molSimplify: A Toolkit for Automating Discovery in Inorganic Chemistry

Efthymios I. Ioannidis, Terry Z. H. Gani, and Heather J. Kulik*

We present an automated, open source toolkit for the first-principles screening and discovery of new inorganic molecules and intermolecular complexes. Challenges remain in the automatic generation of candidate inorganic molecule structures due to the high variability in coordination and bonding, which we overcome through a divide-and-conquer tactic that flexibly combines force-field preoptimization of organic fragments with alignment to first-principles-trained metal-ligand distances. Exploration of chemical space is enabled through random generation of ligands and intermolecular complexes from large chemical databases. We validate the generated structures with the root mean squared (RMS) gradients evaluated from density functional theory (DFT), which are around 0.02 Ha/au across a large 150 molecule test set. Comparison of molSimplify results to full optimization with the universal force field reveals that RMS DFT gradients are improved by 40%. Seamless generation of input files, preparation and execution of electronic structure calculations, and post-processing for each generated structure aids interpretation of underlying chemical and energetic trends. © 2016 Wiley Periodicals, Inc.

DOI: 10.1002/jcc.24437

Introduction

High-throughput screening has taken center stage in guiding heterogeneous catalysis[1] and materials discovery,[2] where the order of crystal structure lattices restricts the possible number of structures that must be computationally enumerated. Molecular catalyst motifs remain an attractive target for computational screening efforts. Although some experimental[3,4] and computational[5–8] screening efforts have been carried out for inorganic complex discovery, robust, user-friendly tools for the rapid generation and assessment of inorganic complexes are not widely available.

To accelerate discovery in inorganic chemistry, both flexible generation of 3D structures and efficient first-principles calculations to evaluate properties are necessary. In organic chemistry, molecular screening benefits from tools that have been developed to store[9,10] and analyze[11–13] chemical structures and connectivity, facilitating rapid property evaluation based on connectivity[14] or generation of 3D structures from these descriptions.[15,16] Within the first-principles community, OpenBabel[17] and Avogadro[18] provide users access to these rapid structure generation tools in a user-friendly graphical user interface (GUI). In the solid state, the Atomic Simulation Environment[19] provides a useful Python interface to periodic electronic structure codes. However, such cheminformatics and heterogeneous catalysis tools are not straightforwardly extended to molecular inorganic chemistry. Instead, users have relied on searching commercial databases of experimentally characterized compounds. More recently,[20,21] fragment-based and rules-based structure generation tools have been developed for 3D coordinate generation, as long as free-to-use fragments are placed into a predefined database, but the seamless connection to first-principles modeling in an open source framework is not yet available.

In this article, we introduce an efficient, reliable approach to the high-throughput screening of inorganic complexes in our open source molSimplify toolkit. This streamlined procedure enables chemical discovery by automating molecular and intermolecular complex structure generation, job preparation and execution, and post-processing analysis to elucidate correlations of electronic or geometric descriptors with energetics. In the Code Overview section, we provide a description of the code layout and the detailed routines involved in (i) structure generation, (ii) simulation automation, and (iii) post-processing analysis. We then present Benchmarking results of our approach over a 190 molecule test set. Finally, we provide the Conclusions and outlook for our software toolkit.

Code Overview

The developed software utility (molSimplify) is an open source workflow that incorporates geometric manipulation routines necessary for the generation of transition metal complexes, automated setup and execution of electronic structure calculations, and post-processing data analysis. The software is designed for maximum flexibility, supporting a range of coordination numbers, metals, ligand identities, and ligating geometries. The code both builds coordination complexes starting from a single metal atom and functionalizes complex structures or generates supramolecular complexes. Although intended for inorganic chemistry, these tools are

E. I. Ioannidis, T. Z. H. Gani, H. J. Kulik
Department of Chemical Engineering, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139 E-mail: hjkulik@mit.edu

Contract grant sponsor: MIT energy initiative and a Reed Grant from the MIT Research Support Corporation; Contract grant sponsor: National Science Foundation; Contract grant numbers: ECCS-1449291 and ACI-1053575

straightforwardly applicable to a number of challenges ranging from studying non-covalent interactions in organic molecules[22] to building complex, selectively functionalized molecules such as dendritic polymers.[23] Our structure generation tools are designed to enable fast reaction mechanism screening by evaluating binding interactions and generating candidate reactive intermediates in catalysis. The code accelerates high-throughput chemical discovery through interaction with internal and external chemical databases to generate new candidates and with both internal and external postprocessing routines to aid data interpretation and trend elucidation. In addition to a commandline version, molSimplify is available through a GUI to make the code accessible to the broader scientific community.

Code Architecture

The molSimplify code is written in Python 2.7,[24] a modern high-level programming language that is widely used in the computational chemistry community. Python's object-oriented approach enables natural representation of components we manipulate in our framework including atoms molecules, molecular fragments, and supramolecular complexes. The molSimplify program makes use of the OpenBabel[25] toolbox via the pybel Python bindings[17] to convert between chemical formats and perform force field optimizations of generated structures. Portability of the code is maintained by including other commonly used Python modules such as NumPy in the installation. Users may run and install the molSimplify on any Linux or Linux-based (i.e., Mac OS X) platform where Python and OpenBabel are available.

The user interacts with molSimplify in one of three ways: at a commandline interface (CLI), with an input file, or through a GUI. The GUI has been developed using the Qt5 library with its corresponding Python bindings (PyQt5). The GUI enables at-a-glance molecule building in a manner intended to be intuitive for the broader scientific community. Thus, the code consists of two distinct parts: (i) the modules that build the GUI and interact with the user and (ii) the processing core that executes structure generation and processing features.

The framework is comprised of over 13,000 lines of code and includes three main modules (Fig. 1) described in greater detail in the rest of this article:

1. The main structure generation module builds and modifies inorganic complexes and supramolecular complexes. It also generates the appropriate input files and job scripts to run first-principles quantum chemistry calculations on the generated structures.
2. The database module interacts with external chemical databases for ligand search and retrieval to be fed back into the structure generation module.
3. The post-processing module contains routines that monitor progress of quantum chemistry calculations, analyze their outcomes, and post-process wavefunction and charge density data to elucidate structure-property correlations. This module also interacts with external codes for some electronic structure analysis.

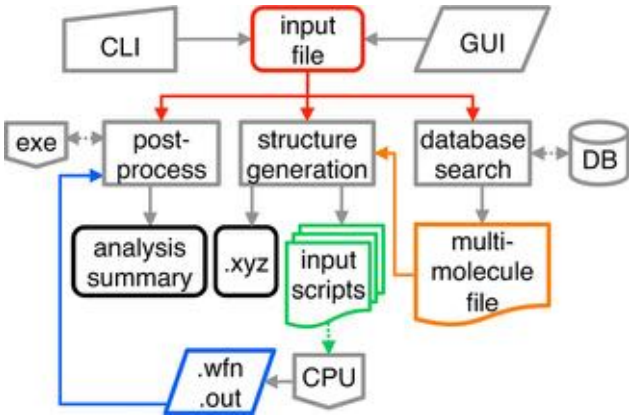

Figure 1. Flowchart for molSimplify code. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Although the modules are designed to work together, the user may prefer to use the code to only generate structures and use other freely available tools to prepare job files or post-process results.

We have represented all objects in the molSimplify code by classes ranging from the GUI class with subclasses for its various widgets to the atom3D and the mol3D class that represent atoms and molecules, respectively, in the structure generation module. These structure-oriented classes contain routines that perform geometric operations, manipulate the molecules by adding, deleting, or combining with other molecules, and calculate molecular or fragment properties. Example properties include the center of mass (COM) of a molecule, root-mean square deviation (RMSD) distance of one molecule with other molecules, and the number of hydrogen atoms connected to specific atoms.

The code stores both core (e.g., metal) and ligand structures in internal libraries that are prepopulated with some common metals and ligands. The GUI enables the user to add to these internal libraries and supports user inputs in the form of 2D representation Simplified Molecular Input Line Entry System (SMILES) strings[26] or 3D coordinates from .mol[27] or .xyz files. The code supports user-specified ligand taxonomy in the form of ligand groups and designations for whether a ligand only adds to organic cores or forms metal-ligand (ML) bonds, aiding subsequent database retrieval. To convert 2D SMILES representations to 3D coordinates, we combine OpenBabel[17,25] modules with our custom molecule class mol3D that carries out geometric manipulations relevant for inorganic complexes.

