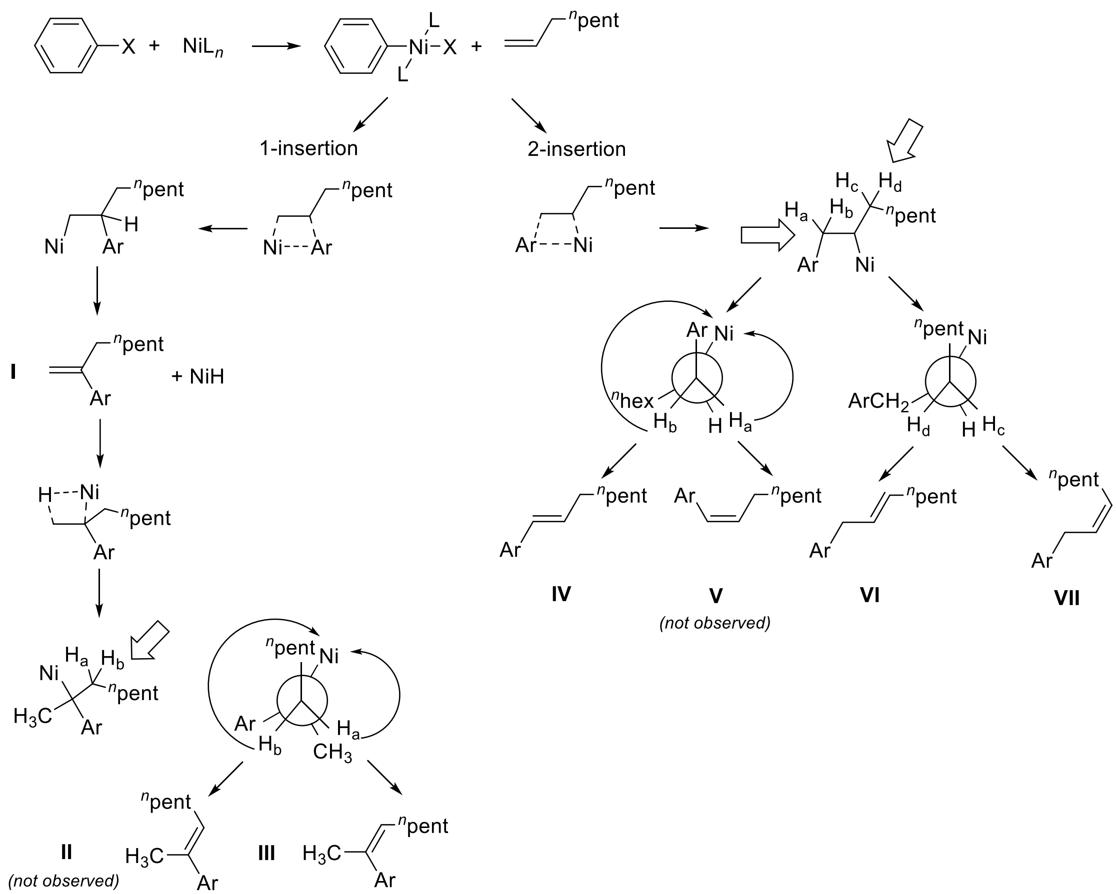


Figure 12. Novel byproducts identified in model reaction

The first step in the reaction where differentiation of products occurs is at the nickel insertion, which takes place at either the 1 or 2 position on the substrate's terminal alkene. Product **I** is the result following 1-insertion through the remainder of the catalytic cycle and stopping at the product, but product **III** can be formed from product **I** if the resulting nickel hydride species is not deprotonated sufficiently fast. If not, it isomerizes the product to a more thermodynamically stable internal alkene.

If instead the nickel inserts at the 2 position, the remaining 4 possible products come into play. After migratory insertion takes place, there are 4 possible hydrogens that can be eliminated ($H_{a\text{-}d}$, top right) to give products. The Newman projections shown in *Figure 12* demonstrate that exactly which hydrogen is eliminated is controlled by the steric interactions of the substrate and the aryl cross-coupling partner, which are further complicated by the presence of the ligated nickel species. Only product **V** was not observed, since the steric demands of the beta-hydride elimination are too energetically unfavorable.

In summary, products **I** and **IV** were expected given the previous literature report, but in reality they are accompanied by **III**, **VI**, and **VII**. Further complicating analysis was the fact that every possible combination of these 5 products was also observed in the dataset. The binary branched:linear selectivity ratio being used prior to this point was, in reality, missing more than half of the reaction products in some cases. Thus, for subsequent modeling, a new **y** metric was devised: the sum of yields of **I** and **III** (the products of initial 1-insertion) minus the sum of the yields of products **IV**, **V**, and **VI**. This new approach, hereafter abbreviated as BMI, maintains the key criteria for our ML algorithms, in that it presents the result as a single scalar value that can be regressed. It also is carefully devised so that it 'asks only a single question' of the data. This new ratio is decided by a single step in the mechanism (insertion), so in essence the algorithm is only being asked to discern to what degree the ligand promotes insertion in a certain orientation. By ignoring these additional products, we had previously been asking the algorithm to predict not only a ligand that favored 1 or 2 insertion, but one that would prevent subsequent isomerization and preferentially beta hydride eliminate.

Drawing again from contemporaries in the field[12], random forest regression[48] was attempted with this new target in hand. Random forests are a unique, ensemble approach to machine learning wherein a collection of binary decision trees are assembled in large numbers to predict outcomes for a given input sample. Initial attempts showed some promise, averaging MAE values of 0.l5. This was not quite at the threshold of ‘confidence’ to motivate moving forward into model validation, a costly process. Further additions to the initial dataset were needed, which led to the development of a third and final software tool adjacent to this primary project: *AIMSim*[49].

Part 3.1: Development of AIMSim

Physical investigation of chemical reactivity has always been expensive, both in terms of time investment and monetary cost. This push for implementing ML speaks to the desire to avoid that cost, but doing so necessitates some initial experimental study. In order to minimize resource waste, it pays to know *a priori* whether or not an experiment will add worthwhile data to an existing dataset. Worthwhile could mean that the new data explores an untouched area of chemical space, or that it minimizes or maximizes some desired molecular descriptor. To help answer these questions, Himaghna and I developed *AIMSim*.

AIMSim uses molecular fingerprints — binary lists representing functional groups in a molecule — to generate similarity matrices for a given database of molecules. Potential new additions can be checked against this matrix, allowing the user to optimize for new data points which are highly dissimilar to existing examples, exploring more chemical space. For this project, *AIMSim* was able to identify an additional set of twenty ligands which were structurally unique within the existing data and would enhance the coverage of possible phosphine ligands. Over time, though, *AIMSim* grew to include molecular fingerprints, similarity metrics, and comparison methods applicable to a broad range of fields.

Just like *Crow* and *py2sambvca*, *AIMSim* is free and open-source, available for anyone to download, extend, and use in their daily workflows. It too is implemented in Python and is explicitly focused on facile addition of new features; we hope to see the cheminformatics community engage with the tool. *AIMSim* could easily be extended to have applications in materials science or general dataset preparation, the latter currently being attempted by me and Himaghna.

Part 4: Modeling with Enhanced Data

With this enhanced dataset modeling results improved markedly. Again using random forest regression, MAE dropped to 0.07, corresponding to an average error in the predicted product yield of less than 10%. This is on par with similar method applied to this field and nearing the limits of what experimentalists would consider experimental accuracy for bench-scale organic synthesis. Shown in *Figure 13* is the PEP for the best performing model.

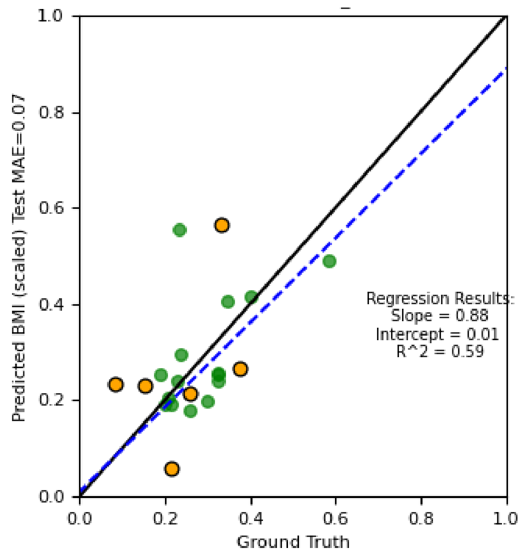


Figure 13. Random forest regression results on enhance dataset

The original dataset is pictured in green and the additional data from the revisiting step is shown in orange. Despite a handful of outliers, the model overall performs well on the dataset. The slope of the prediction line is near unity and the correlation coefficient, though far from unity, is higher than all other models trained previously.

With this more accurate model in hand, it is now possible to move on to the experimental validation of the computational model. To do this, the model will be queried with a number of hypothetical phosphine ligands that do not yet exist, and one will be selected and synthesized to verify the model's prediction. Doing so ensures that the model is not just predictive on paper, and that in the actual physical space of the real world it is able to understand the chemistry.

Chapter 5: Verifying Model Results

Part 1: Identification of a Novel Ligand

Identifying a new ligand to synthesize is made far simpler by this alternative modeling approach, as it allows for the rapid parameterization of tens of thousands of possible phosphines. Using a combinatorial approach, all of the individual fragments used to assemble the ligands in the dataset were recombined to form novel ligands. Electronic parameters were already known, and steric parameters were easily retrieved owing to the short execution time of the MCMM simulations and the scripts previously prepared.

The model identified a number of potential top performers (high **y** value), which are shown in *Figure 14*, and of that set one ligand was selected for experimental investigation.

