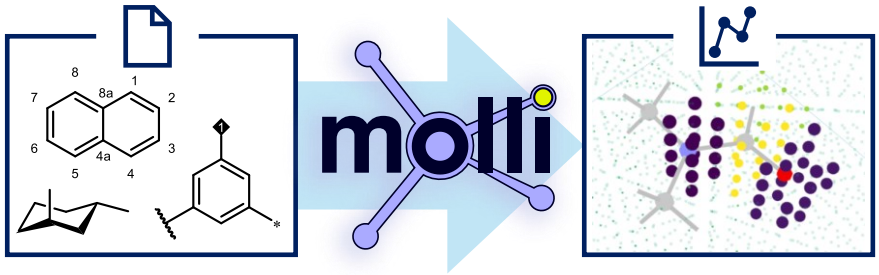
molli: A General-Purpose Python Toolkit for Combinatorial Small Molecule Library Generation, Manipulation, and Feature Extraction.

Alexander S. Shved,* Blake E. Ocampo, Elena S. Burlova, Casey L. Olen, N. Ian Rinehart, and Scott E. Denmark*

Department of Chemistry, University of Illinois at Urbana-Champaign, Urbana, Illinois 61801, United States

* shvedalx@illinois.edu, sdenmark@illinois.edu



ABSTRACT: The management and analysis of large *in silico* molecular libraries is pivotal in many areas of modern chemistry. The adoption and success of data-oriented approaches to chemical research is dependent on the ease of handling large collections of *in silico* molecular structures in a programmatic way. Herein, we introduce the MOLecular LIbrary toolkit, "molli", which is a Python 3 chemoinformatics module that provides a streamlined interface for manipulating large *in silico* libraries. Three-dimensional, combinatorial molecule libraries can be expanded directly from two-dimensional chemical structure fragments stored in CDXML files with high stereochemical fidelity. Geometry optimization, property calculation, and conformer generation are executed by interfacing with widely used computational chemistry programs such as OpenBabel, RDKit, ORCA, and xTB/CREST. Conformer-dependent grid-based feature calculators provide numerical representation suitable for diversity analysis, and interface to robust three-dimensional visualization tools provide comprehensive images to enhance human understanding of libraries with thousands of members. The package includes command-line interface in addition to Python classes to streamline frequently used workflows. This work describes the development and implementation of molli 1.0 and highlights the available functionality. Parallel performance is benchmarked on various hardware platforms and common workflows are demonstrated for different tasks ranging from optimized grid-based descriptor calculation on catalyst libraries to NMR prediction workflow from CDXML files.

KEYWORDS: cheminformatics, python, molecular formats, descriptors, parallel computations

1 INTRODUCTION

Modern synthetic chemistry is increasingly incorporating theoretical and empirical data-oriented approaches for designing functional small molecules, understanding reaction pathways, and predicting and optimizing reaction outcomes.[1–5] In recent years, medium- to high-throughput experimentation techniques have provided access to large data sets suitable for subsequent statistical analysis and predictive modeling.[6–10] Critically, encoding molecules in a machine-readable format is essential before any computational analysis of the physical molecular entities can commence.[11] Although great strides have been made in the high-throughput generation of *empirical* chemical data, suitably general tools for the high-throughput generation of *in silico* chemical data are lacking.

Representations of molecules with calculated features range from computationally simple to highly complex. In general, feature extraction from a molecule can be accomplished by considering, in order of increasing computational complexity: (1) only the atoms and bonds encoded in the molecular graph, (2) the three-dimensional (3D) shape, and (3) the full electronic structure of the molecule.[12] Molecular graph-based feature extraction methods such as fingerprinting[13] are fast but may lack 3D information that is critical for certain optimization problems. Indeed, the low-energy conformations of a molecule play an essential role in determining its chemical properties and recent interest in incorporating 3D information into molecular graph objects has led to a variety of feature extraction methods employing graph neural networks.[14–16] More challenges in representation arise when considering conformational flexibility, solvation, catalyst-substrate interaction and other molecular features that can only be described by full explicit 3D molecular encoding.

Our interest in molecular representation stems from our attempts at modelling quantitative structure-(enantio)selectivity relationships (QSSR) in enantioselective chemical reactions using chiral, small molecule catalysts.[17] Our group and others have designed a variety of alignment-dependent molecular interaction field (MIF) descriptors intending to capture the relevant features of a chiral catalyst that lead to high enantioselectivity.[18–20] A particular catalyst scaffold typically offers numerous options for analogue synthesis at well-defined positions on the structure and each analogue then has potentially many possible conformers. Therefore, our workflow required the ability to write custom code to manipulate large collections of 3D molecular structures and perform high-throughput computations on combinatorially constructed libraries of compounds.[21] In 2019, this laboratory disclosed the ccheminfolib toolkit,[18] an early iteration of a software package designed to handle combinatorial construction of large *in silico* libraries. One of the main

motivations for the creation of a new software package was *to establish a modern, convenient and extensible interface* that would allow rapid prototyping of chemical library-oriented workflows. Since the disclosure of ccheminfolib, we sought to address the following problems:

1. Generation of molecule and conformer libraries directly from ChemDraw™ .CDXML files with stereochemical fidelity.
2. Parallelization mechanisms and the capability for the parallel processing of chemical libraries with external computational software.
3. Address the performance issues in storage and retrieval of molecular entities from the disk, as well as calculating the grid-based descriptors,

As a result, we began the project to create the MOLecular LIbrary toolkit python 3 package we have dubbed "molli".

