Geometry Optimization: A Comparison of Different Open-Source Geometry Optimizers

Akhil Shajan,[†] Madushanka Manathunga,[†,‡] Andreas W. Götz,[*,¶] and Kenneth M. Merz, Jr.[*,†,‡]

[†]*Department of Chemistry, Michigan State University, East Lansing, Michigan 48824, United States*

[‡]*Department of Biochemistry and Molecular Biology, Michigan State University, East Lansing, Michigan 48824, United States*

[¶]*San Diego Supercomputer Center, University of California San Diego, La Jolla, California 92093- 0505, United States*

E-mail: agoetz@sdsc.edu; kmerz1@gmail.com

Abstract

Based on a series of energy minimizations with starting structures obtained from the Baker test set of 30 organic molecules, a comparison is made between various open-source geometry optimization codes that are interfaced with the open-source QUantum Interaction Computational Kernel (QUICK) program for gradient and energy calculations. The findings demonstrate how the choice of the coordinate system influences the optimization process to reach an equilibrium structure. With fewer steps, internal coordinates outperform Cartesian coordinates while the choice of the initial Hessian and Hessian update method in quasi-Newton approaches made by different optimization algorithms also contributes to the rate of convergence. Furthermore, an available open-source machine learning method based on Gaussian Process Regression (GPR) was evaluated for energy minimizations over surrogate potential energy surfaces with both Cartesian and internal coordinates, with internal coordinates outperforming Cartesian. Overall, geomeTRIC and DL-FIND with their default optimization method as well as with GPR-based model using Hartree–Fock theory with the 6-31G** basis set, needed a comparable number of geometry optimization steps to the approach of Baker using a unit matrix as the initial Hessian to reach the optimized geometry. On the other hand, the Berny and Sella offerings in ASE outperformed the other algorithms. Based on this we recommend using the file-based approaches, ASE/Berny and ASE/Sella, for large-scale optimization efforts, while if using a single executable is preferable, we now distribute QUICK integrated with DL-FIND.

1 Introduction

The importance of intermolecular interactions in atomistic simulations of materials, chemical reactions, and biological processes is well known. Optimizing the molecular geometry to find the stationary points on the potential energy surface, followed by calculation of molecular properties and characterization of the interactions at a certain geometry, is a core technique used in theoretical chemistry to analyze molecular structure and intermolecular interactions.[1–3] By taking into account the energy, $E(x_0)$, at a point x_0 on a potential energy surface, we can use the Taylor series to form a quadratic approximation to describe the energy at a nearby point, $x = x_0 + \triangle x$:

$$E(x) = E(x_0) + g_0^T \triangle x + 1/2 \triangle x^T H_0 \triangle x$$

where g_0 is the gradient vector (dE/dx) at x_0, H_0 is the Hessian matrix (d^2E/dx^2) at x_0, and $\triangle x = x - x_0$. Most nonlinear optimization algorithms are based on this local quadratic approximation of the potential energy surface.[4,5] By differentiating with respect to the coordinates, one could form an approximation for the gradient, which is given by:

$$g(x) = g_0 + H_0 \triangle x$$

On the potential energy surface, the gradient, $g(x)$, vanishes at a stationary point, $g(x) \equiv \nabla E = 0$. Hence, in the local quadratic approximation of the potential energy surface, the displacement to the minimum at the stationary point is given by:

$$\triangle x = -H_0^{-1} g_0$$

This is known as the Newton-Raphson step. It is an integral part of almost all quantum chemistry geometry optimization approaches. The gradient here can be obtained by differentiation of the energy with respect to the coordinates, and the Hessian can be obtained using numerical or analytical methods of the second derivative of the energy with respect to the coordinates.

Atomistic studies are of interest for a variety of stationary points, such as minima, transition states, and conical intersections.[6–10] The most effective and extensively used techniques for identifying these stationary points are Newton and quasi-Newton techniques, which typically employ a sequential optimization cycle, in which a guess optimal geometry is steadily enhanced by employing the gradient and either an exact or an approximate Hessian. For minimization, the Hessian must contain exclusively positive eigenvalues (i.e., positive definite). If one or more eigenvalues are negative, the step will be in the direction of a first-order or higher-order saddle point, which can be useful when locating a transition state.

The choice of the coordinate system, the optimization algorithm, the quality of the Hessian, and the algorithm used to determine the step, $\triangle x$, are four factors that influence the efficiency of a geometry optimization. Strong coupling between coordinates, narrow gullies, and curved valleys provide significant hurdles to even the best optimization algorithms, making the choice of a suitable coordinate system essential to the efficiency of any optimization. Since the Cartesian coordinate system is the easiest approach to define the molecular geometry, it is usually used for the evaluation of the energy and gradient. However, the potential energy surface is strongly nonlinear and coordinates are coupled, so only small steps can be taken downhill. Cartesian coordinates perform significantly worse for flexible, acyclic systems than they do for constrained, cyclic molecules. In Cartesian coordinates, it is also much harder to impose constraints.[11,12] Internal coordinates on the other hand are better at reflecting the overall atomic motions. Internal coordinates can describe displacements along curved pathways and decouple various types of molecular displacements. As a result, the optimization algorithm can proceed with greater efficiency.[13,14]

It is possible to calculate the entire Hessian matrix analytically[15] for each step at the current point on the potential energy surface. Only one step is needed to get to the minimum of the energy if the local potential energy surface is quadratic. Since actual potential energy surfaces are rarely quadratic, one must take several Newton-Raphson steps to get at a stationary point. It is thus often impractical or undesirable to directly calculate the full Hessian due to the amount of computational work this typically involves. Instead one can resort to quasi-Newton techniques, whose foundation is a computationally cheap approximation of the Hessian. At each stage of a quasi-Newton optimization, the Hessian is improved using the difference between the calculated gradient change and the change anticipated by the approximate Hessian.

$$H_{\text{new}} = H_{\text{old}} + \triangle H$$

This updated Hessian can be used to determine the step, $\triangle x$. The Hessian can be updated in a number of ways, and several algorithmic schemes can be used to determine the step and the new coordinate system along the minimum to get at the optimized geometry.[16–19]