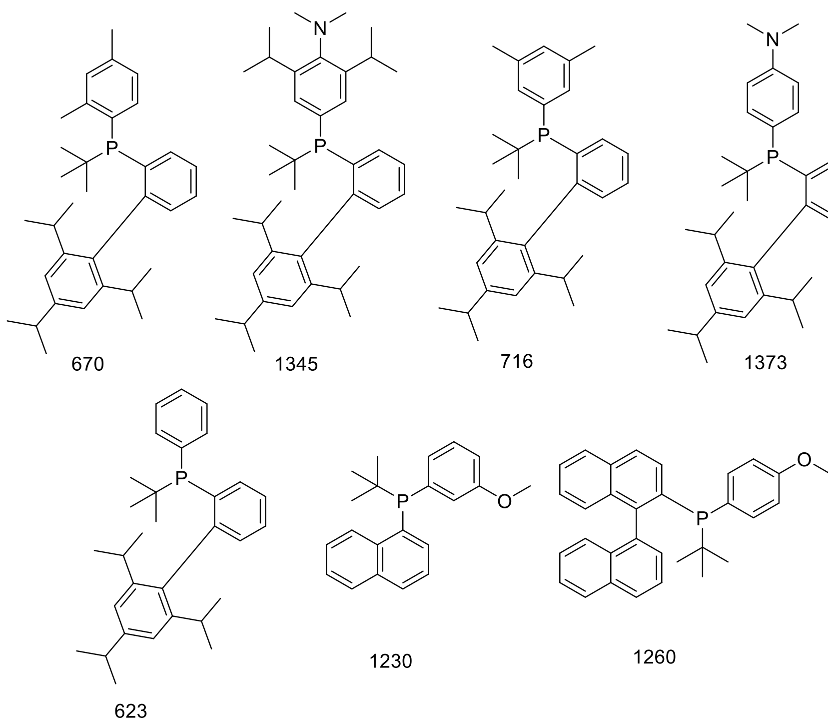


Figure 14. Virtual ligands identified by trained model

Of note is that every ligand in this dataset contains the tert-butyl fragment, and many contain the 2',4',6'-triisopropyl biaryl backbone typical of the Buchwald family of ligands[29]. Of this set, ligand 716 was chosen for further study.

716

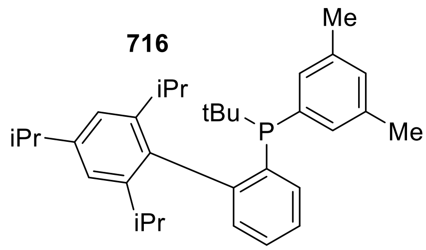


Figure 15. Novel ligand identified for study

Ligand 716 was selected for its perceived synthetic accessibility as well as the assumed ease of synthesis. The dimethylphenyl fragment is easily accessible via commercially available 1-bromo-3,5-dimethylbenzene[50], tert-butyldichlorosphosphine is a ubiquitous starting material for this class of ligand[29, 51], and the remaining biaryl fragment is similar to known literature examples and easily synthesized[52].

Part 2: Attempted Synthesis of Novel Ligand

Given the literature precedent discussed above, the synthetic route shown in *Figure* 15 was proposed to access the Ligand 716 ligand. At optimal throughput the synthesis could be complete within the span of three working days, though this ideal circumstance never came near fruition.

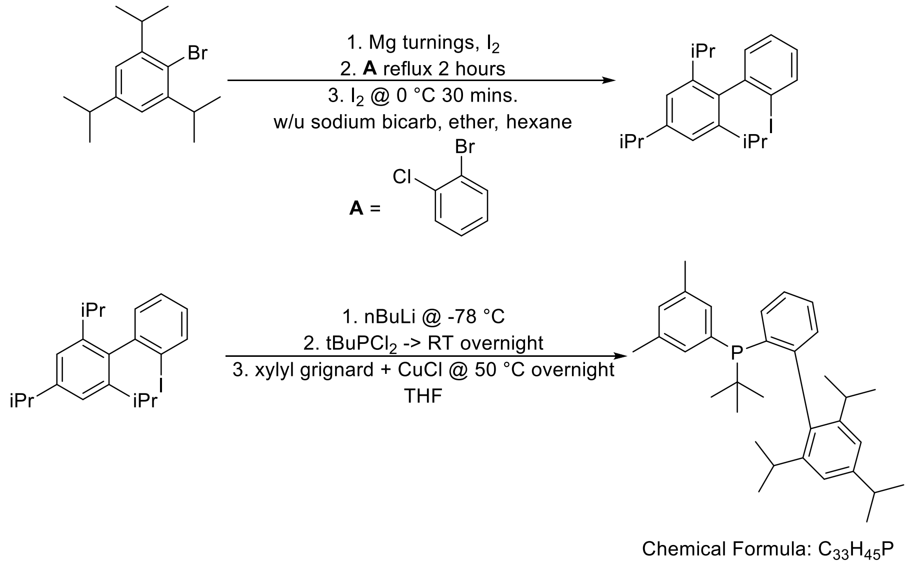


Figure 16. Proposed synthetic route to Ligand 716

The first phase of this synthesis was straightforward and prdocued the biaryl fragment readily and in good yields via a known route for a similar compound. The challenges began to arise in the final step of the second phase of the reaction. Attachment of the biaryl fragment to the tert-butyl chlorophosphine proceeds easily through a nucleophilic substitution reaction. Subsequent attachment of the final phenyl fragment proved to be challenging. A variety of methods were attempted, none of which were successful. Simple lithium-halogen exchange via organolithium was unsuccessful at reaction times between 24 and 96 hours. The addition of copper chloride to form organocuprate species[53], which is intended to replace the lithium *in situ* and better align the molecular orbital energies of the chlorophosphine and aryl fragment, could not generate the product. Refluxing the reaction mixture for days on end to drive the reaction forward was also unsuccessful, with Ligand 716 never observed by NMR prior to or after isolation. The reaction would only ever reach the product shown in *Figure 17*, the result of the tert-butylbiarylchlorophosphine being exposed to air and water in the course of the workup.

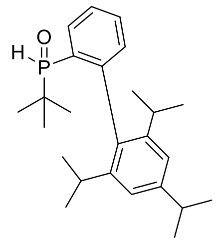


Figure 17. Result of Ligand 716 Synthesis

It was thus determined that this highly sterically encumbered ligand was kinetically inaccessible to synthesize with the methods being used. An additional complicating factor may stem from the fact that the underlying ML model has no physics constraints and is agnostic from the mechanism of the reaction. The consequence is that it cannot predict possible candidates that are actually feasible to make. This possibility discourages simply returning to the list of predicted high performing candidates, as it is just as likely that they cannot be synthesized. It may be true that the ligands would be excellent at promoting reactivity in exactly the way requested of the model, but they exist outside of accessible chemical space so it becomes a moot point. I did not have time to attempt the synthesis of the other lead ligands outlined above, nor attempt an alternative synthetic pathway.

Ultimately, given the limitations of time and the imposing structures of many of the other ligands outlined above, the decision was made not to pursue further experimental validation of our approach. In addition, after the release of *kraken*[39], discussed above, we also reflected on the overall value of this approach to the community. The goal of this project was to circumvent the expense of computing parameters using DFT, which *kraken* does for the entire chemistry community since its results are publicly available. The method under construction here was originally a workaround for the problem of massive studies requiring unreasonable computational time, but the *kraken* study took that issue head on in a brute-force way by combining the resources of multiple institutions and then sharing the results. While other points about rapid prototyping and translational applications still stand, the release of this tool hampers enthusiasm from the community about the utility of the approach.

Chapter 6: Conclusion

This case study set out to investigate whether typical machine learning approaches to the optimization of Heck-type cross-coupling reactions could be supplanted by an alternative, hybrid approach that leveraged computationally inexpensive simulations to reduce overall expense. Throughout the course of the work, challenges related to data scarcity and understanding of the background chemistry were faced. Ultimately, the model created using the alternative requested impractical chemicals be synthesized in order to optimize the reaction performance. Time constraints hampered further efforts to attempt alternative syntheses or correct this deficit. The final blow was dealt with the release of data retrieved using the typical strategy, effectively eliminating the need for simulation altogether for subsequent researchers by having shared it online for free. Fortunately, the development of secondary software alongside the primary aims of the project yielded multiple tools that are now actively in use by the community at large. *Crow* is used at the UD HTE facility to enable rapid retrieval and presentation of results from studies. *py2sambvca* has been applied to diverse fields and attracted contributions from the cheminformatics community. *AIMSim* helps solve an open problem in the world of experimental investigation and is free, open-source, and ready for others to explore.

SUPPORTING INFORMATION

for

EXPLORING A COMPUTATIONALLY TRACTABLE APPROACH FOR FLEXIBLE MACHINE LEARNING IN ORGANIC CHEMISTRY: CASE STUDY IN NICKEL-CATALYZED CROSS-COUPLING

by

Jackson Warner Burns

in collaboration with

Himaghna Bhattacharjee*, Robert Martin‡, Katerina Korch†, Dionisios Vlachos*, Osvaldo Gutierrez‡, and Donald Watson†

**University of Delaware, Department of Chemical Engineering*

†University of Delaware, Department of Chemistry and Biochemistry

‡University of Maryland, Department of Chemistry and Biochemistry

A thesis submitted to the Faculty of the University of Delaware in partial fulfillment of the requirements for the degree of Honors Bachelor of Chemical Engineering with Distinction

2022

Appendix 1: Experimental Methods

Cross Coupling Reactions

All cross coupling reactions completed as part of this study were done according to the procedure set forth in the initial study on which this study is based[15]

Synthesis of Ligand 716

Attempts at the synthesis of Ligand 716 were completed according to the procedure that follows.