2 COMBINATORIAL LIBRARY GENERATION PIPELINE

2.1 CDXML File Parsing

Most computational workflows start with either 1D representations (SMILES) or 3D representations (.xyz or .mol files). We frequently faced challenges associated with the 1D representations. Axial and planar chirality cannot be encoded in SMILES strings and the stereochemical information is therefore lost upon the library generation. Although 3D structures are devoid of such limitations, they pose a considerable challenge to generate *en masse*. We believe that one of the most desirable ways to generate large libraries of 3D structures is by correctly interpreting their 2D chemical depictions. Existing CDXML conversion methods offer limited support for a number of desirable features such as atom labeling, stereochemical hint perceptions, isotopic notations, etc. (Figure 1A). We report our implementation of an improved parser in molli.

One of the important contributions to the parser was the realization of stereochemical hint perception. For all acyclic stereobonds[22] leading from an atom, the connected fragment (determined by the breadth-first graph traversal) was rotated by ±60° or ±90° depending on the number of adjacent atoms (See the Supporting Information p. S5 for more details). Endocyclic stereobonds are subjected to simple out-of-plane displacement of the participating atoms (Figure 1B). This way of interpreting the structures results in better starting geometries for subsequent minimization because of fewer atom overlaps and nudging of the *z*-coordinate toward the basin of geometric convergence.

This parsing reduced the number of unanticipated consequences, such as configurational inversion upon a forcefield minimization. It proved useful in the context of axial and planar chirality interpretation into 3D representation wherein no simple designators can typically be assigned and enforced by ChemDraw™ or related packages (Figure 1B). Parsing CDXML files to serialized objects can be executed directly from the command line with the `molli parse` command, or by using the `CDXMLFile` interface (Figure 1C).

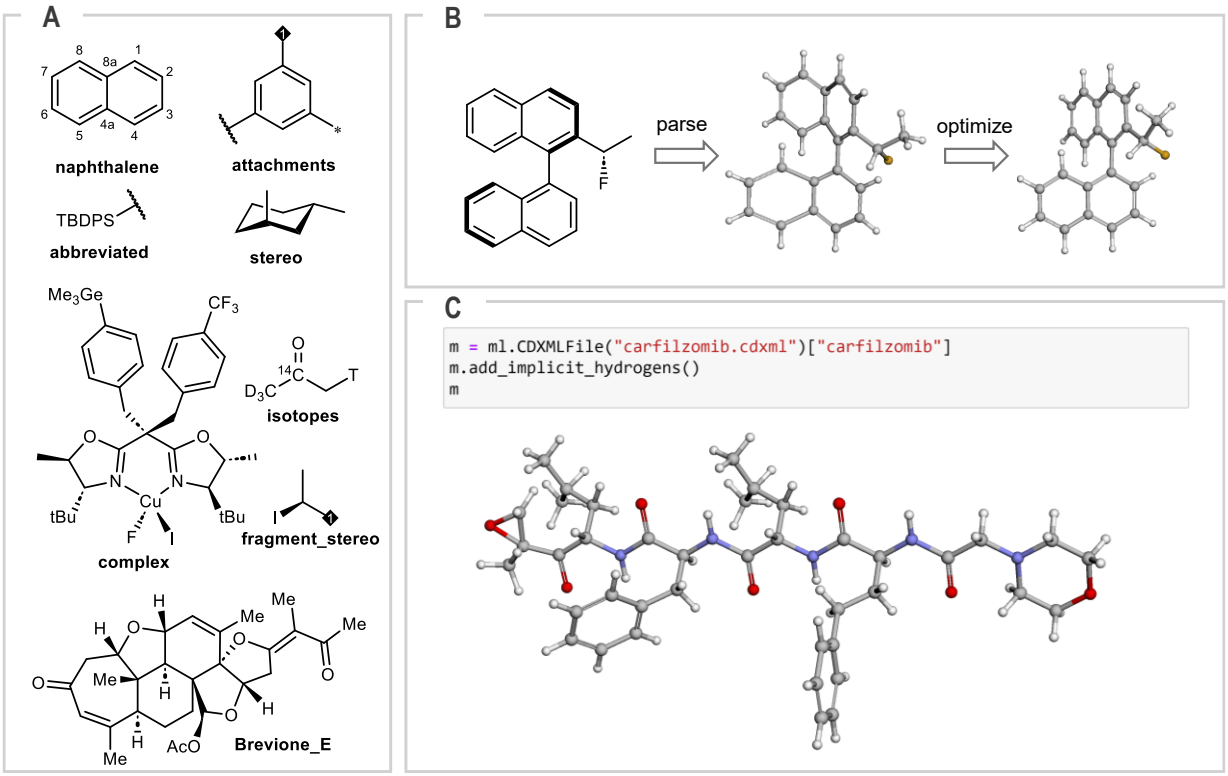


Figure 1. Molli CDXML parsing capabilities. (A) commonly recognized and parsed elements: atom labels, attachment points, abbreviations and stereobonds. The panel represents a valid input file for molli parsing. (B) Recognition of stereochemical hints by out-of-plane displacements and rotations for recognition of stereochemical information. (C) Jupyter notebook interface with inline molecule display.

2.2 Combinatorial Library Expansion from CDXML Files

Combinatorial library expansion can be performed programmatically in Python or directly from the command line with the `molli combine` command. Starting from CDXML files with the relevant fragment structures, labels, and attachment points denoted with native CDXML attachment point markup (Figure S2A, see the Supporting Information) molli joins the fragments on the basis of user-specified expansion rules (Figure S2B). Molli assigns new labels to the expanded combinatorial library members derived from the composite fragment labels and outputs a serializable `MoleculeLibrary` object (Figure S2B). We have previously reported the generation of a bis(oxazoline) (BOX) combinatorial library (Figure S2C) comprising a total 96,120

members, with 267 options for 4,4'-oxazoline substitution, nine options for 5,5'-oxazoline substitution including stereochemical analogues relative to the 4,4'-positions, and 40 options for substitution at the methylene group bridging the two oxazoline rings.[23] With the streamlined workflow described in Figure S2, we successfully obviated manual creation of the full expanded .CDXML file shown in Figure S2C.