Structure Generation

The structure generation module is the central core of molSimplify and introduces tools with broad application in computational chemistry. This module synthesizes wide-ranging features including: simple transition-metal complex generation, custom molecule functionalization, functional group replacement, database search and retrieval for ligands in new molecule discovery, and supramolecular complex generation.

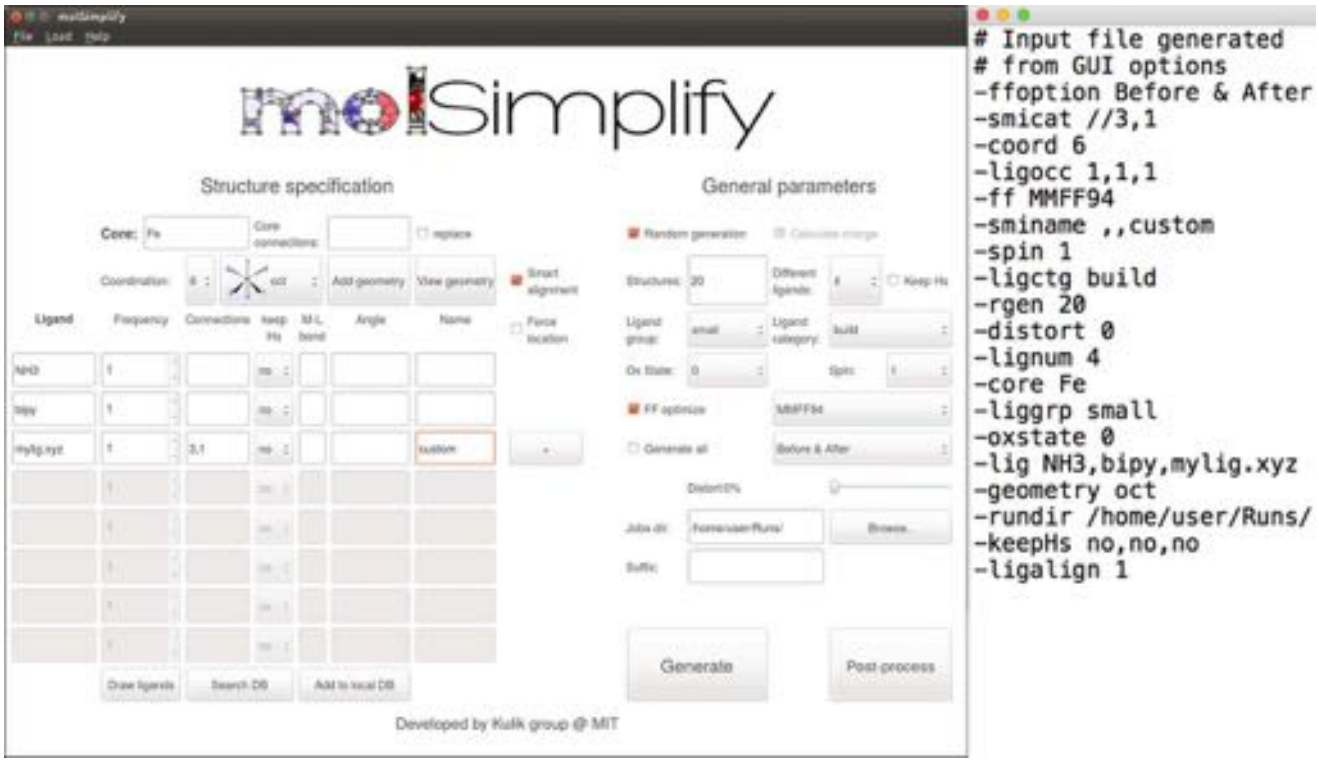

Figure 2. Example octahedral complex generation in GUI (left) or with input file (right). [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

General approach

In the general inorganic complex structure generation approach, the user specifies a (i) central metal, (ii) coordination number and geometry, (iii) a list of ligands to fill the coordination sphere, and (iv) the numbers and indices of ligand atoms that coordinate to the metal. Using this user input (example of input file and GUI input shown in Fig. 2), the code then generates 3D coordinates of the inorganic complex.

Coordination Templates. The code supports metal coordination numbers ranging from 1 to 8 with native support for the 13 most common coordination geometries (e.g., CN = 6 trigonal prism or octahedral). These geometries are encoded by a coordination template (CT) that is a set of points in space with the appropriate positioning and angles between a metal center and direct ligand atoms (e.g., 7 points in space for an octahedral complex CT). The code also supports custom molecular cores (see Custom Cores) by defining a backbone that includes all custom core (e.g., a porphyrin) atoms and the points in space for connecting atoms (CAs) (e.g., of proximal and distal ligands). The user may define custom CTs (e.g., CN > 8) by adding them to the coordination dictionary and providing a geometry of the CT. Once added, this new CT will be automatically incorporated into the GUI and CLI.

molSimplify flexibly modifies user-input structures, as outlined in the Custom Cores and Modify Function sections, and we recommend using these features rather than adding CTs unless the user has frequent need for an alternate CT. The default CTs correspond to the most symmetric forms for each CN and geometry, but the user may specify a distortion angle that measures the degree of displacement with respect to the original ML bond vector (BV) in spherical coordinates. For high-throughput screening, a randomized distortion feature may be selected to add increasing amounts of noise to the final geometry.

Denticity-Dependent Ligand Alignment. Strategies for ligand alignment to a CT differ based on the ligand's denticity. The code identifies ligand denticity by the number of CAs in the user-provided or internal library definition of the ligand. For instance, a carboxylate may be specified as either a monodentate or bidentate ligand, depending on the defined CAs (i.e, 1 vs. 2). The code sorts all ligands and prioritizes the ones with highest denticity. The larger number of constraints in aligning multidentate ligands as well as their bulky nature typically necessitates preferential alignment to minimize overall steric repulsion in the final complex. However, the user may choose separately to force molSimplify to place ligands in the coordination site specified and/or to override the order in which ligands are aligned (Fig. 2).

The multiple connection atoms in multidentate ligands simplify their alignment with respect to nonchelating ligands (Fig. 3). The code places the loaded ligand randomly with respect to the CT and translates the ligand to align the first CA to the target connection point (CP) on the CT. The code minimizes the distance of the second CA to the additional CT CP through rotations (Fig. 3). The rotation and translation matrices used in the code are described in the Supporting Information. The code determines the second CP from a look-up dictionary of denticity-

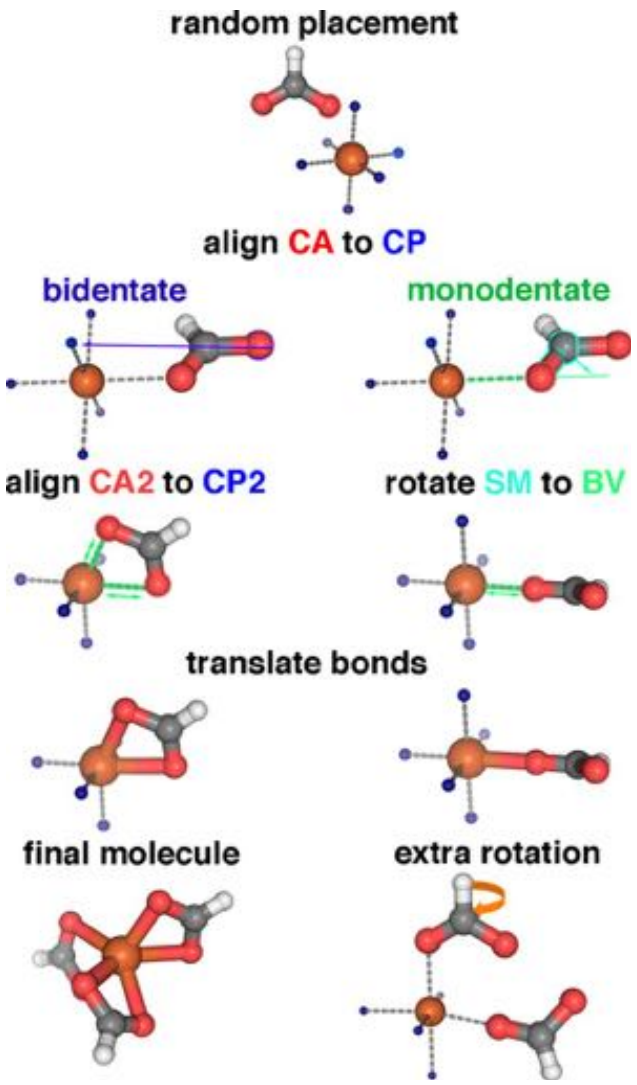

Figure 3. Alignment procedure for bidentate (left) and monodentate (right) ligands including alignment of CAs to CPs, rotating submolecules (SMs) around BVs, and extra rotations to minimize steric repulsion for monodentate ligands. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

dependent CP combinations on a CT. If alignment is poor between a bidentate ligand and the CT, the ligand angles are stretched by the code until the MMFF94 force-field evaluated energy is raised by no more than 5 kcal/mol per ligand. The final ligand structure is then aligned symmetrically. For higher-denticity ligands, remaining atoms are aligned through rotation and retrieval from the lookup table. Select tetradentate ligands (e.g., EDTA) are supported, but very high-denticity ligands may be better described as a custom core with functionalization points (see Custom Cores). As an example, metallocenes (see Fig. 6) may be generated by loading a custom core from the prepopulated internal database or by specifying two cyclopentadienyl ligands with a custom "CM" CA representing the center of the ring to the metal.