To an oven-dry 3-neck flask with condenser under inert atmosphere add magnesium turnings (2.5 eq) and wet with THF (final conc. 3M). Add a small crystal of iodine and allow to stir briefly until color lightens. Gradually inject 1,3,5-triisopropyl-2-bromobenzene (1 eq) and the remaining THF. After all material is injected, reflux at 80 Celsius for 2-4 hours. Inject ortho-bromochlorobenzene slowly, continue reflux for 2-4 hours. Cool mixture to 0 Celsius, add iodine (1 eq) and allow to warm to RT over 30 minutes. Quench remaining iodine and Grignard with sodium sulfite, extract with diethyl ether 2-4 times, dry with magnesium sulfate, filter, concentrate, wash with hexane, and high vac. Should yield off-white powder, iodo-biphenyl fragment. If water remains in sample, dissolve in toluene and concentrate on rotovap 3x, high vac to remove trace toluene, and check again.

To an oven-dry schlenk flask under inert atmosphere, add the above product (1 eq., evac. & backfill 3x) in THF (0.25M). Cool to -78 degrees Celsius and inject nBuLi (2 eq) and allow to stir for >=30 minutes. Inject tertbutyl-dichlorophosphine (1 eq) and allow to warm to RT overnight. Take a PNMR of the crude reaction mixture to check for completeness. To a second oven-dry 3-neck flask with condenser under inert atmosphere flame-dry magnesium turnings (2.2 eq), coo l with N2, add I2 (evac. & backfill 3x), add THF (0.5M) to wet solids, gradually inject both the remaining THF and inject 1,3-dimethyl-5-bromobenzene (2 eq), stir for 5 mins. Reflux at 80 C until clear, ~ 1-2 hrs. Using a cannula, transfer the liquid layer from the second flask into the first flask and stir overnight.

Product is air sensitive so workup must be performed under inert atmosphere. Sparge 2 round bottom flasks of sodium bicarbonate and brine for at least 15 minutes to remove trace dissolved oxygen. Attach an oil bubbler to the main flask and SLOWLY (dropwise or slower) inject the sparged sodium bicarbonate to quench the remaining Grignard and any acid generated. Allow the mixture to stir for 10 minutes, stop stirring and allow layers to separate, and then remove the aqueous layer via a syringe. Repeat this washing step one additional time. Inject brine, allow to stir for ten minutes, stop stirring, allow layers to form, and then remove the brine via a syringe. Add sodium sulfate and allow to stir for 10 minutes, stop stirring, and transfer via a filtered cannula the remaining organic layer into a receiving flask. Remove solvent under vacuum with applied heat from an oil bath.

Use degassed, dry deuterated solvent for NMR samples and seal under inert atmosphere or use an NMR tube fitted with a Young's tap.

Appendix 2: Computational Methods

Below is the Python script used to generate descriptors for each of the ligand conformers generated from the Monte Carlo Multiple Minimum Search.

```
import os
import subprocess
import re
import numpy as np
import vg
import py2sambvca as p2s
from wSterimol.wsterimol.sterimoltools import calcSterimol

# dictionaries to convert integer to ligand or substituent
from functions import substituent_rosetta, ligand_rosetta
# open output file
outfile = open(r'maestro data.txt', 'w')
outfile.write("""conformer number\tligand num\tligand name\tP-A1 bond 
length\tP-A2 bond length\tP-A3 bond length\tS1-P-S2 angle\tS2-P-S3-S1 
angle\tS3-P-S1 angle\tcone angle\tcone angle Pd\tburied 
volume\tL\tB1\tB5\n""")
# open temporary file for later use
temp = open("py2sambvca_tmp.pdb","w")
temp.close()
# loop through each of the resulting conformers
for conf_pdb in os.listdir(r'conformers_as_pdb'):
    breakon = breakon + 1
    # # print(conf_pdb)
    # open the file, read data
    with open(r'conformers_as_pdb/' + conf_pdb, 'r') as file:
        conf_data = file.readlines()

    # get the title, which is the ligand number and translate to the 
ligand name
    ligand_num = conf_data[3].split()[1]  # 4th line, trailing number
    ligand_name = ligand_rosetta[ligand_num]

    # get the ID's of phosphorus
    pattern = re.compile(r"^HETATM\s*(\d*).*0\.00           P  ")
    for line in conf_data:
        m = pattern.search(line)
        if m:
            phos_id = m[1]
            break
    # get the id's of the three substituents based on the phos id
    pattern = re.compile(r"^CONECT   [ ]?{}   [ ]?(\d*)   [ ]?(\d*)   
[ ]?(\d*)$".format(phos_id))
    for line in conf_data:
        m = pattern.search(line)
        if m:
            sub_1_id = m[1]
            sub_2_id = m[2]
            sub_3_id = m[3]
            break
```

```
    # read coordinates of phosphorus and the three substituent
carbons
    pattern = re.compile(r"^HETATM\s*{} .* ([-]?\d*\.\d*)  [ ]?([-
]?\d*\.\d*)  [ ]?([-]?\d*\.\d*)\s*1\.00\s*0\.00.*$".format(phos_id))
    for line in conf_data:
        m = pattern.search(line)
        if m:
            phos_x = float(m[1])
            phos_y = float(m[2])
            phos_z = float(m[3])
            break
    pattern = re.compile(r"^HETATM\s*{} .* ([-]?\d*\.\d*)  [ ]?([-
]?\d*\.\d*)  [ ]?([-]?\d*\.\d*)\s*1\.00\s*0\.00.*$".format(sub_1_id))
    for line in conf_data:
        m = pattern.search(line)
        if m:
            sub_1_x = float(m[1])
            sub_1_y = float(m[2])
            sub_1_z = float(m[3])
            break
    pattern = re.compile(r"^HETATM\s*{} .* ([-]?\d*\.\d*)  [ ]?([-
]?\d*\.\d*)  [ ]?([-]?\d*\.\d*)\s*1\.00\s*0\.00.*$".format(sub_2_id))
    for line in conf_data:
        m = pattern.search(line)
        if m:
            sub_2_x = float(m[1])
            sub_2_y = float(m[2])
            sub_2_z = float(m[3])
            break
    pattern = re.compile(r"^HETATM\s*{} .* ([-]?\d*\.\d*)  [ ]?([-
]?\d*\.\d*)  [ ]?([-]?\d*\.\d*)\s*1\.00\s*0\.00.*$".format(sub_3_id))
    for line in conf_data:
        m = pattern.search(line)
        if m:
            sub_3_x = float(m[1])
            sub_3_y = float(m[2])
            sub_3_z = float(m[3])
            break

    # calculate interatomic distance  https://github.com/lace/vg
    phos_sub_1_distance = vg.euclidean_distance(np.array([sub_1_x,
sub_1_y, sub_1_z]), np.array([phos_x, phos_y, phos_z]))
    phos_sub_2_distance = vg.euclidean_distance(np.array([sub_2_x,
sub_2_y, sub_2_z]), np.array([phos_x, phos_y, phos_z]))
    phos_sub_3_distance = vg.euclidean_distance(np.array([sub_3_x,
sub_3_y, sub_3_z]), np.array([phos_x, phos_y, phos_z]))
    # calculate bond angle - treat sub - P bond as a vector,
translate by phos coords
    sub_1_phos_sub_2_angle = vg.angle(np.array([sub_1_x-phos_x,
sub_1_y-phos_y, sub_1_z-phos_z]),
                                        np.array([sub_2_x-phos_x,
sub_2_y-phos_y, sub_2_z-phos_z]))
```

```
    sub_2_phos_sub_3_angle = vg.angle(np.array([sub_2_x-phos_x,
sub_2_y-phos_y, sub_2_z-phos_z]),
                                    np.array([sub_3_x-phos_x,
sub_3_y-phos_y, sub_3_z-phos_z]))
    sub_3_phos_sub_1_angle = vg.angle(np.array([sub_3_x-phos_x,
sub_3_y-phos_y, sub_3_z-phos_z]),
                                    np.array([sub_1_x-phos_x,
sub_1_y-phos_y, sub_1_z-phos_z]))

    # calculate coordinates for dummy palladium atom - using
methodology from here: https://www.molnac.unisa.it/OMtools/SambVca-
Manual.html#Point-03
    # average of three substituents to get point above phsosphorus
    above_phos_x = (sub_1_x + sub_2_x + sub_3_x) / 3
    above_phos_y = (sub_1_y + sub_2_y + sub_3_y) / 3
    above_phos_z = (sub_1_z + sub_2_z + sub_3_z) / 3
    # line of phosphorus and this point is the axis for the metal
    # use this vector, normalize (unit vector), translate to
phosphorus, scale by appropriate bond length
    metal_vector = vg.normalize(np.array([phos_x-above_phos_x,
phos_y-above_phos_y, phos_z-above_phos_z]))
    metal_vector = metal_vector * 2.28  # coords are already in
Angstrom
    metal_x = phos_x + metal_vector[0]
    metal_y = phos_y + metal_vector[1]
    metal_z = phos_z + metal_vector[2]

    # # print("not sure if angle calculations are correct")
    # cone angle - sub to phos vector and above phos to phos vector
    sub_1_angle = vg.angle(np.array([sub_1_x-phos_x, sub_1_y-phos_y,
sub_1_z-phos_z]),
                            np.array([above_phos_x-phos_x,
above_phos_y-phos_y, above_phos_z-phos_z]))
    sub_2_angle = vg.angle(np.array([sub_2_x-phos_x, sub_2_y-phos_y,
sub_2_z-phos_z]),
                            np.array([above_phos_x-phos_x,
above_phos_y-phos_y, above_phos_z-phos_z]))
    sub_3_angle = vg.angle(np.array([sub_3_x-phos_x, sub_3_y-phos_y,
sub_3_z-phos_z]),
                            np.array([above_phos_x-phos_x,
above_phos_y-phos_y, above_phos_z-phos_z]))
    cone_angle = (sub_1_angle + sub_2_angle + sub_3_angle) / 3
    # # print((sub_1_angle + sub_2_angle + sub_3_angle))
    # # print(cone_angle)
    # cone angle with dummy Pd - metal to sub vector and metal to
phos vector
    sub_1_angle_Pd = vg.angle(np.array([sub_1_x-metal_x, sub_1_y-
metal_y, sub_1_z-metal_z]),
                            np.array([phos_x-metal_x, phos_y-metal_y,
phos_z-metal_z]))
    sub_2_angle_Pd = vg.angle(np.array([sub_2_x-metal_x, sub_2_y-
metal_y, sub_2_z-metal_z]),
```

```
                            np.array([phos_x-metal_x, phos_y-metal_y,
phos_z-metal_z]))
    sub_3_angle_Pd = vg.angle(np.array([sub_3_x-metal_x, sub_3_y-
metal_y, sub_3_z-metal_z]),
                            np.array([phos_x-metal_x, phos_y-metal_y,
phos_z-metal_z]))
    cone_angle_Pd = (sub_1_angle_Pd + sub_2_angle_Pd +
sub_3_angle_Pd) / 3
    # # print((sub_1_angle_Pd + sub_2_angle_Pd + sub_3_angle_Pd))
    # # print(cone_angle_Pd)

    # for calculation of buried volume and sterimol, need to add Pd
atom
    file_num = re.split(r"[\_\.]", conf_pdb)[1]
    with open(r'conformers_as_xyz/' + r'lig_{}.xyz'.format(file_num))
as xyz_file:
        xyz_data = xyz_file.readlines()
        # num_atoms = len(xyz_data) - 2
    xyz_data = [str(int(xyz_data[0])+1) + "\n"] + xyz_data[1:2] +
["Pd {}  {}  {}\n".format(str(metal_x), str(metal_y), str(metal_z))]
+ xyz_data[2:]
    with open(r'py2sambvca_tmp.xyz', "w") as out_xyz:
        out_xyz.writelines(xyz_data)

    # buried volume https://www.molnac.unisa.it/OMtools/SambVca-
Manual.html
    buried_vol = p2s.py2sambvca(r'py2sambvca_tmp.xyz', [1], [1,
int(phos_id)+1], [int(sub_1_id)+1, int(sub_1_id)+1, int(sub_1_id)+1])
    buried_vol.write_input()
    buried_vol.calc()
    # bv = buried_vol.get_buried_vol()
    bv = 4
    # buried volume is an extension of cone angle with dummy Pd

    # calculate the sterimol parameters, adapted from
https://github.com/bobbypaton/wSterimol
    # generate pdb file with new metal
    path = os.getcwd() + r'/'
    # # print("/usr/bin/obabel", "-ixyz", path+"py2sambvca_tmp.xyz",
"-opdb > py2sambvca_tmp.pdb")
    # # subprocess.run(["/usr/bin/obabel", "-ixyz",
path+"py2sambvca_tmp.xyz", "-opdb > py2sambvca_tmp.pdb"])
    os.system("/usr/bin/obabel -ixyz " + path + "py2sambvca_tmp.xyz -
opdb > py2sambvca_tmp.pdb")
    # calc sterimol requires the file path, not the opened file
    temp = calcSterimol(r'py2sambvca_tmp.pdb', 'bondi', 1,
int(phos_id)+1)
    lval = temp.lval
    B1 = temp.B1
    B5 = temp.newB5
    # # print(lval)
    # # print(B1)
```

```
    # # print(B5)

outfile.write("""{}\t{}\t{}\t{}\t{}\t{}\t{}\t{}\t{}\t{}\t{}\t{}\t{}\t
{}\t{}\n""".format(
        file_num, ligand_num, ligand_name, phos_sub_1_distance,
phos_sub_2_distance, phos_sub_3_distance,
        sub_1_phos_sub_2_angle, sub_2_phos_sub_3_angle,
sub_3_phos_sub_1_angle,
        cone_angle, cone_angle_Pd, bv, lval, B1, B5
    ))

buried_vol.clean_files()
os.remove("py2sambvca_tmp.xyz")
os.remove("py2sambvca_tmp.pdb")
```