2.3 Molecular Object Collections

Modern cheminformatics tools offer a multitude of ways of storing chemical information for singular molecules or small collections. We identified a need to access molecules or conformer ensembles from large collections without the necessity to create a full-fledged database. While this is possible to store many standard molecular files (.mol2, .sdf or .xyz), this leads to significantly inflated disk footprint, poor portability, and an additional requirement for parsing the human-readable chemical formats. To address this weakness, various solutions to compress multiple molecular files into commonly used archives, such as ZIP files, were explored. While this addresses the footprint and portability problems, it does not obviate the need for parsing of the textual information into the program data structures. An alternative solution to this problem was inspired by the structure of the GDBM (GNU Database Manager) database format.[24] The data is stored in a binary form where the offsets of the data keys and records are easy to calculate, allowing access to any data record in constant time. This structure is referred to as a uKV (micro key-value storage file). Molli implements this structure such as the maximum length of the key of 255 bytes, and the maximum length of the value of 4.29 GB. The default way of serializing molecular objects was chosen to be MessagePack[25] owing to its fast read/write performance. The ability to store molecules in a format that does not require and thus stores data in binary data structures gives the advantage of considerable space saving as well as a significant improvement in reading/writing performance (Table 1). Repeated atom and bond data storage for conformer ensembles is avoided and only the coordinates are stored in a contiguous float array.

Owing to significant improvements (Table 1) in read times (150×), storage size (6.5×), and random data access, the uKV file format is preferred and therefore molli features two dedicated `Collection` subclasses, `ConformerLibrary` and `MoleculeLibrary` that were made using uKV file as the default storage backend. Although compressed ZIP files represent a viable alternative in terms of size, the reading/writing speeds are significantly inferior. This process reads 1300 conformer ensembles per second on average, which effectively eliminates the input/output bottleneck for most applications.

Table 1. Comparison of Data Storage and Reading Efficiency.[a]

Backend	Size [MB]	Read time (1 core) [s]	Write time (1 core) [s]
Directory[.mol2]	924.9	101.5	89.5
Zip[.mol2] (uncompressed)	925.0	100.6	89.5
Zip[.mol2] (compressed)	126.9	145.7	n/a
uKV-file (.clib)	139.2	0.6	0.4

[a] Obtained with the BPA dataset. Compressed ZIPFile was obtained with Deflate level 5 algorithm. Reported timings are average of 3 repetitions. Read time represents the time to construct the object from its serialized version. Tests performed on System 3 (see the SI for details).

To demonstrate the broader implications of the proposed molecular storage, an example that is relevant to storing conformers for medicinally relevant molecules is provided. The data from the MoleculeNet[26] subset of the GEOM[27] dataset was reimported as a molli .uKV file (see the Supporting Information) The chosen storage format was, once again, efficient and user-friendly. A 2.1 GB compressed .tar.gz archive was seamlessly converted into a 2.8 GB uKV file (1.8 times smaller than the uncompressed pickle files and properties stored in a separate file) but more importantly featuring the data *annotations directly embedded* as attributes in the `ConformerEnsemble` instances.

3 PARALLEL CALCULATION PIPELINE

In a typical workflow, tasks such as geometry optimizations, conformer generations, and property calculations are done in parallel. Typically, these calculations are carried out with external software[28] by a unified process in which: (1) a set of input files is prepared, (2) a worker process receives said input files and shell commands to execute, (3) the commands are run, and the output is captured, and (4) the necessary files are subsequently transferred to permanent storage and are analyzed. Molli implements a parallel job pipeline that allows computation of molecular properties with external software such as XTB, CREST, NWChem and ORCA, and it can be easily extended to any other package (see Supporting Information section 6.1 for more details). Here, we illustrate two workflows to demonstrate the flexibility that molli API may offer.

3.1 KRAS inhibitor rotational barrier estimation.

Hindered rotation around single bonds, which results in axial chirality, is an important motif in catalysts and pharmaceuticals.[29,30] The barrier height may not always be straightforward to estimate experimentally and doing so in a high throughput sense with minimal involvement may significantly facilitate pre-screening of synthetic candidates before their experimental evaluation. Herein we demonstrate how this workflow could be setup with the aid of molli parsing and

combinatorial creation. The workflow started with the CDXML file which was deliberately constructed to mimic the original figure[31] as closely as possible (Figure 2). Parsing the CDXML files with the help of molli results in the `MoleculeCollection` files that were subsequently subjected to the combinatorial expansion protocol. Coarse structure minimization with MMFF94, as implemented in OpenBabel 3.1.0 yielded the initial guess structures. An XTB relaxed surface scan was then used to scan the potential energy surface with respect to the rotation around the C–N bond by constraining the appropriate dihedral atoms. It was crucial that the implementation of parsing in molli allowed the labelling of the corresponding atoms in the drawing that consequently enables facile input file generation. Analysis of the relaxed surface at the GFN2 method allowed the identification of good guess structures for the rotational transition states. An ORCA transition state search was partially successful; out of nine transition states, it was able to locate six of them correctly. The remaining three structures could be assembled in a more streamlined fashion; the core of successfully identified transition state was dissected along the C–N bond and the substituent was then replaced with the desired ones. Simple rotation to constrain the dihedral angle allowed the generation of more reasonable guess structures. The computed barriers closely matched the experimentally observed ones (Table 3).