Monodentate alignment is slightly more challenging due to fewer constraints on ligand placement. The code identifies a free CP on the CT to align the CA. If not defined by the user or internal library, the first atom in the monodentate ligand is the default CA. As in the multidentate case, the code places the monodentate ligand randomly with respect to the CT and then translates the ligand to align the CA to an available CP (Fig. 3).

The code then defines a fragment of the ligand consisting only of the CA and ligand atoms directly bonded to the CA as a "sub-molecule." The alignment routine rotates the ligand until the sub-molecule's COM is aligned along the axis defined by the metal center and CP (Fig. 3). The sub-molecule COM alignment is necessary for bulky and asymmetric ligands. When the COM coincides with the CA, as in highly symmetric ligands, the code performs additional rotations to minimize any overlap between any atoms in the ligand and the metal core.

Next, the code rotates the ligand around the ML axis to minimize steric repulsion with adjacent ligands (Fig. 3) by simultaneously maximizing average distance between the ligands' centers of mass ($d_{\mathrm{com}_1-\mathrm{com}_2}$) and the minimum distance between any two atoms ($d_{12_{\mathrm{min}}}$). We use an optimization scheme that varies rotations to maximize the objective function:

$$F_{\mathrm{opt}} = d_{\mathrm{com}_1-\mathrm{com}_2} + \alpha \log d_{12_{\mathrm{min}}}, \quad (1)$$

where α was set to 10 based on trial and error.

ML Distances. The final step in alignment of ligands of any denticity is ligand translation to set the ML bond length. The ML distances are set based on an internal library of geometry-optimized bond lengths obtained with density functional theory (DFT). The chosen ML value is specific to the user-specified ML combination, charge state, and spin state. If the code cannot find an exact match for the ML distance in the library for the user-specified combination or a match to metal and direct ligand elements only, the code instead uses the sums of covalent radii of the two atoms. All presently available computed ML distances and their source are provided in the Supporting Information. The user may optionally specify custom values for the ML bond lengths or force use of covalent radii sums through the GUI or CLI. The user may use custom ML values for each ligand, for instance, to generate Jahn–Teller distorted octahedral complexes with longer axial bonds than equatorial bonds (M-L bond entry is shown in GUI in Fig. 2). Once all operations on the ligand are completed, the code updates the core structure by combining the previous core with the aligned ligand into a single molecule object.

Force Field Optimization. To produce an optimal starting structure for first-principles simulation, the molSimplify code carries out multiple partial force field optimizations during structure generation. As a ligand is generated from SMILES or loaded from 3D coordinates, the user may choose to preoptimize the individual ligand structure. We have found that this procedure is particularly useful for building very bulky ligands that are challenging for the OpenBabel 2D SMILES to 3D coordinate transformation. The code performs this and other optimizations by default with the MMFF94[28] force field, but we

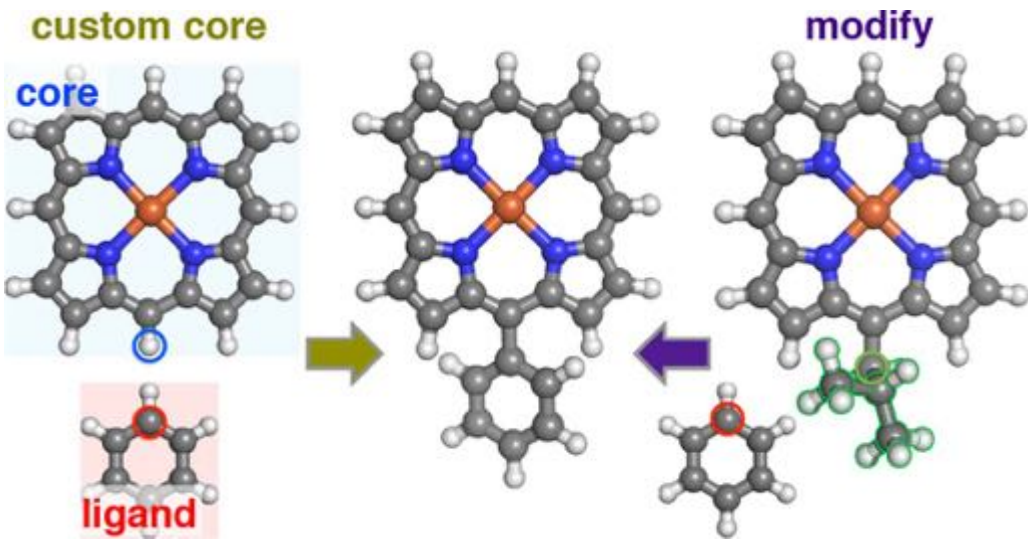

Figure 4. Phenyl porphine structure generation from custom core functionalization (left) of porphine by benzene or modification (right) of a functionalized porphine with benzene. Connecting atoms (red), modification point and downstream atoms (green), and connection point (blue) are highlighted. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

also support other force fields (e.g., the universal force field [UFF][29]) available via the OpenBabel module. Once the entire structure is built, the code will optionally optimize the entire structure. In this case, the core and all CAs are held fixed, and the force field optimization serves to further minimize steric repulsion between ligands. This divide-and-conquer approach for partial treatment with chemistry-specific DFT-trained values and selective force field optimization step is valuable for producing near-optimal starting geometries (see Benchmarking molSimplify).

Customized cores

Within molSimplify, the user may choose more complex initial structures around which new ligands or functional groups are attached, for example, in studying the effect of functionalizing a porphyrin. We refer to this feature as the "custom core" option in which the user may specify a multiatom structure obtained from 3D coordinates (.xyz or .mol) or a SMILES string. Such an approach is also useful in studying complexes with multiple metal centers. In the current version of molSimplify, the main distinction between functionalizing a standard or custom core is that in the latter case the structure may only be functionalized with singly coordinating ligands.

In this approach, the user specifies the custom core coordinates and the connection atoms in the structure that will be functionalized. If multiple functionalizations are desired at a single point in the structure, the connection atom should be listed each additional time. These connection atoms are the non-terminal atoms to which new functional groups should be added, and any existing excess hydrogen atoms on that connection atom in the custom core will be removed (Fig. 4).

Ligand alignment is carried out in a similar manner to the standard procedure except for an alternative definition of the target CP for each ligand. We determine the optimal position of the CA by randomly placing a dummy carbon atom near the specified custom core connection atom and minimizing steric repulsion from adjacent atoms through dummy atom rotation in spherical coordinates and force field optimization. This approach is customized for the case where the CA is a metal atom itself including both minimizing sterics and setting the M-L bond from the database. In both cases, the goal is to minimize overlap between the newly placed ligand and the original structure. Once this optimum CP for the CA is identified, the structure building proceeds as in the simple core case, including individual ligand optimization. We repeat this procedure for all ligands that must be added to the custom core and optionally carry out a final force field optimization of the completed structure to further minimize any ligand-ligand steric repulsion.