The following two scripts are used from the linux command line with appropriate software installed to convert a given SMILES string to a folder of minimum energy conformers.

```
#!/bin/bash

# reads through a list of SMILES strings and calls a MCMM on each
# first command line argument - name of the input file
# second command line argument - first index of molecule to read
# third command line argument - last line to read

moleculeNumber=1
while IFS= read -r molecule; do
        # skip molecules not in the range
        if [ $moleculeNumber -lt $2 ]; then
                moleculeNumber=$((moleculeNumber + 1))
                continue
        fi

        ebreak=0
        while [ ! -e virtualHTE-
output/molecule"${moleculeNumber}".zip ]
        do
                echo " # # # # # "
                echo "Working on" $moleculeNumber $molecule
                ./SMILES2confs $molecule virtualHTE-
output/molecule"${moleculeNumber}".zip
                BACK_PID=$!
                wait $BACK_PID
                ebreak=$((ebreak + 1))
                if [ $ebreak -gt 3 ]; then
                        echo "Unable to simulate molecule"
$moleculeNumber $molecule
                        break
                fi
                # attempt to fix license error #2, maybe bash is
somehow doing some kind of
```

```
                # loop unwinding and trying to execute multilple
simulations at once?
                # let's wait on the PID until it return

        done

        # break at the end of the specified range
        if [ $moleculeNumber -eq $3 ]; then
                break
        fi
        moleculeNumber=$((moleculeNumber + 1))
done < $1
```

`SMILES2confs` executable referenced in the code above:

```
#!/bin/bash

start=`date +%s`

# this shell script uses executables included with MacroModel to read
a SMILES
# string from the command line and perform a Monte Carlo Multiple
Minimum
# conformational search

# delete previous output, if it exists
if ([ ! -z "$2" ] &&  [ -e "$2" ]) || [ -e SMILES2confs_out.zip ];
then
        echo "WARNING: Previous SMILES2confs output found,
overwriting."
        rm SMILES2confs_out.zip "$2" >> /dev/null 2>&1
fi

# $1 is the SMILES string, provided on the command line as "CP(C)C"

# write the smiles string to a temporary file
echo $1 > SMILES2confs_tmp.smi

# use structvonvert to convert this to a sd file with 3d coords
/opt/schrodinger2020-2/utilities/babel -ismi SMILES2confs_tmp.smi -
omol2 SMILES2confs_tmp.mol2 --gen3d

# delete the SMILES file
rm SMILES2confs_tmp.smi

# use converter to go from MOL2 format to maestro input format
/opt/schrodinger2020-2/utilities/mol2convert -imol2
SMILES2confs_tmp.mol2 -omae SMILES2confs_tmp.mae >> /dev/null 2>&1
if [ $? == 1 ]; then
        echo "MOL2 to MAE conversion completed with warnings."
fi

# delete the MOL2 file
```

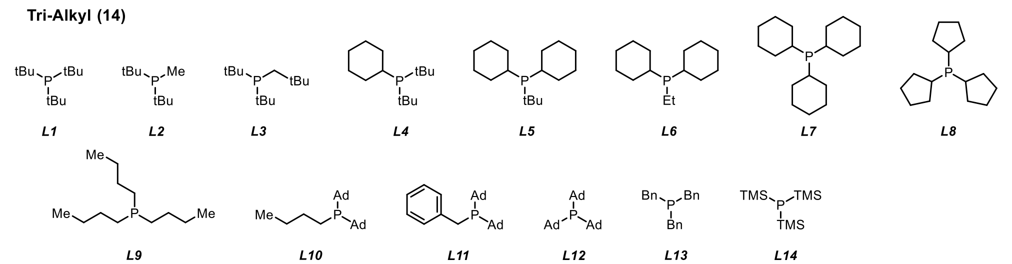
```
# rm SMILES2confs_tmp.mol2

# write the macromodel input file
echo "JOB_TYPE CONFSEARCH
INPUT_STRUCTURE_FILE SMILES2confs_tmp.mae
CONFSEARCH_METHOD mcmm
OUTCONFS_PER_SEARCH 50
FORCE_FIELD MM3*
SOLVENT CHCl3
DIELECTRIC_CONSTANT 1.0
CHARGES_FROM Force field
CUTOFF None
MINI_METHOD PRCG
CONFSEARCH_STEPS 500
MAXIMUM_ITERATION 500
CONVERGE_ON Gradient
CONVERGENCE_THRESHOLD 0.05
ENERGY_WINDOW 3.0
CONFSEARCH_STEPS_PER_ROTATABLE 50
ELECTROSTATIC_TREATMENT Constant dielectric" >>
SMILES2confs_maestro_job

# run the job
macromodel SMILES2confs_maestro_job

# remove the job file and mae file
rm SMILES2confs_maestro_job SMILES2confs_tmp.mae

# every second, check to see if the calculation is done
ebreak=1
while [ ! -e SMILES2confs_maestro_job-out.mae ]
do
        echo -n "."
        sleep 1
        ebreak=$((ebreak + 1))
        if [ $ebreak -gt 600 ]; then
                echo "Timeout - printing log file:"
                cat SMILES2confs_maestro_job.log
                exit 0
        fi
done
echo "simulation complete!"
echo -n "Writing output..."


# loop through output structures, converting to pdb and xyz, and zip
x=1
while [ $x -lt 51 ] # config file is set to only allow 50 structures
do
        # redirect STDOUT to a new mae file and STDERR into a tmp
file
```