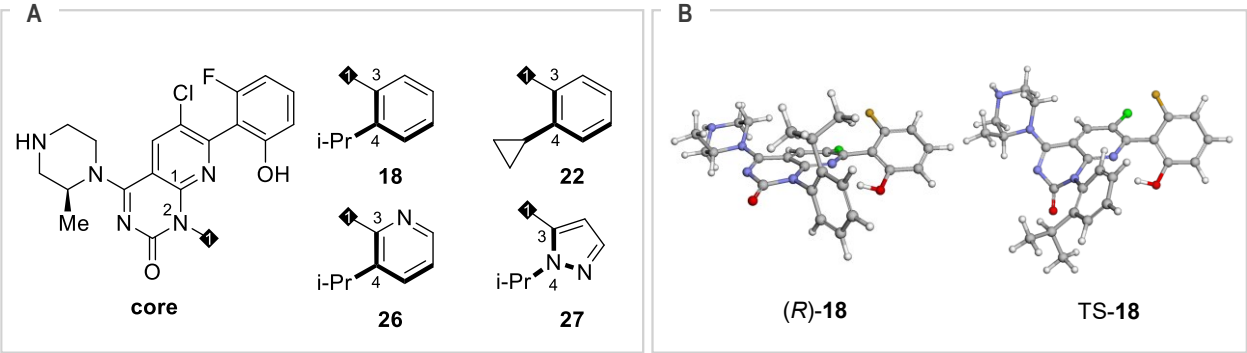


Figure 2. KRAS inhibitor rotational barrier estimation workflow. (A) Fragment of CDXML file that was used for parsing and library assembly. For a full list of structures see Supporting Information, section 6.3. (B) Obtained representative equilibrium geometries of *R*-isomers and transition states. Of note is the remarkable distortion of the 2-pyrimidinone ring from planarity in the transition state owing to severe strain.

Table 3. Summary of predicted vs. observed racemization barriers at B97-3c level of theory (in kJ mol^{-1}). For a full list of structures see Supporting Information, section 6.3.

Compound	Exp.	Pred.
18[*]	108.8	108.3
22	104.6	103.2
23	>125.5	141.0
24	>125.5	150.0
25	121.3	146.1
26	98.3	92.6
27	90.0	73.7
28	73.2	69.9
29	107.9	101.3

3.2 GIAO-DFT NMR prediction workflow.

Prediction of NMR spectra, particularly ^{13}C NMR spectra is a common task encountered in structural elucidation and revision. Although modern computational tools allow fast GIAO-DFT NMR prediction, a complete cycle workflow that automates the task to start with a ChemDraw™ file and orchestrates the required computations, is not generally available using open-source tools. A major advance towards this goal is the CENSO program that enables this workflow starting from the 3D ensemble representations.[32]

The workflow starts with parsing the 3D structures from the .CDXML file to yield a molecule collection (Figure 3A). Basic minimization with the MMFF94[33] force field as implemented in OpenBabel followed by conformer generation with CREST v4 workflow[34] created the desired conformer ensembles. These ensembles were subjected to geometry evaluation with the B97-3c method as implemented in ORCA. Upon conformer generation, the NMR isotropic shieldings were calculated with PBE0 / pcSseg-2[35] + CPCM(chloroform).[36] Molli features simple syntax that is used to compute the NMR properties (Figure 3B). Molli implements a parser of output files, which was used to scrape thermochemical and magnetic properties and stores them within the molecule objects. Boltzmann weights were computed, and the resulting weighed averaged NMR chemical shifts were subsequently compared to the experimental data showing close correspondence (Tables S5-S12). We observe average errors in range [1.2, 2.0] ppm with maximum errors in range [3.1, 4.0] ppm, consistent with the general expectations of DFT prediction methods.

[*] The compound labels throughout the manuscript were chosen to be non-standard on purpose. This is to demonstrate that the source. CDXML files can be constructed with the compounds labeled arbitrarily. We chose to label ours the way they were labeled in the original publications.

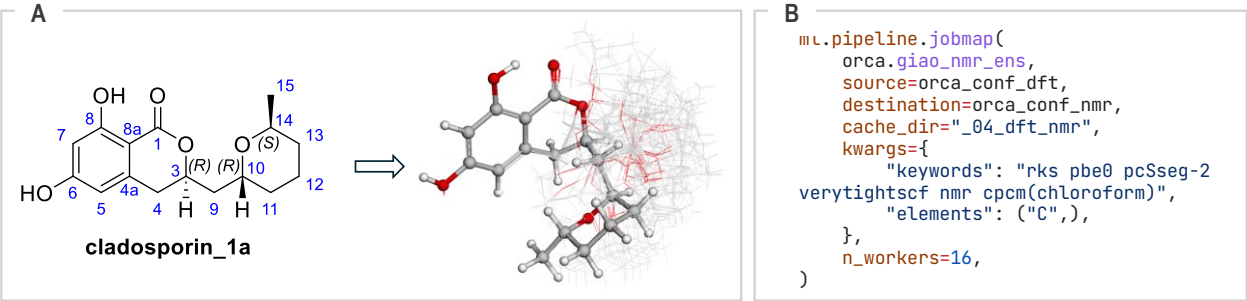


Figure 3. (A) CDXML parsing and conformer generation workflow results for cladosporin. (B) minimal code example for GIAO NMR predictions.