Additionally, the initial estimate of the Hessian or second derivative matrix does have an effect on the performance of geometry optimization with quasi-Newton methods; the more precise the initial estimate, the faster the convergence.[2,20,21] One can use an exact Hessian, which is expensive. However, often the estimate does not have to be extremely precise since the Hessian matrix gradually improves in quality as it is updated using gradient information during the search for the minimum. The unit matrix is the most basic and popular approximation, in which case the first optimization step corresponds to a gradient descent. The nature of the atoms, the bonds connecting them, and other pertinent structural information about the molecule are all ignored in this case despite the fact that this is an unbiased choice. All connection between coordinates is initially ignored, and flexible coordinates (such as torsion and ring deformation) are not separated from stiff modes (such as bond stretching). At the price of additional optimization steps, this data must be gathered during the optimization process. For cyclic compounds, whose coordinates are by nature strongly coupled, this is a poor approximation. Simple schemes to approximate the initial Hessian based on parameterized values for bond, angle and dihedral force constants have thus been proposed to reduce the number of required geometry optimization steps.[2,22,23]

Recently, the integration of machine learning techniques into traditional methods for calculating potential energy surfaces (PES) has led to even more efficient optimization approaches. In particular, machine learning can accelerate the location of minima[24,25] and transition states.[26,27] One of the most popular approaches in this field is to use surrogate potential energy surfaces. Surrogates are constructed by fitting a model, such as a Gaussian process regression (GPR) model, to the data points that have already been evaluated. This model is then used to suggest new geometries for evaluation, without the need for expensive *ab initio* calculations. As more data points become available, the surrogate PES is continuously updated and used to identify the location of the next guess minimum. This guess point is then subject to single-point energy and gradient evaluations on the true PES and then added to the dataset. The accuracy of the surrogate PES predicted by a GPR model heavily relies on the appropriate selection of a kernel function. The kernel function determines the covariance between the input space points and directly impacts the accuracy of the predicted PES. Compared to classical second-order methods, such as the local quadratic approximation in the quasi-Newton method discussed above, surrogate-based approaches offer a more accurate representation of the PES, particularly in regions where the true PES is highly nonlinear or difficult to sample. The careful selection of the appropriate kernel function and coordinate system is essential in constructing an accurate GPR model that can precisely predict the PES and ultimately reduce the number of required geometry optimization steps.[28,29] Further discussion on this topic will be presented later in the paper.

These algorithmic schemes can be found in most electronic structure software packages. However, many of these packages are either not open source or their optimized geometry optimization algorithms are not generally available for inclusion into other open-source projects. Here, we discuss the performance of various open-source geometry optimization codes including both conventional methods and a GPR-based machine learning method for energy minimization, coupled with the free and open-source QUantum Interaction Computational Kernel (QUICK) program[30] for energy and gradient calculations. Several of the new GPR models are not available in open-source software, restricting our exploration to the method that is available in the development version of DL-FIND. While this model is effective, other more efficient models have been reported in the literature which, however, are not yet publicly available neither in free and open-source nor commercial software. We compare these results to the legacy optimizer in QUICK, basing results on the number of iterations it takes to converge geometries of an established test set of representative molecules.

2 Overview of Open-Source Geometry Optimization Software

2.1 QUICK–Legacy Optimizer

The QUantum Interaction Computational Kernel (QUICK) program[30] is an open-source, GPU enabled,[31,32] *ab initio* and density functional theory program,[33] which has been developed for QM and QM/MM calculations with Gaussian basis functions.[34,35] It contains a limited-memory Broyden-Fletcher-Goldfarb-Shanno (L–BFGS) optimization algorithm which uses a Cartesian coordinate system as input. L–BFGS is an optimization technique from the family of quasi-Newton methods that limits its memory usage while approximating the Broyden-Fletcher-Goldfarb-Shanno algorithm (BFGS).[36,37]

In general, the BFGS update of the inverse Hessian is:[3]

$$\triangle H^{-1} = \frac{\triangle x \triangle x^T}{\triangle x^T \triangle g} - \frac{H_{\text{old}}^{-1} \triangle g \triangle g^T H_{\text{old}}^{-1}}{\triangle g^T H_{\text{old}}^{-1} \triangle g}$$

where the updated inverse Hessian is used to form the Newton step, $\triangle x = H^{-1} \triangle g$. The L-BFGS method avoids the complete Hessian or its inverse from being stored because doing so would need $O(n^2)$ memory for n variables. Since L-BFGS starts with a diagonal inverse Hessian and only stores the x and g vectors from a few prior iterations, the storage requirement is only $O(n)$. The inverse Hessian is expressed as a diagonal Hessian plus the updates using the saved vectors. The new coordinates, $x_{\text{new}} = x_{\text{old}} - H^{-1} g_{\text{old}}$, may be expressed in terms of dot products between vectors, therefore the product of the updated inverse Hessian and the gradient only requires $O(n)$ work.

2.2 DL–FIND

DL-FIND is an open-source geometry optimization library that provides methods for local minimization, conical intersection optimization, population-based (global) optimization, reaction path optimization, and transition state search making it a versatile choice for molecular codes.[38]

Cartesian coordinates, redundant internal coordinates, and hybrid delocalized internal coordinates (HDLC) are available options for DL–FIND geometry optimizations. To compare the optimizers with the GPR model that uses the Matérn kernel[39] to build the surrogate potential energy surface, the development version of DL-FIND was utilized alongside the stable version utilizing the L–BFGS method. The stable version of DL–FIND with the L–BFGS method has been the default geometry optimizer in QUICK since version QUICK–22.03.[30]

The common interface between both versions and the QUICK optimizer library is as shown in Figure 1. The main DL–FIND geometry optimization driver routine is called from QUICK and does not exit until the optimization is finished. During the optimization, DL–FIND calls another interface routine when it needs to exchange information with QUICK during an optimization step. The input parameters are obtained from QUICK using the initial interface call. These are the initial coordinates and the user-specified DL–FIND settings that control the optimization algorithm. The iterative optimization process then starts, and a new call is placed each time DL–FIND requires an energy or gradient. DL–FIND provides a set of Cartesian coordinates to QUICK and receives back the required information. During each optimization step, relevant information on the progress of the optimization is printed, and at the end of the geometry optimization, the routine reports the optimized set of Cartesian coordinates back to QUICK. To employ the GPR model in geometry optimization, a new surrogate PES is constructed incrementally using a specified coordinate system transformed from Cartesian coordinates after each *ab initio* calculation. The conventional L-BFGS algorithm is used to iteratively locate the minimum on this surrogate PES (micro-iterations), and then another *ab initio* calculation is performed in the Cartesian coordinates obtained from back-transformation. This entire cycle (macro-iterations) is repeated until the *ab initio* gradient is below the preset convergence threshold.[40]