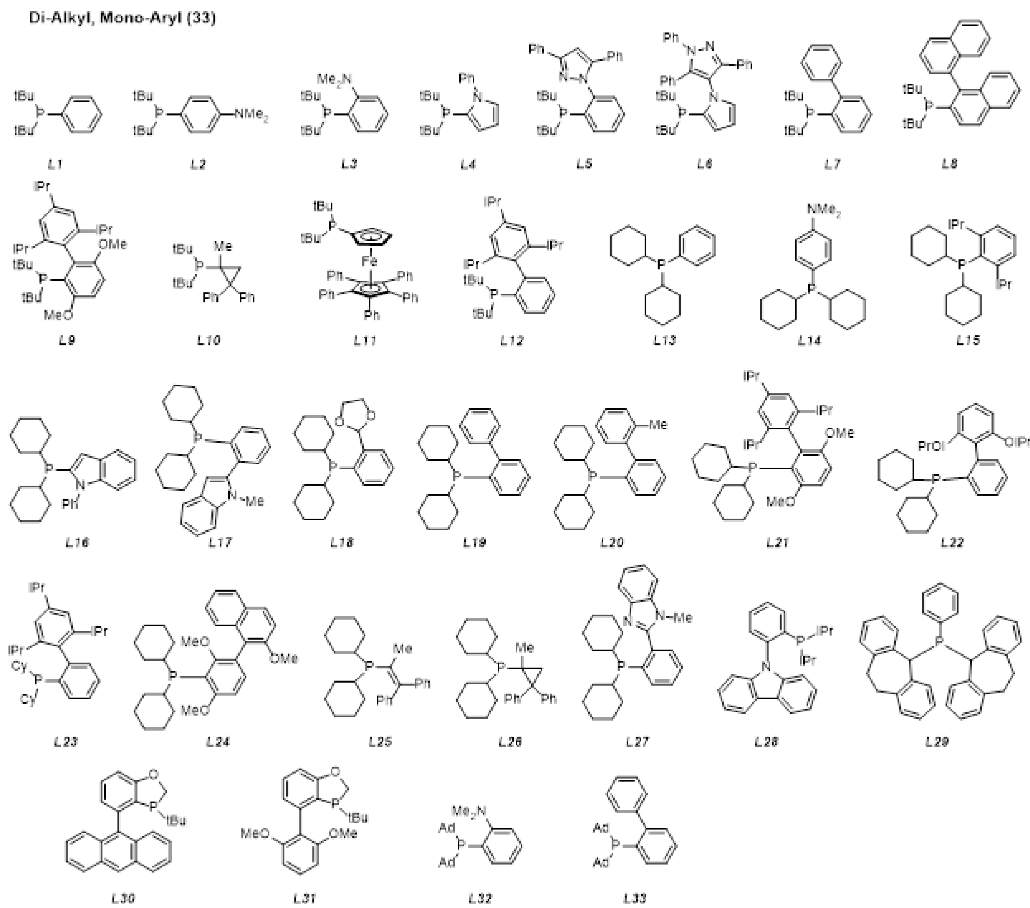
```
        /opt/schrodinger2020-2/utilities/maesubset -n "${x}"\:"${x}"
SMILES2confs_maestro_job-out.mae > currentstruct"${x}".mae 2> err.tmp

        # if you have already included all the structures and are
trying to run with a bad range
        # masesubset just warns you but does not return 1, so we have
to test if size of STDERROR
        # is greater than 0
        if [ -s err.tmp ]; then
                break
        fi

        # attempt to fix license problem #1 - maybe using two
schrodinger utilities one after the other
        # is causing conflicts?
        # sleep 5

        # convert the MAE into a pdb and then the pdb into an xyz
        /opt/schrodinger2020-2/utilities/structconvert -imae
currentstruct"${x}".mae -opdb currentstruct"${x}".pdb >> /dev/null
2>&1
        if [ $? == 1 ]; then
                echo "MAE to PDB conversion completed with warnings."
        fi

        # also trying to fix license problem
        # sleep 5
        /opt/schrodinger2020-2/utilities/babel -ipdb
currentstruct"${x}".pdb -oxyz currentstruct"${x}".xyz >> /dev/null
2>&1
        if [ $? == 1 ]; then
                echo "PDB to XYZ conversion completed with warnings."
        fi
        x=$((x + 1))
done

# remove temporary error file and maestro files
rm err.tmp
# for some reason, this command does not work as expected if it is
the last command in the script
# cat SMILES2confs_maestro_job.log
rm SMILES2confs_maestro_job*

echo " done."
echo "Compressing output structures:"
# zip all the resulting files, then delete them
if [ -z "$2" ]; then
        zip SMILES2confs_out.zip currentstruct*.xyz
currentstruct*.pdb
else
        zip "$2" currentstruct*.xyz currentstruct*.pdb
fi
```

```
rm currentstruct*
end=`date +%s`
echo "Completed in" $((end-start)) "seconds."
```

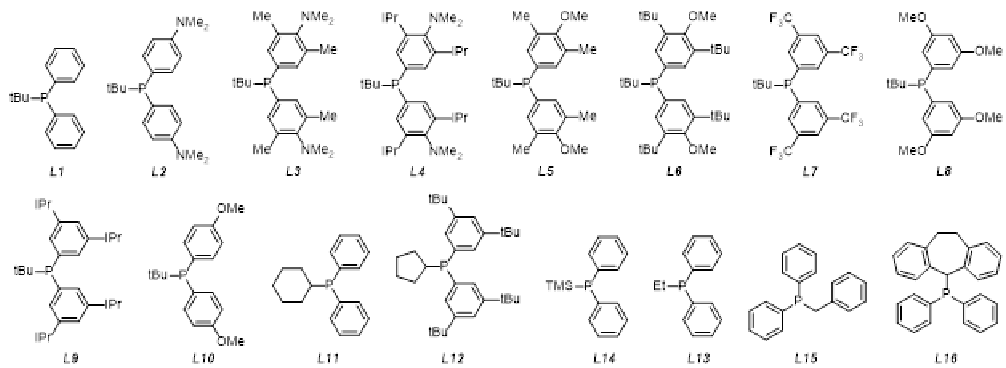
Appendix 3: Complete HTE Data

Below is a complete list of all ligands evaluated in the HTE screening used to generate data for the model reaction, as well as the raw tabulated data.

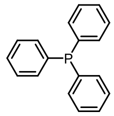




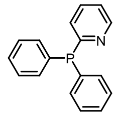
Mono-Alkyl, Di-Aryl (16)



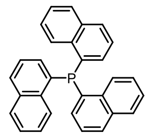
Tri-Aryl (31)