4 GRID-BASED DESCRIPTOR OPTIMIZATION

A particular kind of the MIFs that is found use in our laboratory, is the grid-based conformer-averaged indicator field (GBCA) descriptors, such aas the average steric occupancy descriptor (ASO) and average electronic indicator field (AEIF). A naïve implementation of the GBCA descriptors suffers from significant, unfavorable scaling dependencies with respect to the grid size. This step was very computationally expensive to carry out on libraries of tens of thousands of molecules, requiring high performance computational hardware. To eliminate the slow process of descriptor computation, we decided to perform an optimization of this process. Molli employs two levels of optimization of the computing process. The optimization of the GBCA descriptors began by outsourcing numerically intensive arrayed calculations to a more efficient C implementation of the numpy package (Table 2). A 25-40-fold acceleration was observed; however the processing time was still high for a large library. We employed an auxiliary C++ sublibrary (called `molli_xt`) that was created through the use of pybind11.[37] Two functions were implemented that reproduced the behavior of SciPy's[38] cdist function that computes the distance matrix (and an analogous function was made that would compute a higher dimensional analog of the distance tensor). These functions calculated large arrays of distances between grid points and corresponding atomic positions with ~10-50% acceleration on the arrays of relevant size as compared to SciPy implementation. Up to two-fold acceleration was achieved when the computation was restricted to single precision floats that was sufficiently accurate for GBCA calculations. The final aspect of optimization came from the efficient partitioning of the grid points into proximal and distal prior to the calculation. To enable this process, the *k*-d tree[39,40] data structure was used to optimize the problem of finding the closest atoms to given grid points, as well as eliminating remote grid points that fall far outside the van der Waals surface of the molecule. We are delighted to report that

overall we were able to achieve a 1,700× acceleration of the process compared to a naïve python implementation, and a 50× acceleration as compared to naïve numpy approach.

Table 2: Benchmarking Results of GBCA Descriptor Calculation.[a]

Grid point spacing, Å	**1.5**	**1.0**	**0.7**
Number of grid points	3510	11362	32832
Descriptor vector sparsity (mean ± stdev)	92.0±4.4%	91.6±4.6%	91.5±4.7%
Pruned grid sparsity (mean ± stdev)	86.7±6.5%	86.0±6.7%	85.9±6.8%
Naïve python ASO, s	175.4	580.8	1686.5
Naïve numpy ASO, s	5.0	14.3	67.1
Scipy cdist optimized ASO, s	0.8	2.6	7.3
molli cdist ASO, s	0.5	1.8	4.9
KDTree & molli cdist optimized ASO, s	0.1	0.5	1.2

[a] Timings are reported on the BPA catalyst **65_vi** (88 atoms, 215 conformers). Benchmarks reported on System 3 (see the SI for details)

With the optimized GBCA calculation protocol in hand, the benchmark calculations were performed on the full BPA dataset[19] consisting of 806 entries and a total of 99,680 conformers, as well as on BOX dataset[23] consisting of 72,542 entries and 4,662,551 conformers. The calculations on the BPA dataset could be performed on a laptop computer (system 3) within two minutes. Computing of the BOX dataset under identical conditions took ca 1.5 h, which could be sped up considerably by employing more parallel processes on a workstation. Using a 64-core computation, ASO computation for the BOX dataset was complete under five minutes. This result represents a marked enhancement in speed and enables the calculation of descriptors with chemical resolution (0.75 Å spacing or below).

4.1 Molecule, Ensemble and Descriptor Visualization

By virtue of being a pure Python library, molli can be easily interfaced with a few different visualization libraries. Molli uses two different engines for visualization purposes: 3DMol.js[41] is used for simpler molecular renderings inside Jupyter notebooks (see Figures 1 and 2 for examples). This implementation allows a very simple in-place visualization that helps the end user understand the contents of their molecular or conformer libraries much better without the need to transfer the data to a third-party program for rendering (Figure 2).

Highly dimensional grid-based descriptors are particularly hard to interpret by a chemist without relying on the visual representation. To enable the visualization of these descriptors, as well as to enable their chemical interpretation, we employ another visualization capability using the pyvista package, which is a convenient set of wrapping functions over the VTK (Visualization ToolKit) package.[42,43] This engine can be employed for molecular rendering and it performs

particularly well for visualizing high-dimensional, grid-based descriptors in context of conformer ensembles. Figure 4 illustrates the directions of the maximal variance in the ASO and AEIF descriptors, corresponding to the locations of largest steric and charge distribution diversity in the BPA catalyst library (see also Figures S4-S18).

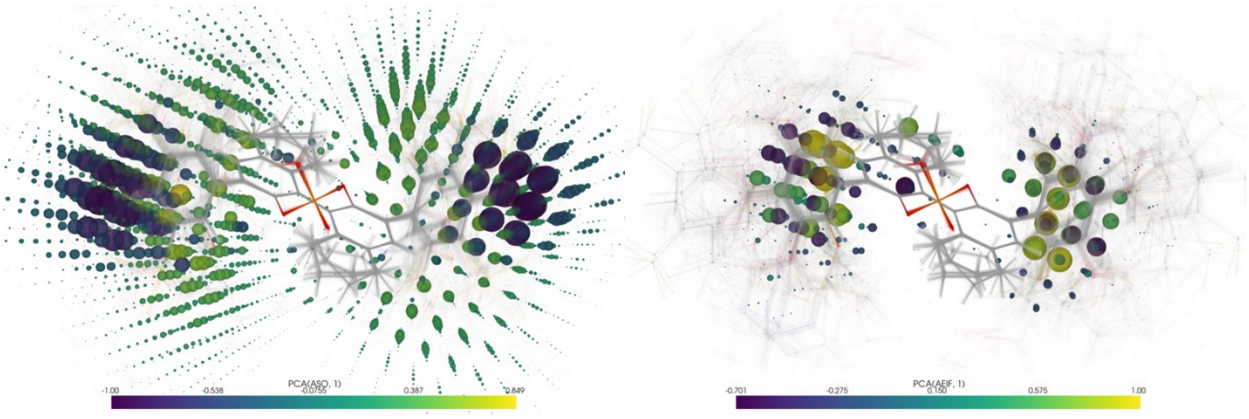


Figure 4. Normalized PCA1 loadings of ASO (left) and AEIF (right) descriptors of the BPA dataset overlayed with the conformer ensemble visualization. A 1.0 Å spacing grid was chosen for the visualization.