2.3 GeomeTRIC

GeomeTRIC is an open-source geometry optimization program that takes input in form of Cartesian coordinates and executes external electronic structure codes through wrapper functions with file-based data-exchange to obtain energy and gradients. Translation-rotation-internal coordinates (TRIC), delocalized internal coordinates (DLC), hybrid delocalized internal coordinates (HDLC), and redundant internal coordinates are all implemented by GeomeTRIC.[41] The code supports optimizations with constraints and transition state searches.

Translation-rotation-internal coordinates (TRIC) are created by including translations and rotations in the primitive internal coordinates set coupled with existing delocalization techniques. The initial Hessian in the space of primitive internal coordinates is a diagonal matrix with a few minor adjustments adopted from Schlegel's proposed values.[42] The translations and rotations, and atomic Cartesian coordinates are given force constants of 0.05, bonds and angles involving non-covalent distances are given force constants of 0.1, and dihedral angles have their force constants set to 0.023.

GeomeTRIC uses a wrapper around QUICK, which is unlike DL–FIND as shown in Figure 1. Cartesian coordinates are used as input by GeomeTRIC, which transforms them to internal coordinates for optimization. After each cycle of optimization, it updates its own Hessian estimate using the BFGS algorithm. In order to generate an input file for QUICK to obtain the energy and Cartesian gradient, internal coordinates are converted to Cartesian coordinates at the end of each cycle. The Cartesian gradient from QUICK is used to calculate the internal coordinates gradient followed by Hessian update to estimate the search direction and step size.

2.4 Atomic Simulation Environment (ASE)

The Python-based Atomic Simulation Environment (ASE) was designed with the goal of setting up, directing, and analyzing atomistic simulations. Numerous features of ASE include molecular dynamics with various controls, including thermostats, structure improvement utilizing atomic forces, saddle point searches on potential energy surfaces, genetic algorithms for structure or chemical composition optimization, basin hopping or minima hopping algorithms for global structure optimization, analysis of phonon modes for solids or molecular vibrational modes.[43]

Here, we focus exclusively on the geometry optimization methods provided by ASE, in particular the ASE internal L–BFGS optimizer, and the Sella and Berny algorithms. We will emphasize the Sella and Berny methods more as the L–BFGS approach has previously been summarized. The Berny geometry optimization algorithm is based on an earlier program written by H. B. Schlegel.[44] Sella is an open-source tool used for automating the process of finding saddle points and minimizing molecular structures.[45] It works by converting Cartesian coordinates into internal coordinates, automatically replacing pathological linear angles with improper dihedrals, and introducing necessary modifications like dummy atoms and constraints to ensure that the dummy atom does not drift unnecessarily over the course of optimization.[46] The algorithm utilizes Hessian diagonalization based on internal coordinates to determine the direction of minima, leading to a partially-exact Hessian matrix. This matrix guides the optimization process using a state-of-the-art constrained partitioned rational function approach, effectively steering the system towards local energy minima. The Berny method utilizes a valence force field to construct an estimate of a Hessian at the start of the optimization. This approximation is then updated using the energies and first derivatives computed along the optimization pathway. The update is typically carried out using an iterative BFGS algorithm for minimization and a modified version of the original Schlegel update process for internal coordinate optimizations. It is ideally suited for optimizing covalently bound compounds since it is based on a redundant set of internal coordinates.

ASE uses the L–BFGS method as its internal optimization method of choice, while wrappers are used for the Berny optimizer from the PyBerny implementation and for the Sella optimizer. The actual geometry optimization method as well as the delocalized coordinates, screening functions, etc. are generated either directly or through PyBerny or Sella by the ASE program, which functions as a wrapper around the QUICK program with a file-based interface in the same manner as GeomeTRIC. The ASE program generates all of the QUICK input files and then runs QUICK to obtain the energy and gradient. The working scheme of ASE with QUICK is depicted in Figure 1.