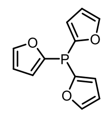
L1



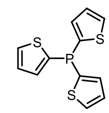
L2



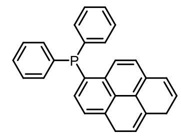
L3



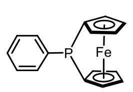
L4



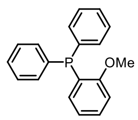
L5



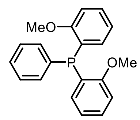
L6



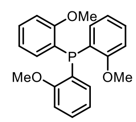
L7



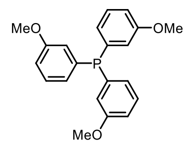
L8



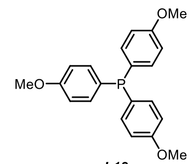
L9



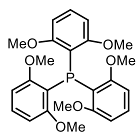
L10



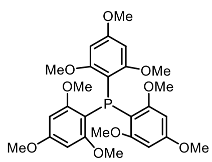
L11



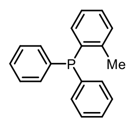
L12



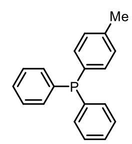
L13



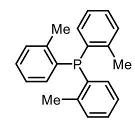
L14



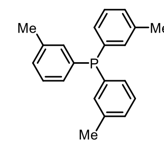
L15



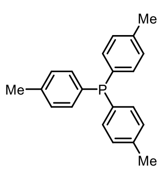
L16



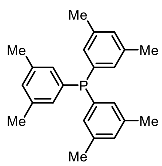
L17



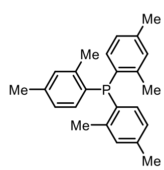
L18



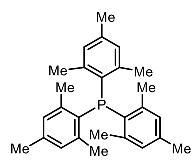
L19



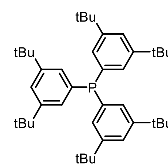
L20



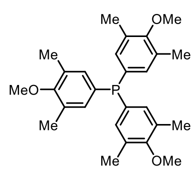
L21



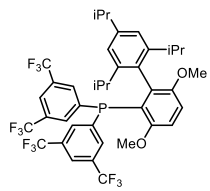
L22



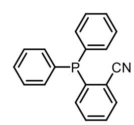
L23



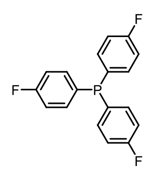
L24



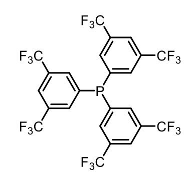
L25



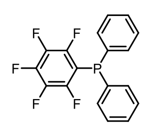
L26



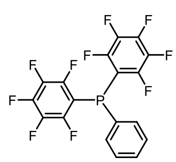
L27



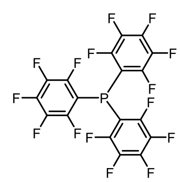
L28



L29



L30



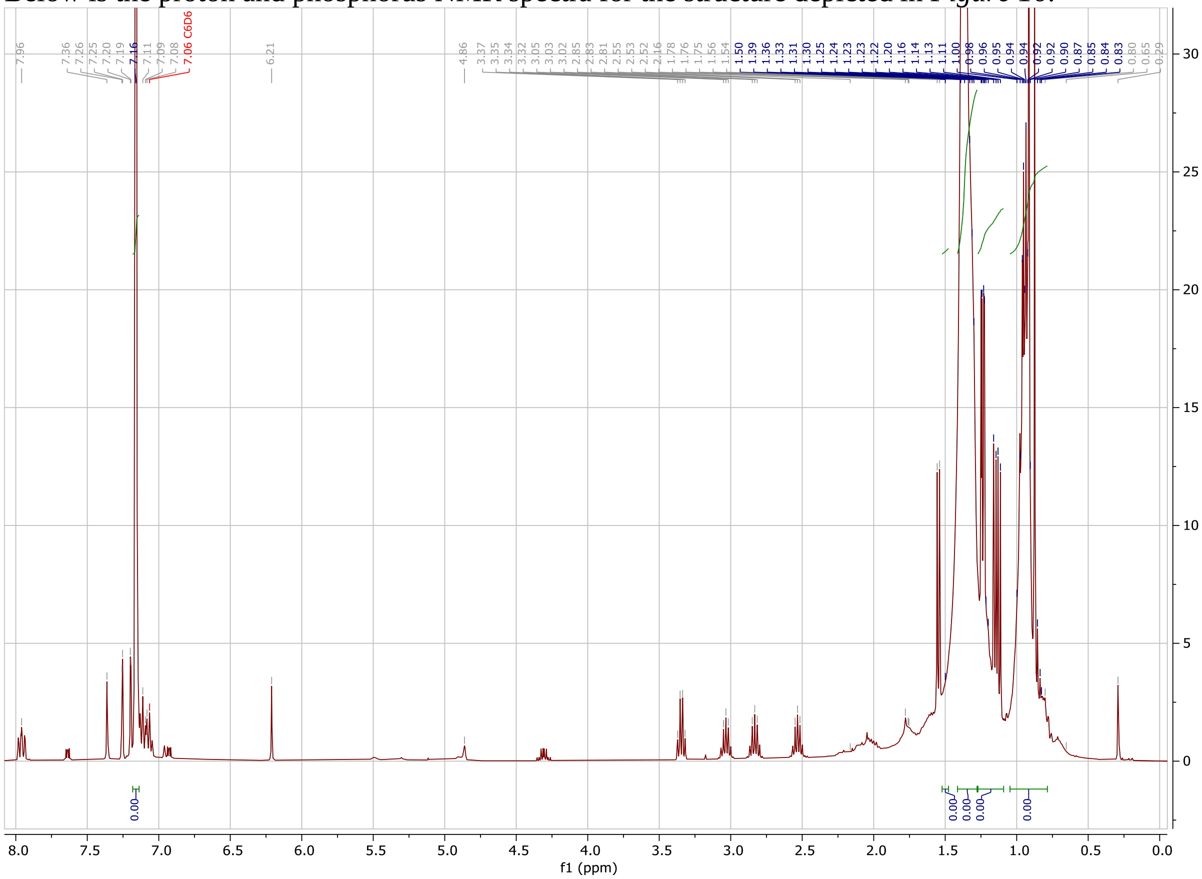
L31

Ligand Name	I_yiel	VI_yiel	VII_yiel	III_yiel	IV_yiel
(tBu)P(tBu)(Me)	4.51	0.1	0.21	0.98	2.86
P(Bu)3	3.75	0.11	0.18	0.55	2.44
(tBu)P(tBu)2	5.19	0.08	0.22	2.24	4.39
P(Cy)3	5.44	0.29	0.17	1.31	1.23
(Ad)P(Ad)(Bu)	0.5	0	0	7	1.71
Ph2MeP	6.22	0.12	0.32	2.48	4.97
(tBu)P(Ph)2	5.6	0.11	0.3	2.28	4.85
(Cy)P(Cy)(Ph)	11.04	0.49	0.13	3.67	1.56
P(p-OMePh)3	5.07	0.23	0.24	0	1.6
SPhos	1.65	0	0	0.7	0
tBuXPhos	0.9	0.21	0.35	0	1.17
XPhos	1	0.15	0.2	0.5	1.36
P(2-furyl)3	5.04	0.15	0.28	1.59	3.9
P(Ph)3	10.57	0.33	0.41	0	1.42
P(o-MePh)3	19.58	0.95	0.6	3.54	5.57
P(3,5-diCF3Ph)3	0	0	16.26	0	1.08
P(1-Nphth)3	1.35	0	0	0	0.75
RuPhos	2.17	0	0	0.52	0.89
BrettPhos	0.84	0.3	0.38	0	2.24
tBuBrettPhos	1.76	0.18	0.24	0.52	2.26
JohnPhos	0.68	0.16	0.14	0	1.35
DavePhos	0.93	0	0	0	0.62
JackiePhos	1.72	0	0.12	0	1.6
SymPhos	0.44	0.45	0	7.43	0.83
(tBu)P(tBu)(p-NMe2Ph)	0	0	0	3.58	1.53
DrewPhos	17.92	0.8	0	0.66	1.66
P(Cp)3	1.95	0	0	0.67	2.03
(tBu)P(Cy)2	5.29	0.09	0.28	2.2	4.6
(tBu)P(tBu)(Cy)	6.79	0.12	0.31	2.69	5.77
(tBu)P(3,5-ditBuPh)2	11.58	0.62	0.37	12.2	5.4
(Cy)P(Ph)2	14.62	0	0.48	0	1.95
(Et)P(Ph)2	4.93	0.09	0.17	0.94	3.17
(Bn)P(Ph)2	6.02	0	0	0	0.58
P(Bn)3	5.32	0.17	0	0.32	0.62
P(2,4-diMePh)3	13.43	0.85	0.44	1.8	3.15
AdBrettPhos	1.36	0.14	0.12	0.37	1.71
P(Ph)2(TMS)	3.62	0.06	0.16	0.63	2.42
(Cy)P(Cy)(p-NMe2Ph)	5.03	0.41	0.07	6.1	2.75
CyJohnPhos	1.44	0	0	1.23	0.8
P(tBu)(p-NMe2Ph)2	5.53	0.36	0.27	5.36	4.73
P(tBu)(3,5-diMe-4-NMe2Ph)2	11.86	0.95	0.35	4.72	4.2
P(tBu)(3,5-diiPr-4-NMe2Ph)2	10.37	0.81	0.39	11.16	4.33
P(tBu)(3,5-diMe-4-OMePh)2	20.46	1.15	0.56	4.63	5.36
P(tBu)(3,5-ditBu-4-OMePh)2	13.24	0.88	0.41	11.88	4.77
P(tBu)(3,5-bisCF3Ph)2	4.28	0.08	0.17	0.71	3.07
P(m-tol)3	9.5	0.22	0.56	0	1.36
P(p-tol)3	8.79	0.35	0.42	0	2.33
P(tBu)(3,5-bisOMePh)2	25.79	1.01	0.66	4.87	5.13
P(p-FPh)3	0	0	0	0	0
P(2,4,6-triMePh)3	6.73	0.1	0.27	1.95	4.88
P(2,4,6-triOMePh)3	4.52	1.45	0.16	0.27	2.15
P(tBu)(3,5-bisiPrPh)2	12.3	0.77	0.39	11.48	4.73
P(tBu)(p-OMePh)2	17.12	1.13	0.46	5.63	5.25
P(Ph)2(pyrene)	13.18	0.96	0.22	0.33	2.3
P(Ph)2(C6F5)	7.21	0.41	0.08	0.58	2.01
P(Ph)2(o-MePh)	20.06	1.25	0.25	0.53	2.57
P(Ph)2(o-OMePh)	21.44	1.68	0.5	0.22	8.51
TyrannoPhos	13.34	0.3	0.16	0	0.96
P(Cy)2(2,6-diiPrPh)	0.88	0	0.1	2.12	2.05
P(Ad)3	0.52	0.06	0.09	2.34	3.39

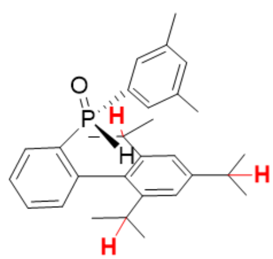
cataCXium PtB	7.23	0.11	0.26	2.28	5.24
Catacxium PlnCy	7.34	0.14	0.32	2.31	5.33
(Et)P(Cy)2	5.94	0.09	0.22	1.45	4.04
P(TMS)3	7.58	0.11	0.23	2.31	5.17
P(Ph)(o-OMePh)2	7.62	0.3	0.34	2.62	0
P(m-OMePh)3	10.66	0.31	0.47	1.27	0
Cy-vBRIDP	0	0	0	0	0
P(3,5-diMe-4-OMePh)3	7.73	0.25	0.85	1.66	0
P(Ph)2(p-MePh)	11.75	0.34	0.48	1.85	0
cataCXium® POMeCy	1.96	0.09	0.89	2	1.17
Cy-cBRIDP	0.38	0.19	0	0	0
cBRIDP	0.64	0.16	0.25	1.21	0.39
trop2-PPh	0.72	0.16	0.32	0	0
P(2-thienyl)3	0.44	0.21	0	0	0
P(tBu)2(o-NMe2Ph)	2.81	0.09	0.16	0.51	2.25
cataCXium ABn	1.83	0.06	1.27	3.45	1.48
DMPP	3.72	1.4	0.17	0.45	1.62
1-Methyl-2-(2-dicyclohexylphosphinophenyl)-1H-benzoimidazole	5.8	0	0	0	0
P(3,5-diMePh)3	9.02	0.21	0.81	1.7	0.3
P(o-OMePh)3	11.15	1.41	0.21	0.52	4.08
P(Ph)(C6F5)2	0.45	0.38	0.05	0	0
(iPr)P(iPr)(Ph-carbazole)	2.1	0.48	0.61	0.63	0.38
tBuMePhos	0.85	0.13	0.2	0.7	0.89
Cy2(o-tolyl)P	8.41	0.62	1.1	4.14	2.22
Ph2(iPr)P	2.8	0.11	0.2	0.58	2.51
Cy2(2,4,6-Me-Ph)P	1.06	0.41	0.29	0.52	0.34
2-[Bis(3,5-trifluoromethylphenylphosphino)-3,6-dimethoxy]-2',6'-dimethylamino-1,1'-biphenyl	1.26	0.12	0.29	0.7	0.79
2-[Bis(3,5-trifluoromethylphenylphosphino)-3,6-dimethoxy]-2',6'-di-i-propoxy-1,1'-biphenyl	6.01	0.23	0.34	0.84	4.05
(S)-NMDPP	24.7	0.54	0.6	13.43	7.62
t-BuMePhos	0.81	0.11	0.21	0.38	1.23
PhDavePhos	4.89	0.15	0.23	0.47	1.34
cataCXium® PICy	5.55	0.22	0.43	1.3	3.96
VPhos	1.3	0.12	0.22	0.6	1.24
(tBu)P(tBu)(Ph)	1.26	0.03	0.04	2.32	2.05
EvanPhos	0.8	0.82	1.16	4.21	0.91
cataCXium® POMeCy	1.96	0.06	1.57	0	1.16
PhCPhos	0.38	0	0	0	0.75
(t-Bu)PhCPhos	0.63	0.16	0.46	0	0.31
Diphenyl[4-(N,N-dimethylamino)phenyl]phosphine	6.62	0.31	0.73	0	1.95
TOP	0	0	0	0	0
2-[Bis(1-methylethyl)phosphino]-1-phenyl-1H-pyrrole	3.25	0.08	0.72	4.14	2.02
1-[2,6-Bis(cyclohexyloxy)phenyl]-2-(di-tertbutylphosphaneyl)-1H-imidazole	0.95	0.17	0.5	0.52	4.1
Mor-DalPhos	0	0	0	0	0.67
CPhos	0.55	0	0	0	0.95
2-Dicyclohexylphosphino-4-(N,N-dimethylamino)-1,1'-biphenyl	4.31	0	0.24	1.73	0.87
PhenCar-Phos	3.81	0	0.04	1.04	1.77
Et PhenCar-Phos	1.12	0	0	0.38	0.89
2-Diphenylphosphino-6-methylpyridine	0.49	0	0	0	0.51
Ph PhenCar-Phos	0.81	0	0	0.35	0.79
[cataCXium® POMetB]	2.48	0.12	0.2	0.72	1.98
PPh2-Andole-Phos	8.14	1.06	0.16	0.38	2.21
tBuDavePhos	0.38	0.1	0.15	1.07	0.74