Modify function

The molSimplify code will also manipulate loaded custom core structures by replacing existing singly bonded functional groups in a feature we refer to as the "modify function." As with a custom core, the user loads the structure and then selects atoms on the molecule as modification points with the "replace" checkbox selected. This atom will be removed along with any downstream atoms connected to the modification point. In this process, the molecule is broken into submolecules, as determined by connectivity of the structure obtained from identifying bond distances smaller than 90% of the sums of covalent radii. The code identifies the submolecule to replace by determining the one that has a smaller number of atoms (Fig. 4). The new user-specified ligand is then aligned along the BV of the previously removed submolecule, and the new functional group to core bond distance is assigned first by sums of covalent radii, followed by standard rotations, and optional force field optimization to reduce steric repulsion. The user may screen initial guesses for transition states in catalysis by providing a custom core for

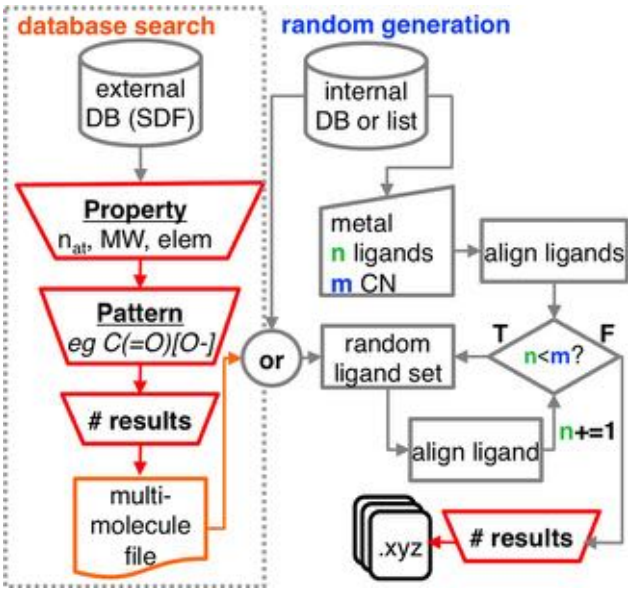

Figure 5. Chemical discovery workflow with optional constrained external database search (left) and constrained random structure generation (right). [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

functionalization or modification that is a good initial guess to the transition state.

Random generation

The constrained random generation module for building inorganic complexes is a unique feature of our framework. The user specifies at minimum the core of the complex and the coordination number but has the flexibility to further constrain the search. If the user specifies no ligand identities, ligands are extracted from the internal database of candidate ligands. One may specify a list of potential ligands, the ligand frequency, and to randomize only over a subset of the total ligands (Fig. 5). The user then specifies how many molecules are desired, and the code creates a randomized sample that falls inside the constraint space.

In addition to generating coordinates, the code creates an input file so that each randomly generated structure may be reproduced at any time. The flexibility of this approach enables both (i) randomized screening of ligands at a single coordination site (e.g., axial ligands in a porphyrin) and (ii) high-throughput generation and discovery of transition-metal complexes beyond the limits of human intuition.

Database search

The molSimplify framework optionally interacts with freely available multimillion molecule chemical databases.[30–36] To take advantage of this feature, the user must download the database of interest (e.g., ChEMBL[32] or eMolecules[35]) in a structure-data file[27] format. The code guides the user through the setup of these databases during installation, as outlined in the user guide. If the user downloads additional databases and places them in the database folder, the code automatically includes them in the database menu. This feature is in addition to the built-in internal database to which users may add their own custom molecules in SMILES, .xyz, or .mol format with the help of the code interface.

We designed the external database feature to enable users to screen a large ligand set with a given ML connectivity in common and survey indirect ligand effects, a frequent goal in catalyst and materials design. Here, we harness tools developed by the informatics community to search for and filter molecules from large databases. The module utilizes OpenBabel to support matching patterns in molecular structure using a SMILES string, the more flexible SMILES extension known as the SMILES arbitrary target specification (SMARTS),[14] or to a 3D structure. To aid structure generation, the user should specify a SMILES or SMARTS pattern (Fig. 5) and indicate the atom numbers in that pattern that are target CAs. To accelerate the search, we suggest generating a fast search file using OpenBabel before using a new database in molSimplify.

The structures matching the search are retrieved and stored in a multimolecule file with SMILES strings in the first column and the appropriate CAs that match the original pattern in the second column. For instance, if the user wishes to search for carboxylate (e.g., C(=O)[O-]) ligands to be aligned in a bidentate fashion to a metal core, the user would specify CAs as 2,3. If the search retrieves acetate (i.e., CC(=O)[O-]), then the code will report the CAs for that result as 3,4 in the multimolecule file. If the user does not specify CAs at runtime, the code defaults to reporting the CA corresponding to the first atom in the pattern. The CAs may be manually revised at any time.

The database feature supports constraints such as total number of desired results, range of numbers of atoms or molecular weight, number of bonds, or chemical properties encoded by flexible SMARTS strings. In addition to OpenBabel-driven filtering, our built-in routines filter out multifragment structures, salts, and duplicate structures. Once the multimolecule file is generated, the user may evaluate the search results and build molecules using the structure generation tool. The "draw ligands" feature generates a vector graphic (SVG) of all molecules in the test set with labeled atom indices to aid inspection of the data set and for any revising of connection or CA selections prior to high-throughput screening. During structure generation, the user specifies the multimolecule file as a ligand, and the CAs are read in from the file. The code loops through and generates an inorganic complex for each specified structure in the file. This approach is also compatible with constrained random generation and user-defined or internal-database structures for remaining coordinating ligands. Although intended for high-throughput screening in inorganic chemistry, the database search feature has broad applicability for sampling chemical space.

Supramolecular complex building

Non-covalent intermolecular interactions[22] are of central interest in many areas of chemistry and catalysis. To aid screening

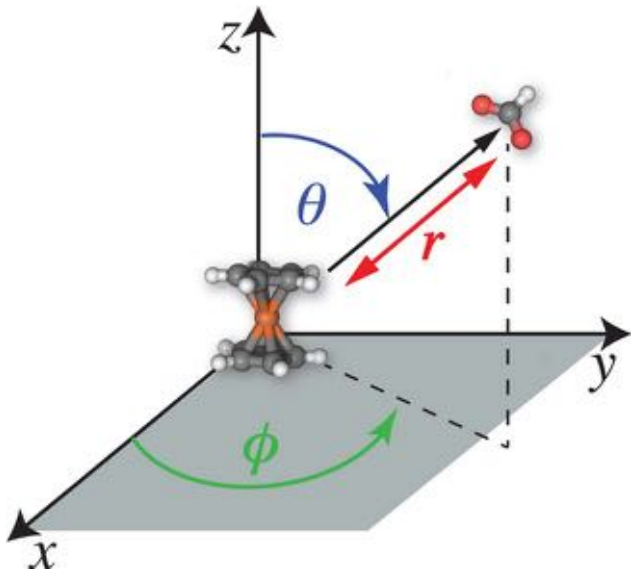

Figure 6. Supramolecular complex generation coordinates: distance (r) and angles (θ, ϕ) of the additional molecule COM with respect to the base molecule COM. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

of such interactions, the molSimplify code also generates supramolecular complexes. For this special case, the user specifies a base molecule, which may be simultaneously optionally functionalized. Then, the user selects the "extra molecule" check box and defines an additional molecule to be placed around the base structure. This additional molecule may be any structure that can be represented by a SMILES string or 3D coordinates, including another inorganic complex originally built by molSimplify. If the user does not specify any input, random values are selected in the code's spherical coordinate system centered at the COM of the base molecule for the radial distance (***r***), azimuthal angle (ϕ), and polar angle (θ) (Fig. 6). As with constrained random generation, input files are generated alongside each structure so the exact structure may be regenerated at any time.

Alternatively, the code will generate a user-specified number of structures under constraint of specified lower and upper distance and angle limits. This supramolecular generation consists of up to 200 randomized attempts to satisfy user constraints while simultaneously avoiding overlap between the two molecules. The distance specified here is added to the distance from the sums of covalent radii between base molecule atoms and additional ligand atoms. Covalent ligands should be generated using the standard approach. If the user wishes to obtain more directional initial configurations of the supramolecular complex, one may specify a subset of ligand atoms to be aligned to the core. Once the final positions are determined, the two molecules are combined into a single coordinate file. The user may optionally request that a line dividing the two molecules is inserted into the coordinates, as is customary in symmetry adapted perturbation theory or energy decomposition analysis calculations.

Simulation Automation

Once the coordinates of the structures are generated, the program places the .xyz files into separate folders that are named with a unique identifier corresponding to the metal, ligands, and coordination geometry of the complex. A common next step in a catalyst or materials discovery workflow is to calculate properties of generated structures with first-principles methods. Our code incorporates a file generator that assists in the construction of input files and scripts for common quantum chemistry codes and high-performance computing queuing systems, respectively.