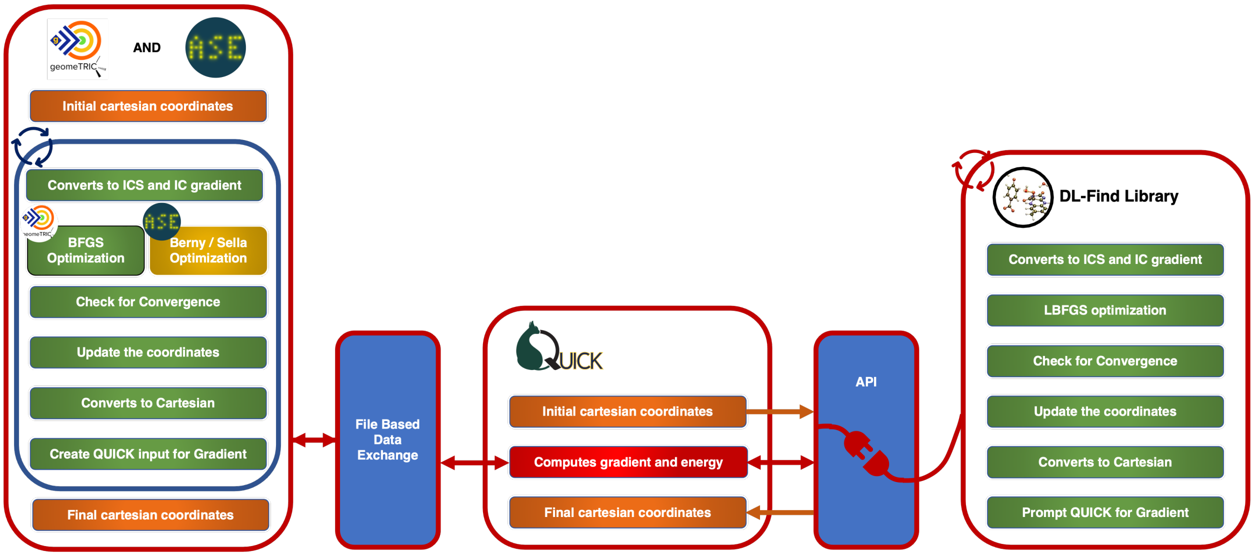

geomeTRIC
AND
ASE
Initial cartesian coordinates
Converts to ICS and IC gradient
BFGS Optimization
Berny / Sella Optimization
Check for Convergence
Update the coordinates
Converts to Cartesian
Create QUICK input for Gradient
Final cartesian coordinates
File Based Data Exchange
QUICK
Initial cartesian coordinates
Computes gradient and energy
Final cartesian coordinates
API
DL-Find Library
Converts to ICS and IC gradient
LBFGS optimization
Check for Convergence
Update the coordinates
Converts to Cartesian
Prompt QUICK for Gradient

Figure 1: QUICK interface with the different open-source optimizers. The optimization steps of various optimizers are indicated by green color boxes where the initial Cartesian coordinates and gradient are transformed to an internal coordinate system (ICS) and internal gradients respectively for further optimization. The red box denotes the gradient computation performed by QUICK for various optimizers. The final coordinates obtained, and the initial coordinates used for calculation are represented by orange boxes. The yellow box denotes the PyBerny or Sella wrapper used by ASE for optimization. QUICK utilizes the DL–FIND library for optimization by passing gradient as required at each iteration. geomeTRIC and ASE carry out optimization independently utilizing QUICK for gradient calculation at each iteration.

3 Methodology

We developed calling interfaces for GeomeTRIC and ASE to execute QUICK by passing Cartesian coordinates, writing a QUICK input file, and retrieving energies and Cartesian gradients from the QUICK output file. While ASE uses the Cartesian coordinate system for the L–BFGS method with the unit matrix as the intial Hessian and the internal coordinate system for the Berny algorithm where the Hessian is estimated using Schlegel's proposed values[42] and the internal coordinate system for the Sella algorithm utilises the scheme of Fischer and Almlof[47] to initializes the Hessian matrix; GeomeTRIC uses Translation-rotation-internal coordinates (TRIC) along with its own Hessian estimate with a few minor adjustments of Schlegel's proposed values for geometry optimization using the BFGS algorithm. DL–FIND has been implemented internally within QUICK and makes use of a non-redundant hybrid-delocalized internal coordinate system and a unit matrix as the initial Hessian for optimization with the L–BFGS method. The development version of DL–FIND was used for GPR-based geometry optimizations in either Cartesian or hybrid-delocalized internal coordinate space. The built-in QUICK–legacy minimizer employs a Cartesian coordinate system with a unit matrix as the initial Hessian for L–BFGS based optimizations. The tests performed with these optimizers give us a clear understanding of the differences in performance between various methods that use the Cartesian coordinate system or any other internal coordinate system along with various Hessian estimations.

All Hartree–Fock (HF) calculations were performed with QUICK using the restricted Hartree–Fock method with the 6-31G** basis set (HF/6-31G**) for the energy and gradient. Results obtained with DL-FIND using different basis sets (HF/STO-3G, HF/6-31G, HF-6-31G*, HF/def2-SVP) as well as a generalized gradient approximation (GGA) and hybrid-GGA density functional method (BP86/def2-SVP, B3LYP/def2-SVP) are summarized in the Supporting Information. The convergence criteria for the geometry optimizations were as follows: a maximum gradient component of less than 0.00045 au, a change in energy from the previous step of less than 10^{-6} Hartree, and a maximum predicted displacement of less than 0.0018 Å per coordinate. These geometry convergence criteria have been commonly utilized by the majority of past research that has examined the benchmark being studied, thus enabling direct comparisons of results. In order to guarantee accurate gradients, we used the TIGHTINT keyword in QUICK, which requests tight numerical cutoffs and SCF convergence. With these settings, the root-mean-square (RMS) change in the density matrix is less than 10^{-7} au and the maximum change in the density matrix is less than 10^{-5} au.