Appendix 4: Spectra

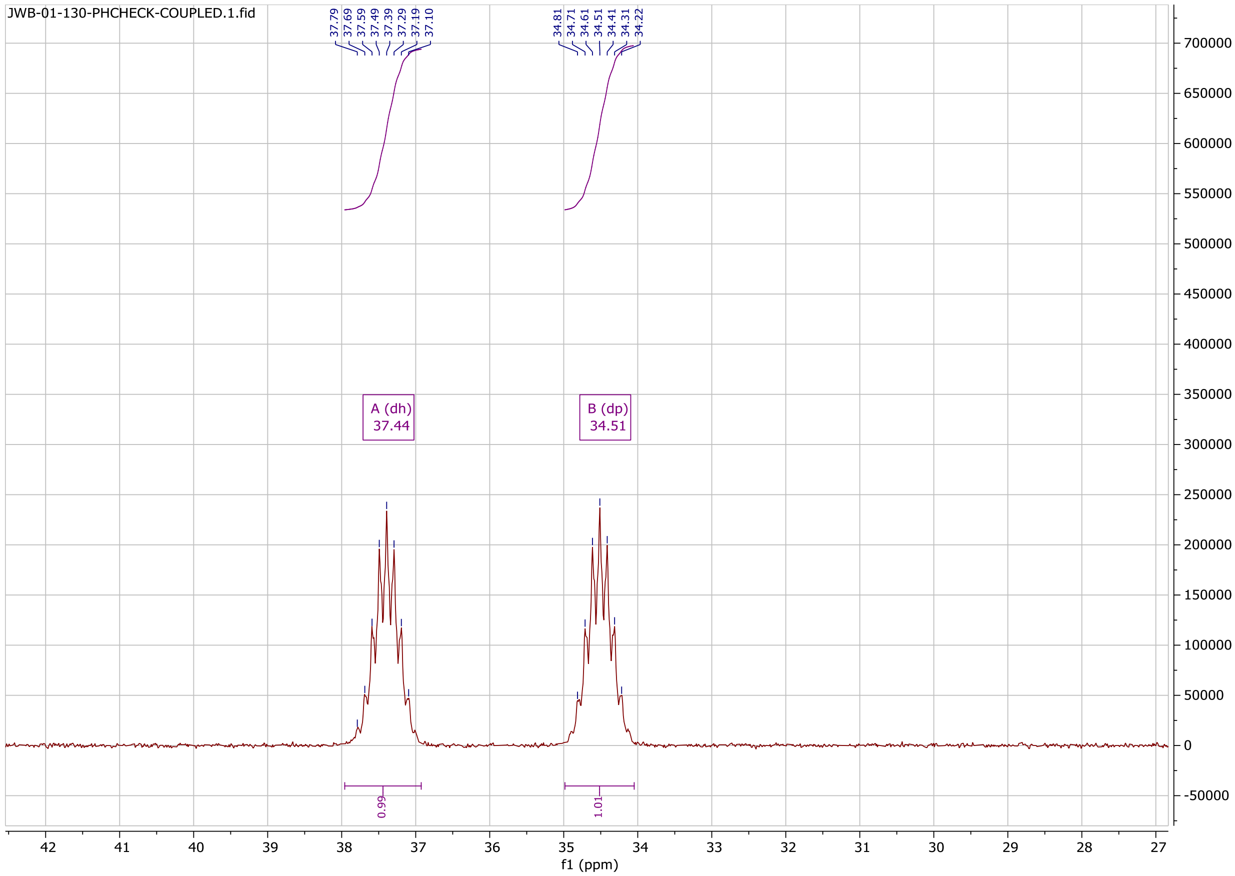
Below is the proton and phosphorus NMR spectra for the structure depicted in *Figure 16*.



While this spectrum is not of sufficient purity for quantitation, the presence of certain peaks yield sufficient information for a structural determination. The three multiplets located between 2.5 and 3.1 ppm correspond to the three protons depicted in red in the structure that follows:



This provides sufficient evidence that there are three unique substituents on the central phosphorus. The presence of the hydrogen on phosphorus is confirmed via PNMR.



This decoupled phosphorus NMR shows splitting, reflecting the presence of the proton on the phosphorus atom.

References

1. Heck, R. F., Acylation, methylation, and carboxyalkylation of olefins by Group VIII metal derivatives. *Journal of the American Chemical Society* **1968,** *90* (20), 5518-5526.
2. Heck, R. F., The addition of alkyl- and arylpalladium chlorides to conjugated dienes. *Journal of the American Chemical Society* **1968,** *90* (20), 5542-5546.
3. Heck, R. F., Aromatic haloethylation with palladium and copper halides. *Journal of the American Chemical Society* **1968,** *90* (20), 5538-5542.
4. Heck, R. F., The palladium-catalyzed arylation of enol esters, ethers, and halides. A new synthesis of 2-aryl aldehydes and ketones. *Journal of the American Chemical Society* **1968,** *90* (20), 5535-5538.
5. Heck, R. F., Allylation of aromatic compounds with organopalladium salts. *Journal of the American Chemical Society* **1968,** *90* (20), 5531-5534.
6. Heck, R. F.; Nolley, J. P., Palladium-catalyzed vinylic hydrogen substitution reactions with aryl, benzyl, and styryl halides. *The Journal of Organic Chemistry* **1972,** *37* (14), 2320-2322.
7. Korch, K. M.; Watson, D. A., Cross-Coupling of Heteroatomic Electrophiles. *Chemical Reviews* **2019,** *119* (13), 8192-8228.
8. Verloop, A., Development and application of new steric substituent parameters in drug design. **1976**.
9. Muratov, E. N.; Bajorath, J.; Sheridan, R. P.; Tetko, I. V.; Filimonov, D.; Poroikov, V.; Oprea, T. I.; Baskin, I. I.; Varnek, A.; Roitberg, A.; Isayev, O.; Curtalolo, S.; Fourches, D.; Cohen, Y.; Aspuru-Guzik, A.; Winkler, D. A.; Agrafiotis, D.; Cherkasov, A.; Tropsha, A., QSAR without borders. *Chemical Society Reviews* **2020,** *49* (11), 3525-3564.
10. De Jesus Silva, J.; Ferreira, M. A. B.; Fedorov, A.; Sigman, M. S.; Copéret, C., Molecular-level insight in supported olefin metathesis catalysts by combining surface organometallic chemistry, high throughput experimentation, and data analysis. *Chemical Science* **2020,** *11* (26), 6717-6723.
11. Rinehart, N. I.; Zahrt, A. F.; Henle, J. J.; Denmark, S. E., Dreams, False Starts, Dead Ends, and Redemption: A Chronicle of the Evolution of a Chemoinformatic Workflow for the Optimization of Enantioselective Catalysts. *Accounts of Chemical Research* **2021,** *54* (9), 2041-2054.
12. Ahneman, D. T.; Estrada, J. G.; Lin, S.; Dreher, S. D.; Doyle, A. G., Predicting reaction performance in C–N cross-coupling using machine learning. *Science* **2018,** *360* (6385), 186-190.
13. Seyferth, D., The Grignard Reagents. *Organometallics* **2009,** *28* (6), 1598-1605.
14. Beletskaya, I. P.; Cheprakov, A. V., The Heck Reaction as a Sharpening Stone of Palladium Catalysis. *Chemical Reviews* **2000,** *100* (8), 3009-3066.
15. Tasker, S. Z.; Gutierrez, A. C.; Jamison, T. F., Nickel-Catalyzed Mizoroki–Heck Reaction of Aryl Sulfonates and Chlorides with Electronically Unbiased Terminal Olefins: High Selectivity for Branched Products. *Angewandte Chemie International Edition* **2014,** *53* (7), 1858-1861.
16. Kohn, W., Nobel Lecture: Electronic structure of matter---wave functions and density functionals. *Reviews of Modern Physics* **1999,** *71* (5), 1253-1266.
17. Frisch, M. J.; Trucks, G. W.; Schlegel, H. B.; Scuseria, G. E.; Robb, M. A.; Cheeseman, J. R.; Scalmani, G.; Barone, V.; Petersson, G. A.; Nakatsuji, H.; Li, X.; Caricato, M.; Marenich, A. V.; Bloino, J.; Janesko, B. G.; Gomperts, R.; Mennucci, B.;