Currently, molSimplify prepares input files for the quantum chemistry packages TeraChem,[37,38] GAMESS,[39] and Q-Chem.[40] The user specifies the charge, spin state, type of calculation, method, basis set and any additional input that would be required for the calculation. Importantly, the code "fact checks" some input choices to minimize user error. If the user-selected charge and spin states are incommensurate with the number of electrons in the associated structure, the code will present a warning and suggest an alternate charge assignment. Additionally, the code will suggest a total charge based on the charge state of selected ligands and a user-selected oxidation state that aids in input generation. Another example of this fact-checking is that when a user wishes to carry out a nonclosed shell calculation in TeraChem, the user should use an unrestricted method with optional level-shifting.[41] Therefore, when a user chooses a nonsinglet spin multiplicity, the input generation automatically defaults to enforcing these other options. The user may also enter a series of common input commands that is stored for future use by selecting the "make default" option.

The molSimplify code generates job scripts for submission to the SGE[42] or SLURM[43] queuing systems, which are used in a majority of supercomputers and local computer clusters. The user can specify job identifiers, designate a queue, run wall time, required memory, modules that are loaded by the environment modules utility, and more custom commands in a text editor field. If the user specifies nothing, the code generates a basic queue script with job run command. As run parameters vary by use case and cluster, the user is encouraged to use the "make default" feature to store commonly used settings.

The integrated creation of structures, input files, and job scripts promotes full automation of the generation and evaluation of properties of inorganic complexes, thus streamlining a procedure that would normally require significant time investment and human interaction. Furthermore, to facilitate sensitivity analysis of the results, the user can specify multiple values of parameters such as exchange-correlation functionals or spin states and the program will generate multiple input files each corresponding to one set of input parameters.

Structure-Property Correlation and Analysis

The primary objective of any high-throughput catalyst and materials design screen is to unearth correlations between

Table 1. Major properties reported or calculated with the molSimplify post-processing module.

Property	Software	Equation
Energy and job information	molSimplify	–
Hirshfeld metal charge	Multiwfn	$Q_M = Z_M - \int \frac{\rho_M(\boldsymbol{r})}{\rho_{\text{pro}}(\boldsymbol{r})} \rho_{\text{mol}}(\boldsymbol{r}) d\boldsymbol{r}$
VDD metal charge	Multiwfn	$Q_M = Z_M - \int_{\text{Voronoi cell}} [\rho(\boldsymbol{r}) - \rho_{\text{pro}}(\boldsymbol{r})] d\boldsymbol{r}$
Mulliken metal charge	Multiwfn	$Q_M = Z_M - \sum_{\mu \in M} Q_\mu$
Delocalization index of metal	Multiwfn	$\delta_M = \sum_B \sum_{k=\alpha,\beta} 2 \sum_{i \in k} \sum_{j \in k} \sqrt{\eta_i \eta_j} S_{ij}(M) S_{ij}(B)$
Localization index of metal	Multiwfn	$\lambda_M = Z_M - \delta_M$
d-band center	molSimplify	$\varepsilon_d = \sum_i^{MOs} \frac{\eta_i \varepsilon_i \sum_j^{d-AOs} (C_j^i)^2}{\sum_j^{d-AOs} (C_j^i)^2}$
HOMO/LUMO energy	molSimplify	$\varepsilon_{HOMO} = \max_{\text{occupied } MOs} \varepsilon_i$, $\varepsilon_{LUMO} = \min_{\text{unoccupied } MOs} \varepsilon_i$
Energy gap	molSimplify	$\varepsilon_{\text{gap}} = \varepsilon_{\text{LUMO}} - \varepsilon_{\text{HOMO}}$
Fermi energy	molSimplify	$\varepsilon_{fermi} = 0.5(\varepsilon_{\text{HOMO}} + \varepsilon_{\text{LUMO}})$
Average orbital occupation	molSimplify	$\eta_{av} = \frac{\sum_i^{\text{occupied } MOs} \eta_i}{\sum_i^{\text{occupied } MOs} 1}$
HELP	molSimplify	$\text{HELP} = \int_v \rho(\boldsymbol{r}) d\boldsymbol{r}$, if $\text{ELF}(\boldsymbol{r}) \geq 0.5$
Average electron distance	molSimplify	$R_{av} = \int_v \frac{\rho(\boldsymbol{r})}{n_e} r^{e-M} d^3\boldsymbol{r}$
Standard deviation of electron distance	molSimplify	$\sigma_{av} = \frac{1}{n_e} \sqrt{\int_v (\frac{\rho(\boldsymbol{r})}{n_e} r^{e-M} - R_{av})^2 d^3\boldsymbol{r}}$
Average ELF	molSimplify	$ELF_{av} = \int_v \frac{\rho(\boldsymbol{r})}{n_e} \text{ELF}(\boldsymbol{r}) d^3\boldsymbol{r}$
Standard deviation of ELF	molSimplify	$\sigma_{\text{ELF}} = \frac{1}{n_e} \sqrt{\int_v (\frac{\rho(\boldsymbol{r})}{n_e} \text{ELF}(\boldsymbol{r}) - \text{ELF}_{\text{av}})^2 d^3\boldsymbol{r}}$
NBO charge, occupations, d-band center	NBO	–

geometric or electronic structure and energetics. To realize this goal, molSimplify gathers output files, parses results, and carries out post-processing analyses (Table 1) intended to be general for a broad set of design challenges in inorganic chemistry. The code aids determination of structure-activity correlations using our developed built-in routines and leveraging external programs with established flexibility for wavefunction and electron density analysis.[44,45] The user is guided through the configuration of interfaces to external codes during molSimplify installation.

The program streamlines large data set collection and analysis by providing an on-the-fly summary that includes calculation descriptors such as the compound name, method used, spin, and charge alongside progress and results indicators including steps in a geometry optimization, convergence status, absolute energies, computing time, and $<S^2>$ values. These parsing routines are stored in the "postparse.py" script, and the user may modify this script to track other quantities for their specific application. All post-processing codes search recursively for .out and .molden files and, therefore, are independent of the user's directory structure.

External code post-processing analysis includes an interface to the natural bonding orbital (NBO)[44] v6.0 code for charge analysis and natural population analysis.[46] From the output of NBO calculations, molSimplify will summarize metal partial charge, average contribution of the metal to shared NBOs with other elements, and d-orbital occupation. We also interface with the Multiwfn[45] code to apply volume-oriented partitioning of the charge density, as exemplified by Bader's quantum theory of atoms in molecules (QTAIM)[47] basin and population analysis. This added feature supports the use of Molden[48] format wavefunction files typically generated by any quantum chemistry software. In addition to Bader's QTAIM charges, this interface extracts partial charges on the metal and direct ligand atoms with commonly used charge analysis schemes including Voronoi deformation density (VDD),[49] Hirshfeld,[50] and Mulliken charges (see Table 1). When using QTAIM theory as implemented in Multiwfn, molSimplify identifies attractors and reports the delocalization index (DI) between two atom-centered attractors as a quantitative measure of the number of electron pairs delocalized (or shared) between two atomic basins for all cases where one atom is a metal atom. Importantly, for many inorganic chemistry applications, multiple attractors may be identified for a single atom. The molSimplify code uses a geometric clustering approach to group attractors and provide summarized DI results on a per-atom basis.

Our built-in codes carry out density topology analysis on Gaussian format cube files (i.e., a 3D grid), which may be generated by Multiwfn from Molden format wavefunction files. By default, molSimplify generates the 3D grid for the total

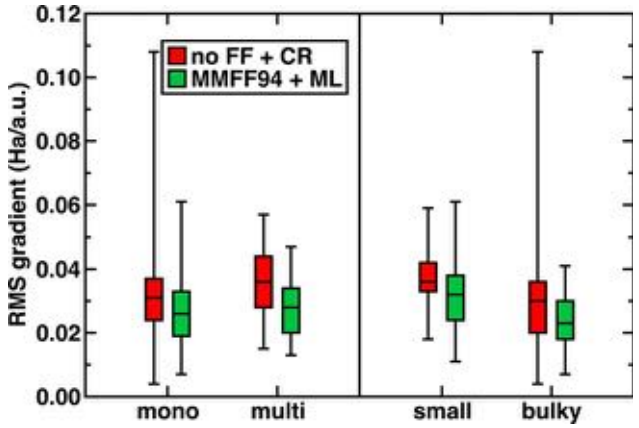

Figure 7. Comparison of RMS gradients with MMFF94 optimization and trained ML bond distances to no FF optimization and covalent radii (CR) bond distances for a 150 molecule semi-random test set grouped either by denticity (monodentate or multidentate) or by size (small or bulky). Whiskers represent the maximum and minimum of each data set. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