5 CONCLUSIONS

Molli comprises a powerful chemoinformatics toolkit that specializes in the creation of large combinatorial libraries of small molecules and parallel computations. A pure pythonic interface enables a seamless transition between a plain chemical drawing to a large *in silico* molecular dataset with preservation of stereochemical integrity. Combinatorial library creation can be performed with ease through both the command line interface as well as by writing custom scripts. Optimized GBCA descriptor calculations one can now easily reproduce the existing ASO and AEIF calculations as well as visualize their corresponding results. Lastly, one can employ the parallelized computational pipeline to compute the properties of isolated molecules and their conformer ensembles with arbitrary external software, of which we provide examples of workflows for XTB, CREST, ORCA and NWChem.

6 ACKNOWLEDGMENTS

We are grateful to the National Science Foundation for financial support of the Molecule Maker Laboratory Institute (NSF CHE 2019897) as well as from NSF CHE 2154237. We thank the W. M. Keck Foundation for the funds that enabled the purchase of the Odyssey HPC cluster. We thank Sara Lambert and Matthew Berry (UIUC NCSA) for their assistance with the GitHub

workflow and insightful discussions. The authors acknowledge Austin Douglas for assistance with establishing the documentation system and Ethan G. M. Mattson for prototyping some of the 240+conformer generation code. We thank Mark Hewitt for his assistance with the cluster computing resources. Finally, we thank Dr. Jeremy Henle and Dr. Andrew Zahrt for the design of ccheminfolib library that inspired the creation of molli.

7 SUPPLEMENTARY INFORMATION

Source code for the project can be found at https://github.com/SEDenmarkLab/molli. The project is available for quick installation Python package index and conda channels. Up-to-date documentation detailing the installation procedure and package usage examples can be found on the documentation portal, https://molli.readthedocs.io). Description of the hardware, additional information about implementation details, as well as the results from the computational pipeline workflows can be found in the attached pdf file. Datasets and the code for workflows discussed in the present manuscript can be downloaded from the Zenodo repository (https://zenodo.org/records/10719791, doi 10.5281/zenodo.10719790)

8 REFERENCES

(1) Taylor, C. J.; Pomberger, A.; Felton, K. C.; Grainger, R.; Barecka, M.; Chamberlain, T. W.; Bourne, R. A.; Johnson, C. N.; Lapkin, A. A. A Brief Introduction to Chemical Reaction Optimization. *Chem. Rev.* **2023**, *123* (6), 3089–3126. https://doi.org/10.1021/acs.chemrev.2c00798.

(2) Wang, H.; Fu, T.; Du, Y.; Gao, W.; Huang, K.; Liu, Z.; Chandak, P.; Liu, S.; Van Katwyk, P.; Deac, A.; Anandkumar, A.; Bergen, K.; Gomes, C. P.; Ho, S.; Kohli, P.; Lasenby, J.; Leskovec, J.; Liu, T.-Y.; Manrai, A.; Marks, D.; Ramsundar, B.; Song, L.; Sun, J.; Tang, J.; Veličković, P.; Welling, M.; Zhang, L.; Coley, C. W.; Bengio, Y.; Zitnik, M. Scientific Discovery in the Age of Artificial Intelligence. *Nature* **2023**, *620* (7972), 47–60. https://doi.org/10.1038/s41586-023-06221-2.

(3) W. Coley, C.; Jin, W.; Rogers, L.; F. Jamison, T.; S. Jaakkola, T.; H. Green, W.; Barzilay, R.; F. Jensen, K. A Graph-Convolutional Neural Network Model for the Prediction of Chemical Reactivity. *Chem. Sci.* **2019**, *10* (2), 370–377. https://doi.org/10.1039/C8SC04228D.

(4) Gómez-Bombarelli, R.; Wei, J. N.; Duvenaud, D.; Hernández-Lobato, J. M.; Sánchez-Lengeling, B.; Sheberla, D.; Aguilera-Iparraguirre, J.; Hirzel, T. D.; Adams, R. P.; Aspuru-Guzik, A. Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules. *ACS Cent. Sci.* **2018**, *4* (2), 268–276. https://doi.org/10.1021/acscentsci.7b00572.

(5) Gao, H.; Struble, T. J.; Coley, C. W.; Wang, Y.; Green, W. H.; Jensen, K. F. Using Machine Learning To Predict Suitable Conditions for Organic Reactions. *ACS Cent. Sci.* **2018**, *4* (11), 1465–1476. https://doi.org/10.1021/acscentsci.8b00357.

(6) Mennen, S. M.; Alhambra, C.; Allen, C. L.; Barberis, M.; Berritt, S.; Brandt, T. A.; Campbell, A. D.; Castañón, J.; Cherney, A. H.; Christensen, M.; Damon, D. B.; Diego, J. E. D.; García-Cerrada, S.; García-Losada, P.; Haro, R.; Janey, J.; Leitch, D. C.; Li, L.; Liu, F.; Lobben, P. C.; Macmillan, D. W. C.; Magano, J.; McInturff, E.; Monfette, S.; Post, R. J.; Schultz, D.; Sitter, B. J.; Stevens, J. M.; Strambeanu, I. I.; Twilton, J.; Wang, K.; Zajac, M. A. The Evolution of High-Throughput Experimentation in Pharmaceutical Development and Perspectives on the Future. *Organic Process Research and Development* **2019**, *23* (6), 1213–1242. https://doi.org/10.1021/acs.oprd.9b00140.