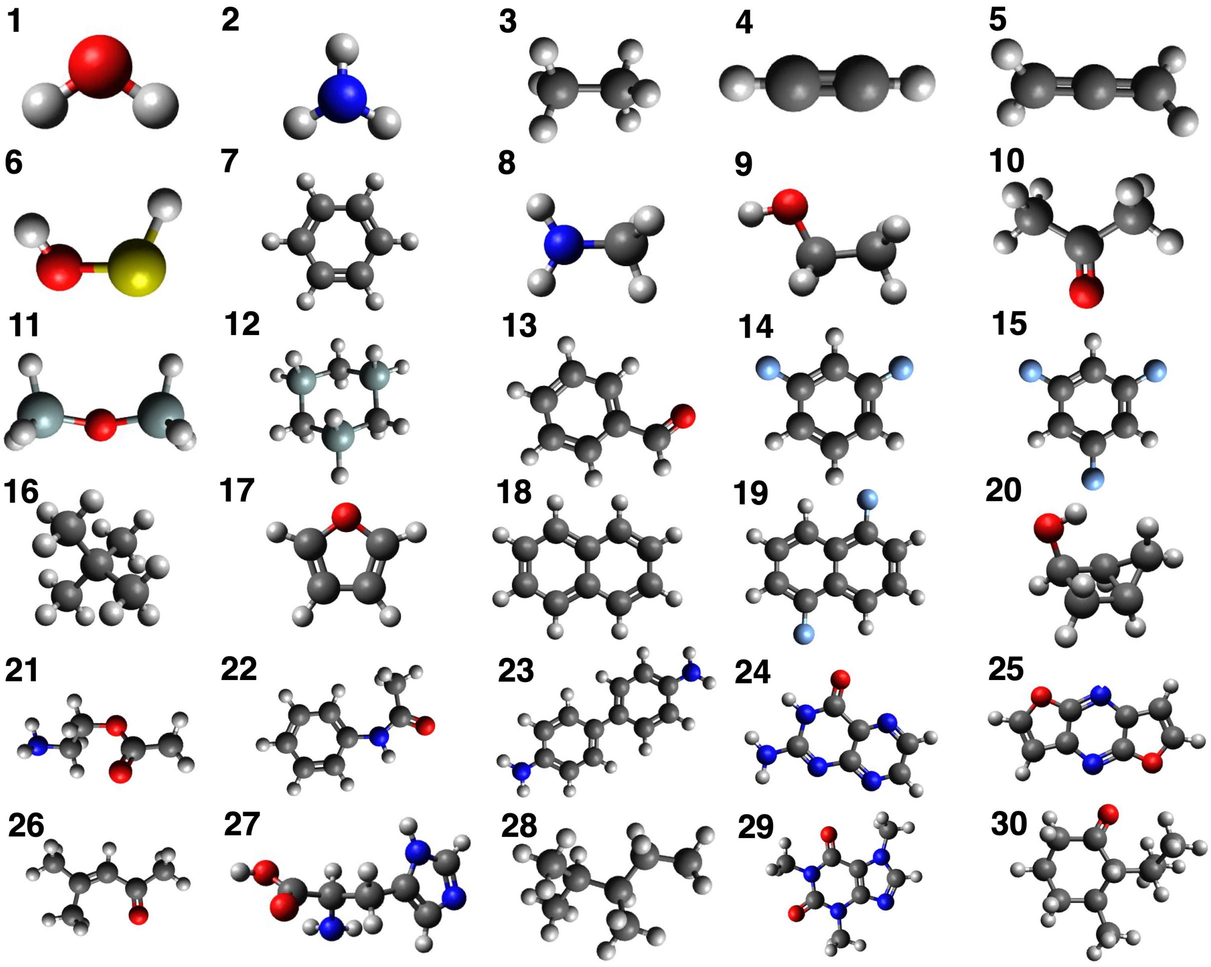


Figure 2: Initial geometries for the molecules in the Baker set. Oxygen is depicted in red, nitrogen in blue, carbon in gray, sulfur in yellow, fluorine in light blue, silicon in dark cyan, and hydrogen in white.

4 Results and discussion

In order to compare different geometry optimizers, a test suite of 30 molecules as depicted in Figure 2, originally suggested by Baker,[20] was used as the initial geometry inputs for the calculations. The test suite contains a variety of compounds, including fused bi- and tri-cyclics like caffeine and difluropyrazine as well as smaller systems like water and ammonia. Input files containing initial geometries for optimization are available in the Supporting Information. Baker used the Eigenvector Following (EF) algorithm[48] at the restricted Hartree–Fock (RHF) level with the STO-3G basis set (HF/STO-3G) for optimizations in both Cartesian and internal coordinates with a convergence criterion for the gradient of 3.0×10^{-4} au. Table 1 lists the number of steps required to reach the minima for each of the molecules for a number of different geometry optimization algorithms in comparison to Baker's results. The sum of optimization steps for all molecules in the test suite and the average number of optimization steps per molecule is also reported. Given the vagaries of examining the data on a molecule-to-molecule basis we feel these averages give the best overall assessment of the performance of the geometry optimization algorithms investigated in this work.

With quasi-Newton techniques Baker reported requiring a total of 765 geometry cycles (on average 26 steps per molecule) with a Cartesian coordinate system while requiring only 371 geometry cycles (on average 12 steps per molecule) with an internal coordinate system. In both cases Baker employed the unit matrix as the initial Hessian. Using an initial Hessian from a molecular mechanics model, Baker managed to further reduce the number of steps to 240 (on average 8 steps per molecule).[20] This data is not shown in Table 1 but the numbers should be kept in mind for the following discussion.

The total number of geometry cycles required to optimize the same set of molecules using the optimizers tested here ranges from 728 for DL–FIND/GPR-Cartesian, 683 for QUICK–Legacy, 613 for ASE/L–BFGS, 356 for DL–FIND/ L–BFGS, 326 for DL–FIND/GPR-Internal, 287 for geomeTRIC, 190 for ASE/Berny, to as low as 187 for ASE/Sella. Geometry optimization in Cartesian coordinates without sophisticated initial Hessian guess

(ASE/L–BFGS, DL–FIND/GPR-Cartesian and QUICK–Legacy) thereby required between 20 and 24 steps per molecule on average. Optimization in internal coordinate space is much more efficient, requiring only 11 to 12 optimization steps (DL-FIND/LBFGS, DL-FIND/GPR-Internal). Using both internal coordinates and a better initial Hessian reduces this further to 6 to 10 optimization steps (ASE/Berny, ASE/Sella, GeomeTRIC). Here ASE/Berny and ASE/Sella use redundant internal coordinates, while geomeTRIC uses translation-rotational-internal coordinates, and DL–FIND uses delocalized internal coordinates. Except for ASE/Berny and geomeTRIC, which use a modified version of the original Schlegel update process, as well as ASE/Sella, all methods initialize the Hessian to be the unit matrix. These results reconfirm the two well-known observations that i) internal coordinates outperform the Cartesian coordinate system for molecular geometry optimizations, and ii) good approximations to the initial Hessian are also important to reduce the number of required steps.

We finally point out that while Baker used the HF/STO-3G method and we are using HF/6-31G**, the observed overall trends are comparable (see Table 1). To provide a more comprehensive understanding of the test suite, we used the DL-FIND/L-BFGS implementation to compare the effect of different levels of theory and basis sets in addition to HF/6-31G**. The results of this comparison are summarized in Table S1 of the Supporting Information. While there are differences for individual molecules, the total number of steps increases only slightly with increasing flexibility in the basis set, from 331 for HF/STO-3G to 356 as mentioned above for HF/6-31G**. The total number of steps remains very similar between HF and representative generalized gradient approximation (GGA) and hybrid-GGA density functional methods. This comparison allows for a clearer and more complete picture of the observed overall trends, which are consistent with those presented in Table 1.