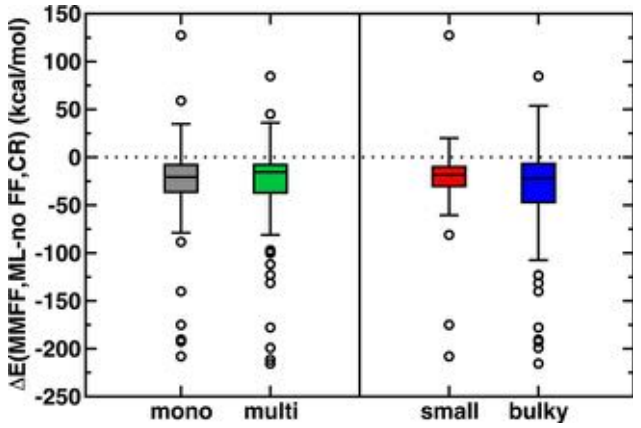

Figure 8. Energy differences with and without FF optimization and trained ML bond distances for the 150 molecule semi-random test set. Whiskers represent the +/− 1.5 interquartile region, and outliers are shown as circles. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

electron density, the spin density, the spin up and down electron densities, and the electron localization function (ELF).[51] Our built-in routines also calculate 3D charge density properties including the average distance and variance (i.e., spread) of the density from metal nuclei and the average value of the ELF. Following this analysis, the code retains the cube files for the user to further process or visualize.

The built-in routines also parse Molden wavefunction files to analyze and report information about molecular orbital character and occupancy, including the energy of the highest occupied molecular orbital (HOMO), the energy of the lowest unoccupied molecular orbital (LUMO), the Fermi energy, and the d-band center.[52] Additionally, the code summarizes occupancies of individual atomic orbital and total subshell contributions from s, p, and d atomic orbitals of the metal center. Extensions to characterize d-band width and shape as well as orbitals centered on other atoms or groups of atoms are straightforward to implement. Some users may prefer to use other post-processing tools, such as cclib[53] or the recently introduced ORBKIT,[54] which is a python toolkit for postprocessing electronic structure calculations. We plan to add broad integration for other postprocessing tools in the near future.

Benchmarking molSimplify

To evaluate the overall robustness of molSimplify as a tool for automating the generation of inorganic complexes, we generated nearly two hundred transition metal complexes spanning a range of ligands and metal centers. Overall, 150 molecules are octahedral Fe(II) complexes randomly generated from a wide pool of ligands of varying coordinating element (C, N, O, P, Cl) and bulk or denticity, whereas the remaining 40 are Cr, Mn, Fe, Co, and Ni complexes[55] that have a similarly diverse set of ligands with both organic and halide coordinating atoms in coordination environments ranging from tetrahedral to octahedral. Complete details and coordinates of the molecules included in both test sets are provided in the Supporting Information.

For each structure generated, we performed a single-point gradient calculation and computed the maximum and root mean squared (RMS) energy gradient, which serves as a metric of the quality of the generated structure by indicating the proximity of the structure to the closest stationary point in a DFT geometry optimization. All calculations presented here were carried out with the TeraChem[37] quantum chemistry package. The hybrid-exchange correlation functional B3LYP[56–58] was used, and in TeraChem the default definition uses the VWN1-RPA form for the LDA VWN[59] component of LYP[56] correlation. Heavy atoms (Fe) were treated with the LANL2DZ effective core potential basis, and the 6-31G* basis was used for the remaining atoms.

In the first 150-molecule randomized test set, we used molSimplify to generate each structure in four ways: (i) no force field optimization and the sum of covalent radii as ML distances, (ii) no force field optimization with trained ML distances from our database, (iii) MMFF94[28] force field optimization initially of both the individual ligands as well as the final structure with sum of covalent radii as ML distances, and (iv) MMFF94 force field optimization of both the individual ligands and the final structure with trained ML bond lengths from our database. To evaluate the accuracy of each method, we compare total energies and maximum and RMS gradients of each molecule. We broadly group the 150 molecules in two ways: (a) by denticity as monodentate (mono) and multidentate (multi) and (b) by size as small or bulky ligands. Comparison of structures obtained with Case i vs. iii or Case ii vs. Case iv reveals that force field optimization improves multidentate and bulky ligand placement but has no effect on smaller ligands, some of which are excluded from force field optimization due to poor bond distance estimation by MMFF94 (see Supporting Information). We thus limit our comparisons to the most divergent Cases, i and iv (Fig. 7).

Table 2. Comparison of UFF and molSimplify structure properties for a 40 molecule test set.

Metric	UFF	molSimplify MMFF94+ML
Lowest energy (#)	10/40	30/40
Median RMS grad. (Ha/a.u.)	0.032	0.019
Median max. grad. (Ha/a.u.)	0.054	0.033

Comparison of RMS gradients for Case i (no FF + CR in Fig. 7) and Case iv (MMFF94 + ML in Fig. 7) reveals that the median RMS gradient is systematically lowered for the 150-molecule set by 11–22%. The magnitude of reduction is largest for bulky or multidentate ligands, and comparison of maximum gradients (see Supporting Information) reveals that the bulky grouping benefits from an 18% reduction in the median maximum gradient when force field optimization and trained ML distances are used. As an example, the largest RMS gradient bulky structure without force field optimization is a tetrapyridyl complex that exhibits a much lower gradient when trained ML distances are used. The MMFF94 equilibrium bond lengths of the smaller methyl isocyanide ligand, conversely, are poor, leading to an increase in RMS gradient when the MMFF94 strategy is used. Overall, RMS and maximum gradients are modest for all molSimplify-generated structures, and the user will likely see the largest benefit from selective-force field optimization for bulky ligand alignment.

In addition to gradients, we compared the total energy of the structures generated in Case i and iv using the same denticity- and size-based groupings (Fig. 8). Both the median (~ -20 kcal/mol) and first through third quartile energy ranges (the box shapes in Fig. 8) systematically suggest favoring the MMFF94 + ML strategy, especially for monodentate and bulky ligands. In addition to the previously identified outliers, trained ML bond lengths are overestimates, for example, for an iron-quinoline complex ($\Delta E = 59$ kcal/mol), or force field optimization may generate sub-optimal ligand conformations, for example, for an ethylamine and Schiff base ligand ($\Delta E = 85$ kcal/mol). Conversely, many outliers have strongly improved energies when using the Case iv strategy, including the commonly used bipyridinyl ligand.

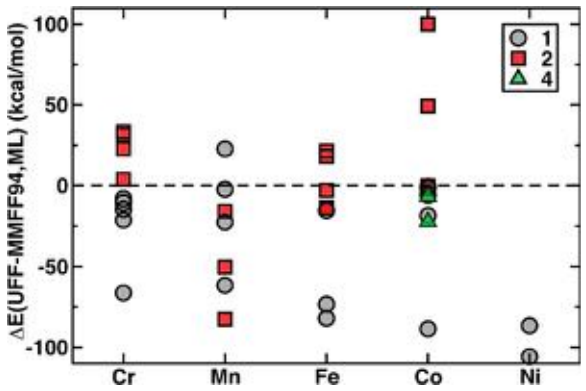

Figure 9. Energy difference between molSimplify- and UFF-generated structures in kcal/mol sorted by metal and maximum ligand denticity: monodentate (gray circles), bidentate (red squares), or multidentate (green triangles). [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

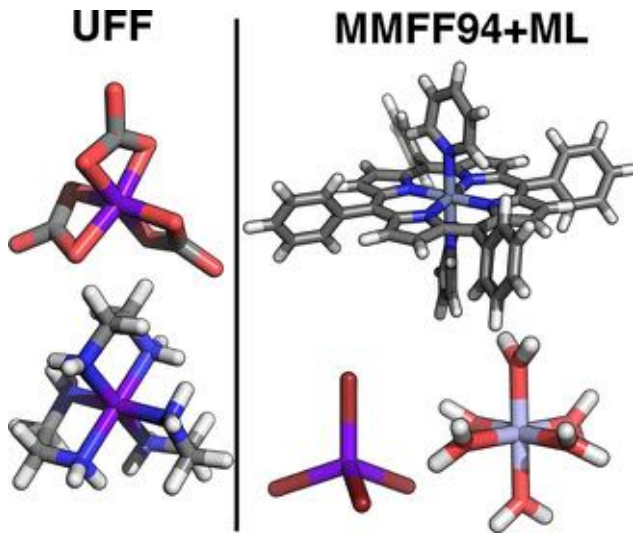

Figure 10. Representative complexes for which UFF (left) or molSimplify (MMFF94 and trained ML distances, right) generates a lower energy structure. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

We now focus on comparison of the most robust molSimplify strategy (MMFF94 partial optimization with trained ML bond lengths) to the UFF,[29] which has been developed for characterization of inorganic complexes. We note that molSimplify also supports UFF force field optimization, but we focus on comparison of our divide-and-conquer ML + MMFF94 strategy against an all UFF strategy. Other structure generation tools, such as OpenBabel that supports UFF, cannot enable an automated workflow of SMILES to UFF-optimized 3D structure because the user must still manually specify the existence and order of ML bonds. Here, we have manually generated the correct bond order prior to UFF optimization in OpenBabel as our control comparison.