(7) Krska, S. W.; DiRocco, D. A.; Dreher, S. D.; Shevlin, M. The Evolution of Chemical High-Throughput Experimentation To Address Challenging Problems in Pharmaceutical Synthesis. *Acc. Chem. Res.* **2017**, *50* (12), 2976–2985. https://doi.org/10.1021/acs.accounts.7b00428.

(8) Collins, K. D.; Gensch, T.; Glorius, F. Contemporary Screening Approaches to Reaction Discovery and Development. *Nature Chem* **2014**, *6* (10), 859–871. https://doi.org/10.1038/nchem.2062.

(9) Isbrandt, E. S.; Sullivan, R. J.; Newman, S. G. High Throughput Strategies for the Discovery and Optimization of Catalytic Reactions. *Angewandte Chemie International Edition* **2019**, *58* (22), 7180–7191. https://doi.org/10.1002/anie.201812534.

(10) Shen, Y.; Borowski, J. E.; Hardy, M. A.; Sarpong, R.; Doyle, A. G.; Cernak, T. Automation and Computer-Assisted Planning for Chemical Synthesis. *Nat Rev Methods Primers* **2021**, *1* (1), 1–23. https://doi.org/10.1038/s43586-021-00022-5.

(11) Wigh, D. S.; Goodman, J. M.; Lapkin, A. A. A Review of Molecular Representation in the Age of Machine Learning. *WIREs Computational Molecular Science* **2022**, *12* (5), e1603. https://doi.org/10.1002/wcms.1603.

(12) Bender, A.; C. Glen, R. Molecular Similarity: A Key Technique in Molecular Informatics. *Organic & Biomolecular Chemistry* **2004**, *2* (22), 3204–3218. https://doi.org/10.1039/B409813G.

(13) Rogers, D.; Hahn, M. Extended-Connectivity Fingerprints. *J. Chem. Inf. Model.* **2010**, *50* (5), 742–754. https://doi.org/10.1021/ci100050t.

(14) Kearnes, S.; McCloskey, K.; Berndl, M.; Pande, V.; Riley, P. Molecular Graph Convolutions: Moving beyond Fingerprints. *J Comput Aided Mol Des* **2016**, *30* (8), 595–608. https://doi.org/10.1007/s10822-016-9938-8.

(15) Cho, H.; Choi, I. S. Enhanced Deep-Learning Prediction of Molecular Properties via Augmentation of Bond Topology. *ChemMedChem* **2019**, *14* (17), 1604–1609. https://doi.org/10.1002/cmdc.201900458.

(16) Ishida, S.; Miyazaki, T.; Sugaya, Y.; Omachi, S. Graph Neural Networks with Multiple Feature Extraction Paths for Chemical Property Estimation. *Molecules* **2021**, *26* (11), 3125. https://doi.org/10.3390/molecules26113125.

(17) Zahrt, A. F.; Athavale, S. V.; Denmark, S. E. Quantitative Structure–Selectivity Relationships in Enantioselective Catalysis: Past, Present, and Future. *Chem. Rev.* **2020**, *120* (3), 1620–1689. https://doi.org/10.1021/acs.chemrev.9b00425.

(18) Henle, J. J.; Zahrt, A. F.; Rose, B. T.; Darrow, W. T.; Wang, Y.; Denmark, S. E. Development of a Computer-Guided Workflow for Catalyst Optimization. Descriptor Validation, Subset Selection, and Training Set Analysis. *J. Am. Chem. Soc.* **2020**, *142* (26), 11578–11592. https://doi.org/10.1021/jacs.0c04715.

(19) Zahrt, A. F.; Henle, J. J.; Rose, B. T.; Wang, Y.; Darrow, W. T.; Denmark, S. E. Prediction of Higher-Selectivity Catalysts by Computer-Driven Workflow and Machine Learning. *Science* **2019**, *363* (6424), eaau5631. https://doi.org/10.1126/science.aau5631.