As mentioned above, the number of geometry cycles required to converge to the optimized geometry can be reduced with a concomitant increase in computational time depending on the Hessian used. Bakken and Helgaker, for instance, already addressed this problem, where it is noted that utilizing precise Hessians at each step reduces the number of required iterations while compromising on total computation time. Only 125 and 111 steps are required for the Baker test set with HF/STO-3G using Cartesian and redundant internal coordinates, respectively, when using exact Hessians at each step.[21]

Baker, in Ref. 20, used natural internal coordinates and the initial Hessian provided by the CVFF force field in combination with an eigenvector following (EF) approach to reduce the number of iterations required for HF/STO-3G geometry optimization to 240 steps. Bakken and Helgaker managed to optimize the Baker test set in 185 steps using a combination of extra-redundant internal coordinates, a good approximate initial Hessian with BFGS updates, and optimized step size.[21] A slightly larger number of 198 steps was reported when using the more flexible 6-31G* basis set. Swart and Bickelhaupt managed to further reduce the number of geometry optimization steps using the delocalized coordinates setup of Baker, a combination of quasi-Newton steps with GDIIS,[49] and a modification of Lindh's force constant model to generate initial Hessians,[50] resulting in an impressive 173 steps for the Baker test set using the PW91 density functional and a large triple zeta Slater type basis set with polarization functions (TZP).[2] It therefore stands to reason that replacing the unit matrix with a better initial Hessian approximation should also reduce the number of geometry optimization steps required by the L-BFGS optimization employed by DL-FIND.

The use of machine learning models such as GPR with efficient kernel functions[24] in conjunction with different internal coordinate systems can also reduce the number of required optimization cycles. For instance, Meyer and Hauser achieved 225 steps for the Baker test set by using reduced redundancy Z-matrix-derived internal coordinates with HF/STO-3G.[28] Raggi *et al.* introduced a restricted-variance optimization (RVO) scheme that utilizes a Hessian model function[23] to generate a non-redundant set of internal coordinates for molecular geometries, making the surrogate model invariant to translations and rotations. With the HF/6-31G basis set, they achieved 225 steps using RVO, while with DFT(B3LYP)/def2-SVP, they obtained 241 steps.[25] Teng, Huang, and Bao demonstrated that combining the gradient-enhanced universal kriging (GEUK) algorithm with an adaptive *ab initio* prior mean function, which incorporates prior physical knowledge into surrogate-based optimization, and incomplete internal/incomplete Cartesian methods can reduce the number of steps to 165 as shown for the BP86/def2-SVP level of theory.[29]

Herein we explored several different open-source quasi-Newton approaches (L-BFGS and some modified BFGS algorithms) and available machine learning (GPR) models, utilizing various internal coordinate systems with either the unit matrix as the starting Hessian (as in the L-BFGS approaches) or a more sophisticated first estimate (geomeTRIC, ASE/Berny, ASE/Sella). In all cases, not unexpectedly, internal coordinates outperformed methods using Cartesian coordinates, but the numerical details of the implementation of the internal coordinate methods make a significant difference (*e.g.*, ASE/Berny *versus* geomeTRIC). Furthermore, the use of more sophisticated initial Hessians and geometry update algorithms resulted in a significant reduction of required steps. Importantly, the open-source implementations that are currently available require approximately the same number of steps as the best traditional or machine-learning based approaches that have been reported in the literature offering the community state-of-the-art optimizers in an open-source offering.

5 Conclusion

We have presented a comparison of different open-source geometry optimization codes available to the community. The performance of the geometry optimization algorithms implemented in these codes has been evaluated for a test set of 30 molecules that was originally proposed by Baker. We find results in line with expectations for the various optimization algorithms. While there are differences for individual molecules, when Cartesian coordinates are used, the optimization algorithms take at least 20 and in the worst case 24 steps on average per molecule of this test set. For internal coordinate optimization algorithms, the number of steps is greatly reduced requiring about 12 steps per molecule when using a unit matrix as

Table 1: Number of steps required to optimize the geometry of molecules from the Baker test set using open-source QUICK with various open-source geometry optimization routines. Results for Baker/Cartesian and Baker/Internal are for HF/STO-3G and taken from Baker's work.[20] All other results are obtained with HF/6-31G**. QUICK-Legacy and ASE/L-BFGS both use L-BFGS in Cartesian coordinate space. ASE/Berny and geomeTRIC use a modified version of the Schlegel update process, while other methods initialize the Hessian as the unit matrix. DL-FIND, ASE/Berny and geomeTRIC use different forms of internal coordinate systems(see discussion).

Molecules	Baker		QUICK-Legacy	ASE			geomeTRIC	DL–FIND		
	EF Cartesian	EF Internal	L–BFGS Cartesian	L–BFGS Cartesian	Berny Internal	Sella Internal	BFGS Internal	L–BFGS Internal	GPR Cartesian	GPR Internal
Water	5	7	7	5	4	4	4	6	6	5
Ammonia	7	7	9	6	4	4	4	6	6	4
Ethane	7	6	10	8	4	4	4	5	7	5
Acetylene	7	5	11	6	5	4	5	7	8	7
Allene	10	8	11	5	5	4	6	6	9	6
Hydroxysulfane	21	17	17	16	6	7	7	15	15	18
Benzene	6	5	16	4	3	3	4	4	6	5
Methylamine	10	7	14	8	4	5	6	7	14	7
Ethanol	18	8	16	15	5	6	5	10	14	9
Acetone	22	7	20	18	5	5	6	10	16	10
Disilyl-ether	27	13	21	30	9	10	14	17	26	10
1,3,5-Trisilacyclohexane	36	14	22	25	4	5	10	20	26	16
Benzaldehyde	17	9	17	17	7	7	11	8	18	7
1,3-Difluorobenzene	8	9	9	10	4	5	9	6	12	7
1,3,5-Trifluorobenzene	7	6	9	6	4	4	11	6	8	7
Neopentane	10	7	11	8	4	3	5	6	15	5
Furan	10	11	13	10	5	6	6	7	14	7
Naphthalene	11	9	12	9	5	5	7	7	13	7
1,5-Difluoronaphthalene	16	11	18	13	5	6	10	7	14	8
2-Hydroxybicyclopentane	45	23	38	39	9	11	12	28	32	23
ACHTAR10	51	19	41	19	10	7	11	15	35	13
ACANIL01	36	10	38	33	7	7	12	9	30	12
Benzidine	26	25	20	41	7	7	13	17	119	16
Pterin	23	12	28	22	11	9	12	12	22	10
Difuropyrazine	21	11	21	16	7	8	10	9	19	8
Mesityl-oxide	38	9	30	23	7	6	10	11	23	11
Histidine	102	44	113	66	14	13	22	34	75	33
Dimethylpentane	29	15	10	22	6	4	21	18	22	13
Caffeine	39	11	32	31	10	8	12	11	30	10
Menthone	100	26	68	82	10	10	18	32	74	27
Sum	765	371	683	613	190	187	287	356	728	326
Average per Molecule	26	12	23	20	6	6	10	12	24	11