For the 40 molecule test set, single points could not be converged for three molecules optimized with UFF, and we assume that they were higher in energy than the molSimplify counterparts that did converge. In total, molSimplify generates a lower energy structure for 75% of the molecules (Table 2). The median RMS gradient and maximum component of the gradient are both lowered by around 40%.

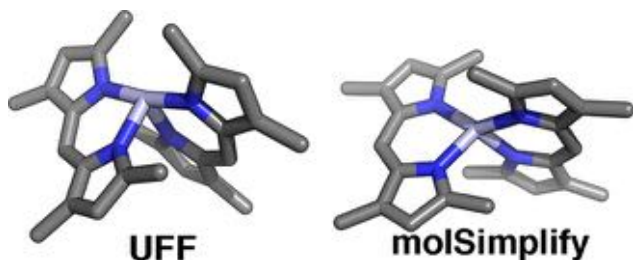

Figure 11. Comparison of UFF and molSimplify generated structures for distorted four-coordinate complex. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Table 3. Comparison of UFF and molSimplify structure properties for distorted four-coordinate Ni(II) complex.

Metric	UFF	molSimplify
Relative energy (kcal/mol)	+28.5	0.0
RMS grad. (Ha/a.u.)	0.031	0.029
max. grad. (Ha/a.u.)	0.094	0.078

Separating the 40-molecule test set by metal center and maximum denticity of the substituent ligands reveals (Fig. 9) that UFF generally outperforms molSimplify for select multidentate ligands. The molSimplify code aligns multidentate ligands by weakly stretching the ligand to increase agreement with the desired CPs. The UFF strategy, conversely, is more flexible for select multidentate alignments. Notably, however, the MMFF94 + ML molSimplify approach outperforms UFF by as much as 100 kcal/mol for relatively simple structures including a hexa-aqua Ni(II) complex, tetrabromo Co(II), and a tetraphenylporphyrin with axial pyridinyl ligands (Fig. 10).

The primary instances where UFF outperforms molSimplify by 50–100 kcal/mol margins are Co(III) complexes with bidentate carbonate or ethylenediamine ligands (Fig. 10). Here, we also note that the current database of ML trained bond lengths is limited to neutral ligands, and additional data for charged ligands, such as carbonate, will be added in the near future to improve molSimplify results further.

Lastly, we demonstrate the flexibility of molSimplify to generate complexes that have distorted coordination geometries. As an illustration, we consider a four-coordinate Ni(II) complex with two dipyrrole ligands that adopts a distorted tetrahedral geometry because the square planar form is sterically hindered. Standard implementations of UFF will only produce a see-saw geometry because the default Ni(II) atom type in UFF is octahedral. The molSimplify procedure, conversely, produces a more realistic distorted tetrahedral structure (Fig. 11).

Here, we specified tetrahedral coordination for the two ligands, and molSimplify places ligands by maintaining the user-specified coordination environment as well as any trained ML bond lengths and minimizing steric repulsion between ligands through partial MMFF94 force field optimization. Alternatively, the user may generate the same geometry starting from a square planar configuration and then specifying custom angles to distort the geometry. Single-point gradient calculations (Table 3) confirm that the molSimplify-generated structure is lower in energy and has smaller maximum and RMS gradients.

Conclusions

In this article, we have presented the molSimplify toolkit, which (i) automates the structure generation of inorganic complexes, (ii) prepares input files and job scripts for electronic structure property determination, and (iii) analyzes structure-property trends through post-processing analysis. Additional features in the structure generation module include database searching and randomized generation for chemical discovery, guided or random supramolecular complex building, and a feature to selectively alter molecular fragments.

We have benchmarked the program over 190 molecules that were both randomly selected and representative of common inorganic complexes. Our trained ML bond distances and selective force field optimization reduced gradients by around 20% compared to unoptimized placement. We confirmed this best use strategy by comparison to UFF optimization results where median molSimplify gradients were 40% lower as well. Overall, molSimplify provides a flexible but robust strategy to generate good starting structures for electronic structure characterization in high-throughput screening efforts.

We expect molSimplify to have wide applicability in fields ranging from bioinorganic chemistry to materials science and catalysis. The current version of the open source code is documented and published online (available at http://molsimplify.mit.edu). Ongoing effort is geared toward extending molSimplify for the generation and study of hybrid molecular-crystalline and other periodic systems.

Acknowledgments

H.J.K. holds a Career Award at the Scientific Interface from the Burroughs Wellcome Fund. This work was carried out in part using computational resources from the Extreme Science and Engineering Discovery Environment (XSEDE). The authors acknowledge helpful conversations with Adam H. Steeves.

Keywords: chemical discovery · structure generation · first-principles simulation · high-throughput screening · python

How to cite this article: E. I. Ioannidis, T. Z. H. Gani, H. J. Kulik. *J. Comput. Chem.* **2016**, *37*, 2106–2117. DOI: 10.1002/jcc.24437

Additional Supporting Information may be found in the online version of this article.

[1] J. K. Nørskov, T. Bligaard, *Angew. Chem. Int. Ed.* **2013**, *52*, 776.
[2] A. Jain, S. P. Ong, G. Hautier, W. Chen, W. D. Richards, S. Dacek, S. Cholia, D. Gunter, D. Skinner, G. Ceder, K. A. Persson, *APL Mater.* **2013**, *1*, 011002.
[3] J. Loch, R. Crabtree, *Pure Appl. Chem.* **2009**, *73*, 119.
[4] J. P. Stambuli, J. F. Hartwig, *Curr. Opin. Chem. Biol.* **2003**, *7*, 420.
[5] E. Burello, G. Rothenberg, *Int. J. Mol. Sci.* **2006**, *7*, 375.
[6] K. N. Houk, P. H. Y. Cheong, *Nature* **2008**, *455*, 309.
[7] Y. Chu, W. Heyndrickx, G. Occhipinti, V. R. Jensen, B. K. Alsberg, *J. Am. Chem. Soc.* **2012**, *134*, 8885.
[8] S. Keinan, M. J. Therien, D. N. Beratan, W. Yang, *J. Phys. Chem. A* **2008**, *112*, 12203.
[9] L. Mak, D. Marcus, A. Howlett, G. Yarova, G. Duchateau, W. Klaffke, A. Bender, R. C. Glen, *J. Cheminf.* **2015**, *7*, 31.
[10] L. Richter, G. F. Ecker, *Drug Discov. Today* **2015**, *14*, 37.
[11] S. Beisken, T. Meinl, B. Wiswedel, L. F. de Figueiredo, M. Berthold, C. Steinbeck, *BMC Bioinf.* **2013**, *14*, 257.
[12] M. J. Martinez, I. Ponzoni, M. F. Diaz, G. E. Vazquez, A. J. Soto, *J. Cheminf.* **2015**, *7*, 1.
[13] Y. Cao, A. Charisi, L. C. Cheng, T. Jiang, T. Girke, *Bioinformatics* **2008**, *24*, 1733.
[14] Daylight Chemical Information Systems Inc. Available at: http://www.daylight.com/products/toolkit.html, accessed on March 30, 2016.