(20) Yamaguchi, S. Molecular Field Analysis for Data-Driven Molecular Design in Asymmetric Catalysis. *Organic & Biomolecular Chemistry* **2022**, *20* (31), 6057–6071. https://doi.org/10.1039/D2OB00228K.
(21) 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.; Aspuru-Guzik, A. A Comprehensive Discovery Platform for Organophosphorus Ligands for Catalysis. *J. Am. Chem. Soc.* **2022**, *144* (3), 1205–1217. https://doi.org/10.1021/jacs.1c09718.
(22) Brecher, J. Graphical Representation of Stereochemical Configuration (IUPAC Recommendations 2006). *Pure and Applied Chemistry* **2006**, *78* (10), 1897–1970. https://doi.org/10.1351/pac200678101897.
(23) Olen, C. L.; Zahrt, A. F.; Reilly, S. W.; Schultz, D.; Emerson, K.; Candito, D.; Wang, X.; Strotman, N. A.; Denmark, S. E. Chemoinformatic Catalyst Selection Methods for the Optimization of Copper–Bis(Oxazoline)-Mediated, Asymmetric, Vinylogous Mukaiyama Aldol Reactions. *ACS Catal.* **2024**, 2642–2655. https://doi.org/10.1021/acscatal.3c05903.
(24) *GDBM*. https://www.gnu.org.ua/software/gdbm/ (accessed 2023-12-22).
(25) *msgpack/msgpack: MessagePack is an extremely efficient object serialization library. It's like JSON, but very fast and small.* https://github.com/msgpack/msgpack/tree/master (accessed 2023-10-28).
(26) Wu, Z.; Ramsundar, B.; Feinberg, E. N.; Gomes, J.; Geniesse, C.; Pappu, A. S.; Leswing, K.; Pande, V. MoleculeNet: A Benchmark for Molecular Machine Learning. *Chem. Sci.* **2018**, *9* (2), 513–530. https://doi.org/10.1039/C7SC02664A.
(27) Axelrod, S.; Gómez-Bombarelli, R. GEOM, Energy-Annotated Molecular Conformations for Property Prediction and Molecular Generation. *Sci Data* **2022**, *9* (1), 185. https://doi.org/10.1038/s41597-022-01288-4.
(28) Alegre-Requena, J. V.; Sowndarya S. V., S.; Pérez-Soto, R.; Alturaifi, T. M.; Paton, R. S. AQME: Automated Quantum Mechanical Environments for Researchers and Educators. *WIREs Computational Molecular Science* **2023**, *13* (5), e1663. https://doi.org/10.1002/wcms.1663.
(29) LaPlante, S. R.; Fader, L. D.; Fandrick, K. R.; Fandrick, D. R.; Hucke, O.; Kemper, R.; Miller, S. P. F.; Edwards, P. J. Assessing Atropisomer Axial Chirality in Drug Discovery and Development. *J. Med. Chem.* **2011**, *54* (20), 7005–7022. https://doi.org/10.1021/jm200584g.
(30) Basilaia, M.; Chen, M. H.; Secka, J.; Gustafson, J. L. Atropisomerism in the Pharmaceutically Relevant Realm. *Acc. Chem. Res.* **2022**, *55* (20), 2904–2919. https://doi.org/10.1021/acs.accounts.2c00500.
(31) *Discovery of a Covalent Inhibitor of KRASG12C (AMG 510) for the Treatment of Solid Tumors | Journal of Medicinal Chemistry*. https://pubs.acs.org/doi/10.1021/acs.jmedchem.9b01180 (accessed 2024-02-17).
(32) *Efficient Quantum Chemical Calculation of Structure Ensembles and Free Energies for Nonrigid Molecules | The Journal of Physical Chemistry A*. https://pubs.acs.org/doi/10.1021/acs.jpca.1c00971 (accessed 2024-02-17).
(33) *Merck molecular force field. I. Basis, form, scope, parameterization, and performance of MMFF94 - Halgren - 1996 - Journal of Computational Chemistry - Wiley Online Library*. https://onlinelibrary.wiley.com/doi/10.1002/(SICI)1096-987X(199604)17:5/6%3C490::AID-JCC1%3E3.0.CO;2-P (accessed 2024-02-17).
(34) Pracht, P.; Grimme, S. Calculation of Absolute Molecular Entropies and Heat Capacities Made Simple. *Chem. Sci.* **2021**, *12* (19), 6551–6568. https://doi.org/10.1039/D1SC00621E.
(35) Jensen, F. Segmented Contracted Basis Sets Optimized for Nuclear Magnetic Shielding. *Journal of Chemical Theory and Computation* **2015**, *11* (1), 132–138. https://doi.org/10.1021/ct5009526.

(36) Boyko, Y. D.; Huck, C. J.; Ning, S.; Shved, A. S.; Yang, C.; Chu, T.; Tonogai, E. J.; Hergenrother, P. J.; Sarlah, D. Synthetic Studies on Selective, Proapoptotic Isomalabaricane Triterpenoids Aided by Computational Techniques. *Journal of the American Chemical Society* **2021**, *143* (4), 2138–2155. https://doi.org/10.1021/jacs.0c12569.
(37) Pybind/Pybind11, 2023. https://github.com/pybind/pybind11 (accessed 2023-10-28).
(38) Virtanen, P.; Gommers, R.; Oliphant, T. E.; Haberland, M.; Reddy, T.; Cournapeau, D.; Burovski, E.; Peterson, P.; Weckesser, W.; Bright, J.; van der Walt, S. J.; Brett, M.; Wilson, J.; Millman, K. J.; Mayorov, N.; Nelson, A. R. J.; Jones, E.; Kern, R.; Larson, E.; Carey, C. J.; Polat, İ.; Feng, Y.; Moore, E. W.; VanderPlas, J.; Laxalde, D.; Perktold, J.; Cimrman, R.; Henriksen, I.; Quintero, E. A.; Harris, C. R.; Archibald, A. M.; Ribeiro, A. H.; Pedregosa, F.; van Mulbregt, P.; SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* **2020**, *17*, 261–272. https://doi.org/10.1038/s41592-019-0686-2.
(39) Bentley, J. L. Multidimensional Binary Search Trees Used for Associative Searching. *Commun. ACM* **1975**, *18* (9), 509–517. https://doi.org/10.1145/361002.361007.
(40) *scipy.spatial.KDTree — SciPy v1.11.4 Manual*. https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.KDTree.html (accessed 2023-12-22).
(41) Rego, N.; Koes, D. 3Dmol.Js: Molecular Visualization with WebGL. *Bioinformatics* **2015**, *31* (8), 1322–1324. https://doi.org/10.1093/bioinformatics/btu829.
(42) Sullivan, C. B.; Kaszynski, A. A. PyVista: 3D Plotting and Mesh Analysis through a Streamlined Interface for the Visualization Toolkit (VTK). *Journal of Open Source Software* **2019**, *4* (37), 1450. https://doi.org/10.21105/joss.01450.
(43) Schroeder, W.; Martin, K.; Lorensen, B. *The Visualization Toolkit (4th Ed.)*; Kitware, 2006.