initial Hessian and as little as 6 steps per molecule when using more sophisticated initial Hessians and geometry update algorithms. Overall, the best performing open-source software was Sella and the Berny algorithm as implemented in PyBerny, both offered through ASE interfaces, closely followed by the BFGS method as implemented in geomeTRIC. Additionally, the release version of DL-FIND, utilizing a conventional L-BFGS algorithm along with delocalized internal coordinates, also demonstrated commendable performance. Given this analysis, we are using the release version of DL–FIND as the default optimizer in QUICK for simplified distribution within a single binary executable and usage without requiring Python wrappers. This integration enhances the inherent optimization framework and introduces supplementary features like conical intersection optimization, reaction path optimization, and transition state search. Additionally, it is worth noting that the development version of DL-FIND will incorporate GPR optimization for both local minima and transition state optimization. This upgrade will be incorporated into QUICK once it becomes the release version. While DL-FIND combined with QUICK offers substantial capabilities, we highly recommend utilizing the Berny algorithm as implemented in PyBerny or the Sella geometry optimization software for large-scale production optimization efforts due to their exceptional performance (*e.g.,* in the creation of large synthetic data sets for ML/AI efforts). These optimizers are accessible for usage with QUICK through the ASE interface as mentioned in the Supporting Information. The powerful combination of QUICK with the latest version of DL-FIND, ASE/Berny or ASE/Sella provides a reliable and robust open-source solution for efficiently optimizing molecular geometry based on *ab initio* and density functional theory methods, combined with the computational capabilities of graphics processing units (GPUs).

6 Data and Software Availability

The features described here are available in QUICK-22.03. The latest version of QUICK and all previous releases are available on GitHub (https://github.com/merzlab/QUICK). Input files used for geometry optimizations are available in the Supporting Information.

Acknowledgement

The authors gratefully acknowledge financial support from the National Science Foundation (NSF) through CSSI Elements grant OAC-1835144 and CSSI Frameworks Grant OAC-2209717. A.S., M.M. and K.M.M. are grateful to the Department of Chemistry and Biochemistry and high-performance computer center (iCER HPCC) at Michigan State University. This work also used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by the National Science Foundation (Grant No. ACI-1053575, resources at the San Diego Supercomputer Center through award TG-CHE130010 to A.W.G.).

Supporting Information Available

Input files with Cartesian coordinates of the starting geometries for geometry optimizations with QUICK–Legacy, QUICK using DL–FIND, geomeTRIC, and ASE reported in this work (ZIP). Table summarizing the results for geometry optimization of the Baker test set using QUICK and DL-FIND with different levels of theory and basis sets (PDF).

References

(1) Schlegel, H. B. Exploring potential energy surfaces for chemical reactions: An overview of some practical methods. *Journal of Computational Chemistry* **2003**, *24*, 1514–1527.

(2) Swart, M.; Matthias Bickelhaupt, F. Optimization of strong and weak coordinates. *International Journal of Quantum Chemistry* **2006**, *106*, 2536–2544.

(3) Schlegel, H. B. Geometry optimization. *WIREs Computational Molecular Science* **2011**, *1*, 790–809.

(4) Kass, R. E.; Dennis, J. E.; Schnabel, R. B. Numerical Methods for Unconstrained Optimization and Nonlinear Equations. *Journal of the American Statistical Association* **1985**, *80*, 247–248.

(5) Scales, L. E. *Introduction to Non-Linear Optimization*; Springer-Verlag, 1985.

(6) Maeda, S.; Ohno, K.; Morokuma, K. Automated global mapping of minimal energy points on seams of crossing by the anharmonic downward distortion following method: A case study of H_2CO. *Journal of Physical Chemistry A* **2009**, *113*, 1704–1710.

(7) Madsen, S.; Jensen, F. Locating seam minima for macromolecular systems. *Theoretical Chemistry Accounts* **2009**, *123*, 477–485.

(8) Dick, B. Gradient projection method for constraint optimization and relaxed energy paths on conical intersection spaces and potential energy surfaces. *Journal of Chemical Theory and Computation* **2009**, *5*, 116–125.

(9) E, W.; Vanden-Eijnden, E. Transition-path theory and path-finding algorithms for the study of rare events. *Annual Review of Physical Chemistry* **2010**, *61*, 391–420.

(10) Schlegel, H. B. Optimization of equilibrium geometries and transition structures. *Journal of Computational Chemistry* **1982**, *3*, 214–218.

(11) Baker, J. Geometry optimization in Cartesian coordinates: Constrained optimization. *Journal of Computational Chemistry* **1992**, *13*, 240–253.

(12) Head, J. D. Partial optimization of large molecules and clusters. *Journal of Computational Chemistry* **1990**, *11*, 67–75.

(13) Pulay, P.; Fogarasi, G. Geometry optimization in redundant internal coordinates. *The Journal of Chemical Physics* **1992**, *96*, 2856–2860.

(14) Baker, J.; Kinghorn, D.; Pulay, P. Geometry optimization in delocalized internal coordinates: An efficient quadratically scaling algorithm for large molecules. *Journal of Chemical Physics* **1999**, *110*, 4986–4991.

(15) Pople, J. A.; Krishnan, R.; Schlegel, H. B.; Binkley, J. S. Derivative studies in Hartree-Fock and Møller-Plesset theories. *International Journal of Quantum Chemistry* **1979**, *16*, 225–241.

(16) Murtagh, B. A.; Sargent, R. W. H. Computational experience with quadratically convergent minimisation methods. *The Computer Journal* **1970**, *13*, 185–194.

(17) Goldfarb, D. A Family of Variable-Metric Methods Derived by Variational Means. *Mathematics of Computation* **1970**, *24*, 23–26.

(18) Broyden, C. G. The Convergence of a Class of Double-rank Minimization Algorithms 1. General Considerations. *IMA Journal of Applied Mathematics* **1970**, *6*, 76–90.