[15] RDKit: Open-source cheminformatics. Available at: http://www.rdkit.org, accessed on March 30, 2016.
[16] C. Steinbeck, Y. Han, S. Kuhn, O. Horlacher, E. Luttmann, E. Willighagen, *J. Chem. Inf. Comput. Sci.* **2003**, *43*, 493.
[17] N. M. O'Boyle, C. Morley, G. R. Hutchison, *Chem. Cent. J.* **2008**, *2*, 1.
[18] M. D. Hanwell, D. E. Curtis, D. C. Lonie, T. Vandermeersch, E. Zurek, G. R. Hutchison, *J. Cheminf.* **2012**, *4*, 17.
[19] S. R. Bahn, K. W. Jacobsen, *Comput. Sci. Eng.* **2002**, *4*, 56.
[20] M. Foscato, V. Venkatraman, G. Occhipinti, B. K. Alsberg, V. R. Jensen, *J. Chem. Inf. Model.* **2014**, *54*, 1919.
[21] M. Foscato, G. Occhipinti, V. Venkatraman, B. K. Alsberg, V. R. Jensen, *J. Chem. Inf. Model.* **2014**, *54*, 767.
[22] K. Müller-Dethlefs, P. Hobza, *Chem. Rev.* **2000**, *100*, 143.
[23] E. R. Gillies, J. M. J. Fréchet, *Drug Discov. Today* **2005**, *10*, 35.
[24] Python Software Foundation. Python Language Reference. Available at: http://www.chemspider.com/, accessed on March 30, 2016.
[25] N. O'Boyle, M. Banck, C. James, C. Morley, T. Vandermeersch, G. Hutchison, *J. Cheminf.* **2011**, *3*, 33.
[26] D. Weininger, *J. Chem. Inf. Comput. Sci.* **1988**, *28*, 31.
[27] A. Dalby, J. G. Nourse, W. D. Hounshell, A. K. I. Gushurst, D. L. Grier, B. A. Leland, J. Laufer, *J. Chem. Inf. Comput. Sci.* **1992**, *32*, 244.
[28] T. A. Halgren, *J. Comput. Chem.* **1996**, *17*, 490.
[29] A. K. Rappe, C. J. Casewit, K. S. Colwell, W. A. Goddard, W. M. Skiff, *J. Am. Chem. Soc.* **1992**, *114*, 10024.
[30] X. Chen, M. Liu, M. K. Gilson, *Comb. Chem. High Throughput Screen* **2002**, *4*, 719.
[31] E. E. Bolton, Y. Wang, P. A. Thiessen, S. H. Bryant. PubChem: integrated platform of small molecules and biological activities. In: Wheeler, R.A. and Wang, D.C. (eds), Annual Reports in Computational Chemistry, Vol. 4. American Chemical Society, Washington, DC, **2008**, pp. 217–241.
[32] A. Gaulton, L. J. Bellis, A. P. Bento, J. Chambers, M. Davies, A. Hersey, Y. Light, S. McGlinchey, D. Michalovich, B. Al-Lazikani, J. P. Overington, *Nucleic Acids Res.* **2012**, *40*, D1100.
[33] J. J. Irwin, T. Sterling, M. M. Mysinger, E. S. Bolstad, R. G. Coleman, *J. Chem. Inf. Model.* **2012**, *52*, 1757.
[34] V. Law, C. Knox, Y. Djoumbou, T. Jewison, A. C. Guo, Y. Liu, A. Maciejewski, D. Arndt, M. Wilson, V. Neveu, A. Tang, G. Gabriel, C. Ly, S. Adamjee, Z. T. Dame, B. Han, Y. Zhou, D. S. Wishart, *Nucleic Acids Res.* **2014**, *42*, D1091.
[35] eMolecules database. Available at: http://www.emolecules.com/, accessed on March 30, 2016.
[36] ChemSpider, Search and Share Chemistry. Available at: http://www.chemspider.com/, accessed on March 30, 2016.
[37] I. S. Ufimtsev, T. J. Martinez, *J. Chem. Theory Comput.* **2009**, *5*, 2619.
[38] Petachem. Available at: http://www.petachem.com, accessed on March 30, 2016.
[39] M. W. Schmidt, K. K. Baldridge, J. A. Boatz, S. T. Elbert, M. S. Gordon, J. H. Jensen, S. Koseki, N. Matsunaga, K. A. Nguyen, S. Su, T. L. Windus, M. Dupuis, J. A. Montgomery, *J. Comput. Chem.* **1993**, *14*, 1347.
[40] Y. Shao, L. F. Molnar, Y. Jung, J. Kussmann, C. Ochsenfeld, S. T. Brown, A. T. B. Gilbert, L. V. Slipchenko, S. V. Levchenko, D. P. O'Neill, R. A. DiStasio, Jr., R. C. Lochan, T. Wang, G. J. O. Beran, N. A. Besley, J. M. Herbert, C. Yeh Lin, T. Van Voorhis, S. Hung Chien, A. Sodt, R. P. Steele, V. A. Rassolov, P. E. Maslen, P. P. Korambath, R. D. Adamson, B. Austin, J. Baker, E. F. C. Byrd, H. Dachsel, R. J. Doerksen, A. Dreuw, B. D. Dunietz, A. D. Dutoi, T. R. Furlani, S. R. Gwaltney, A. Heyden, S. Hirata, C. P. Hsu, G. Kedziora, R. Z. Khalliulin, P. Klunzinger, A. M. Lee, M. S. Lee, W. Liang, I. Lotan, N. Nair, B. Peters, E. I. Proynov, P. A. Pieniazek, Y. Min Rhee, J. Ritchie, E. Rosta, C. David Sherrill, A. C. Simmonett, J. E. Subotnik, H. Lee Woodcock Iii, W. Zhang, A. T. Bell, A. K. Chakraborty, D. M. Chipman, F. J. Keil, A. Warshel, W. J. Hehre, H. F. Schaefer Iii, J. Kong, A. I. Krylov, P. M. W. Gill, M. Head-Gordon, *Phys. Chem. Chem. Phys.* **2006**, *8*, 3172.
[41] V. Saunders, I. Hillier, *Int. J. Quantum Chem.* **1973**, *7*, 699.
[42] Grid Engine. Available at: http://gridscheduler.sourceforge.net/, accessed on March 30, 2016.
[43] A. B. Yoo, M. A. Jette, M. Grondona. D. Feitelson, L. Rudolph, U. Schwiegelshohn, Eds.; Job Scheduling Strategies for Parallel Processing9th International Workshop, JSSPP 2003, Springer, Seattle, WA, USA, **2003**; p. 44.
[44] E. D. Glendening, J. K. Badenhoop, A. E. Reed, J. E. Carpenter, J. A. Bohmann, C. M. Morales, C. R. Landis, F. Weinhold, In *NBO 6.0*; Theoretical Chemistry Institute, University of Wisconsin, Madison, **2013**.
[45] T. Lu, F. Chen, *J. Comput. Chem.* **2012**, *33*, 580.
[46] A. E. Reed, R. B. Weinstock, F. Weinhold, *J. Chem. Phys.* **1985**, 735, 735.
[47] R. F. W. Bader, *Chem. Rev.* **1991**, *91*, 893.
[48] G. Schaftenaar, J. H. Noordik, *J. Comput.-Aided Mol. Des.* **2000**, *14*, 123.
[49] C. Fonseca Guerra, J. W. Handgraaf, E. J. Baerends, F. M. Bickelhaupt, *J. Comput. Chem.* **2004**, *25*, 189.
[50] F. L. Hirshfeld, *Theor. Chim. Acta* **1977**, *44*, 129.
[51] A. D. Becke, K. E. Edgecombe, *J. Chem. Phys.* **1990**, *92*, 5397.
[52] B. Hammer, Y. Morikawa, J. K. Nørskov, *Phys. Rev. Lett.* **1996**, *76*, 2141.
[53] N. M. O'Boyle, A. L. Tenderholt, K. M. Langner, *J. Comput. Chem.* **2008**, *29*, 839.
[54] G. Hermann, V. Pohl, J. C. Tremblay, B. Paulus, H. C. Hege, A. Schild, *J. Comput. Chem.* **2016**, *37*, 1511.
[55] F. A. Cotton, G. Wilkinson. *Advanced inorganic chemistry*, 5th ed; Wiley: New York, **1988**.
[56] C. Lee, W. Yang, R. G. Parr, *Phys. Rev. B* **1988**, *37*, 785.
[57] A. D. Becke, *J. Chem. Phys.* **1993**, *98*, 5648.
[58] P. J. Stephens, F. J. Devlin, C. F. Chabalowski, M. J. Frisch, *J. Phys. Chem.* **1994**, *98*, 11623.
[59] S. H. Vosko, L. Wilk, M. Nusair, *Can. J. Phys.* **1980**, *58*, 1200.

Received: 26 April 2016
Revised: 24 May 2016
Accepted: 8 June 2016
Published online on 1 July 2016