Hratchian, H. P.; Ortiz, J. V.; Izmaylov, A. F.; Sonnenberg, J. L.; Williams; Ding, F.; Lipparini, F.; Egidi, F.; Goings, J.; Peng, B.; Petrone, A.; Henderson, T.; Ranasinghe, D.; Zakrzewski, V. G.; Gao, J.; Rega, N.; Zheng, G.; Liang, W.; Hada, M.; Ehara, M.; Toyota, K.; Fukuda, R.; Hasegawa, J.; Ishida, M.; Nakajima, T.; Honda, Y.; Kitao, O.; Nakai, H.; Vreven, T.; Throssell, K.; Montgomery Jr., J. A.; Peralta, J. E.; Ogliaro, F.; Bearpark, M. J.; Heyd, J. J.; Brothers, E. N.; Kudin, K. N.; Staroverov, V. N.; Keith, T. A.; Kobayashi, R.; Normand, J.; Raghavachari, K.; Rendell, A. P.; Burant, J. C.; Iyengar, S. S.; Tomasi, J.; Cossi, M.; Millam, J. M.; Klene, M.; Adamo, C.; Cammi, R.; Ochterski, J. W.; Martin, R. L.; Morokuma, K.; Farkas, O.; Foresman, J. B.; Fox, D. J. *Gaussian 16 Rev. C.01*, Wallingford, CT, 2016.

18. Houk, K. N., Theoretical Studies of Halocarbene Cycloaddition Selectivities. *Tetrahedron* **1985,** *41*, 1555.

19. Leow, J.-L.; Baron, R.; Casey, P. J.; Go, M.-L., Quantitative structure–activity relationship (QSAR) of indoloacetamides as inhibitors of human isoprenylcysteine carboxyl methyltransferase. *Bioorganic & Medicinal Chemistry Letters* **2007,** *17* (4), 1025-1032.

20. Brethomé, A. V.; Fletcher, S. P.; Paton, R. S., Conformational Effects on Physical-Organic Descriptors: The Case of Sterimol Steric Parameters. *ACS Catalysis* **2019,** *9* (3), 2313-2323.

21. Falivene, L.; Credendino, R.; Poater, A.; Petta, A.; Serra, L.; Oliva, R.; Scarano, V.; Cavallo, L., SambVca 2. A Web Tool for Analyzing Catalytic Pockets with Topographic Steric Maps. *Organometallics* **2016,** *35* (13), 2286-2293.

22. Jover, J.; Cirera, J., Computational assessment on the Tolman cone angles for P-ligands. *Dalton Transactions* **2019,** *48* (40), 15036-15048.

23. Vitek, A. K.; Jugovic, T. M. E.; Zimmerman, P. M., Revealing the Strong Relationships between Ligand Conformers and Activation Barriers: A Case Study of Bisphosphine Reductive Elimination. *ACS Catalysis* **2020**, 7136-7145.

24. Niemeyer, Z. L.; Milo, A.; Hickey, D. P.; Sigman, M. S., Parameterization of phosphine ligands reveals mechanistic pathways and predicts reaction outcomes. *Nature Chemistry* **2016,** *8* (6), 610-617.

25. Bunker, A.; Róg, T., Mechanistic Understanding From Molecular Dynamics Simulation in Pharmaceutical Research 1: Drug Delivery. *Frontiers in Molecular Biosciences* **2020,** *7*.

26. Gentile, F.; Yaacoub, J. C.; Gleave, J.; Fernandez, M.; Ton, A.-T.; Ban, F.; Stern, A.; Cherkasov, A., Artificial intelligence–enabled virtual screening of ultra-large chemical libraries with deep docking. *Nature Protocols* **2022,** *17* (3), 672-697.

27. Pillai, N.; Dasgupta, A.; Sudsakorn, S.; Fretland, J.; Mavroudis, P. D., Machine Learning guided early drug discovery of small molecules. *Drug Discovery Today* **2022**.

28. Burns, J. W. *Crow - High Throughput Experimentation Data Retrieval and Presentation GUI*, figshare, 2020.

29. Ingoglia, B. T.; Wagen, C. C.; Buchwald, S. L., Biaryl monophosphine ligands in palladium-catalyzed C–N coupling: An updated User's guide. *Tetrahedron* **2019,** *75* (32), 4199-4211.

30. Tolman, C. A., Phosphorus ligand exchange equilibriums on zerovalent nickel. Dominant role for steric effects. *Journal of the American Chemical Society* **1970,** *92* (10), 2956-2965.

31. Guzei, I. A.; Wendt, M., An improved method for the computation of ligand steric effects based on solid angles. *Dalton Transactions* **2006,** (33), 3991-3999.

32. Mulliken, R. S., Electronic Population Analysis on LCAO-MO Molecular Wave Functions. III. Effects of Hybridization on Overlap and Gross AO Populations. *The Journal of Chemical Physics* **1955,** *23* (12), 2338-2342.
33. Cioslowski, J., A new population analysis based on atomic polar tensors. *Journal of the American Chemical Society* **1989,** *111* (22), 8333-8336.
34. Payard, P.-A.; Perego, L. A.; Grimaud, L.; Ciofini, I., A DFT Protocol for the Prediction of 31P NMR Chemical Shifts of Phosphine Ligands in First-Row Transition-Metal Complexes. *Organometallics* **2020,** *39* (17), 3121-3130.
35. Lodewyk, M. W.; Siebert, M. R.; Tantillo, D. J., Computational Prediction of 1H and 13C Chemical Shifts: A Useful Tool for Natural Product, Mechanistic, and Synthetic Organic Chemistry. *Chemical Reviews* **2012,** *112* (3), 1839-1862.
36. O'Boyle, N. M.; Tenderholt, A. L.; Langner, K. M., cclib: A library for package-independent computational chemistry algorithms. *Journal of Computational Chemistry* **2008,** *29* (5), 839-845.
37. Zhang, G.; Musgrave, C. B., Comparison of DFT Methods for Molecular Orbital Eigenvalue Calculations. *The Journal of Physical Chemistry A* **2007,** *111* (8), 1554-1561.
38. Griffith, J. S.; Orgel, L. E., Ligand-field theory. *Quarterly Reviews, Chemical Society* **1957,** *11* (4), 381-393.
39. Gensch, T.; dos Passos Gomes, G.; Friederich, P.; Peters, E.; Gaudin, T.; Pollice, R.; Jorner, K.; Nigam, A.; Lindner D'Addario, M.; Sigman, M. S.; al, E., A Comprehensive Discovery Platform for Organophosphorus Ligands for Catalysis. *ChemRxiv* **2021**.
40. Schrödinger Inc. *MacroModel Quick Start Guide*, 9.9; 2012.
41. Van Rossum, G.; Drake, F. L., *Python 3 Reference Manual*. CreateSpace: Scotts Valley, CA, 2009.
42. Burns, J., py2sambvca. **2020**.
43. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; Duchesnay, É., Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* **2011,** *12* (85), 2825-2830.
44. Varnek, A.; Baskin, I., Machine Learning Methods for Property Prediction in Chemoinformatics: *Quo Vadis* ? *Journal of Chemical Information and Modeling* **2012,** *52* (6), 1413-1437.
45. Hy, T. S.; Trivedi, S.; Pan, H.; Anderson, B. M.; Kondor, R., Predicting molecular properties with covariant compositional networks. *The Journal of Chemical Physics* **2018,** *148* (24), 241745.
46. You, J.; Liu, B.; Ying, R.; Pande, V.; Leskovec, J., Graph Convolutional Policy Network for Goal-Directed Molecular Graph Generation. *arXiv:1806.02473 [cs, stat]* **2019**.
47. Garijo del Río, E.; Mortensen, J. J.; Jacobsen, K. W., Local Bayesian optimizer for atomic structures. *Physical Review B* **2019,** *100* (10), 104103.
48. Breiman, L., Random Forests. *Machine Learning* **2001,** *45* (1), 5-32.
49. Bhattacharjee, H.; Burns, J.; Vlachos, D., AIMSim: An Accessible Cheminformatics Platform for Similarity Operations on Chemicals Datasets. *ChemRxiv* **2022**.
50. National Center for Biotechnology Information. PubChem Compound Summary for CID 136357, 1-Bromo-3,5-dimethylbenzene. https://pubchem.ncbi.nlm.nih.gov/compound/1-Bromo-3_5-dimethylbenzene.

51. Bruno, N. C.; Buchwald, S. L., Synthesis and Application of Palladium Precatalysts that Accommodate Extremely Bulky Di-tert-butylphosphino Biaryl Ligands. *Organic Letters* **2013,** *15* (11), 2876-2879.
52. Lichtscheidl, A. G.; Ng, V. W. L.; Müller, P.; Takase, M. K.; Schrock, R. R., Molybdenum Monoaryloxide Pyrrolide Alkylidene Complexes That Contain Mono-ortho-substituted Phenyl Imido Ligands. *Organometallics* **2012,** *31* (6), 2388-2394.
53. Kharasch, M.; Tawney, P., Factors determining the course and mechanisms of Grignard reactions. II. The effect of metallic compounds on the reaction between isophorone and methylmagnesium bromide. *Journal of the American Chemical Society* **1941,** *63* (9), 2308-2316.