(19) Fletcher, R. A new approach to variable metric algorithms. *The Computer Journal* **1970**, *13*, 317–322.

(20) Baker, J. Techniques for geometry optimization: A comparison of Cartesian and natural internal coordinates. *Journal of Computational Chemistry* **1993**, *14*, 1085–1100.

(21) Bakken, V.; Helgaker, T. The efficient optimization of molecular geometries using redundant internal coordinates. *Journal of Chemical Physics* **2002**, *117*, 9160–9174.

(22) Baker, J.; Kessi, A.; Delley, B. The generation and use of delocalized internal coordinates in geometry optimization. *The Journal of Chemical Physics* **1996**, *105*, 192–212.

(23) Lindh, R.; Bernhardsson, A.; Karlström, G.; Åke Malmqvist, P. On the use of a Hessian model function in molecular geometry optimizations. *Chemical Physics Letters* **1995**, *241*, 423–428.

(24) Denzel, A.; Kästner, J. Gaussian process regression for geometry optimization. *The Journal of Chemical Physics* **2018**, *148*, 094114.

(25) Raggi, G.; Galván, I. F.; Ritterhoff, C. L.; Vacher, M.; Lindh, R. Restricted-Variance Molecular Geometry Optimization Based on Gradient-Enhanced Kriging. *Journal of Chemical Theory and Computation* **2020**, *16*, 3989–4001.

(26) Denzel, A.; Kästner, J. Gaussian Process Regression for Transition State Search. *Journal of Chemical Theory and Computation* **2018**, *14*, 5777–5786.

(27) Denzel, A.; Haasdonk, B.; Kästner, J. Gaussian Process Regression for Minimum Energy Path Optimization and Transition State Search. *The Journal of Physical Chemistry A* **2019**, *123*, 9600–9611.

(28) Meyer, R.; Hauser, A. W. Geometry optimization using Gaussian process regression in internal coordinate systems. *The Journal of Chemical Physics* **2020**, *152*, 084112.

(29) Teng, C.; Huang, D.; Bao, J. L. A spur to molecular geometry optimization: Gradient-enhanced universal kriging with on-the-fly adaptive ab initio prior mean functions in curvilinear coordinates. *The Journal of Chemical Physics* **2023**, *158*, 024112.

(30) Manathunga, M.; Shajan, A.; Giese, T. J.; Cruzeiro, V. W. D.; Smith, J.; Miao, Y.; He, X.; Ayers, K.; Brothers, E.; Götz, A. W. et al. QUICK-22.03, University of California San Diego, CA and Michigan State University, East Lansing, MI. 2022.

(31) Miao, Y.; Merz, K. M., Jr. Acceleration of High Angular Momentum Electron Repulsion Integrals and Integral Derivatives on Graphics Processing Units. *Journal of Chemical Theory and Computation* **2015**, *11*, 1449–1462.

(32) Manathunga, M.; Jin, C.; Cruzeiro, V. W. D.; Miao, Y.; Mu, D.; Arumugam, K.; Keipert, K.; Aktulga, H. M.; Merz, K. M., Jr.; Götz, A. W. Harnessing the Power of

Multi-GPU Acceleration into the Quantum Interaction Computational Kernel Program. *Journal of Chemical Theory and Computation* **2021**, *17*, 3955–3966.

(33) Manathunga, M.; Miao, Y.; Mu, D.; Götz, A. W.; Merz, K. M., Jr. Parallel Implementation of Density Functional Theory Methods in the Quantum Interaction Computational Kernel Program. *Journal of Chemical Theory and Computation* **2020**, *16*, 4315–4326.

(34) Cruzeiro, V. W. D.; Manathunga, M.; Merz, K. M., Jr.; Götz, A. W. Open-Source Multi-GPU-Accelerated QM/MM Simulations with AMBER and QUICK. *Journal of Chemical Information and Modeling* **2021**, *61*, 2109–2115.

(35) Manathunga, M.; Aktulga, H. M.; Götz, A. W.; Merz, K. M., Jr. Quantum Mechanics/Molecular Mechanics Simulations on NVIDIA and AMD Graphics Processing Units. *Journal of Chemical Information and Modeling* **2023**, *63*, 711–717.

(36) Nocedal, J. Updating Quasi-Newton Matrices with Limited Storage. *Mathematics of Computation* **1980**, *35*, 773–782.

(37) Liu, D. C.; Nocedal, J. On the limited memory BFGS method for large scale optimization. *Mathematical Programming* **1989**, *45*, 503–528.

(38) Kästner, J.; Carr, J. M.; Keal, T. W.; Thiel, W.; Wander, A.; Sherwood, P. DL-FIND: An open-source geometry optimizer for atomistic simulations. *Journal of Physical Chemistry A* **2009**, *113*, 11856–11865.

(39) Matérn, B. *Spatial variation*; Springer Science & Business Media, 2013; Vol. 36.

(40) Born, D.; Kästner, J. Geometry Optimization in Internal Coordinates Based on Gaussian Process Regression: Comparison of Two Approaches. *Journal of Chemical Theory and Computation* **2021**, *17*, 5955–5967.

(41) Wang, L.-P.; Song, C. Geometry optimization made simple with translation and rotation coordinates. *The Journal of Chemical Physics* **2016**, *144*, 214108.

(42) Schlegel, H. B. Estimating the hessian for gradient-type geometry optimizations. *Theoretica Chimica Acta* **1984**, *66*, 333–340.

(43) Larsen, A. H.; Mortensen, J. J.; Blomqvist, J.; Castelli, I. E.; Christensen, R.; Dułak, M.; Friis, J.; Groves, M. N.; Hammer, B.; Hargus, C. et al. The atomic simulation environment - A Python library for working with atoms. *Journal of Physics Condensed Matter* **2017**, *29*, 273002.

(44) Birkholz, A. B.; Schlegel, H. B. Exploration of some refinements to geometry optimization methods. *Theoretical Chemistry Accounts* **2016**, *135*, 84.

(45) Hermes, E. D.; Sargsyan, K.; Najm, H. N.; Zádor, J. Sella, an Open-Source Automation-Friendly Molecular Saddle Point Optimizer. *Journal of Chemical Theory and Computation* **2022**, *18*, 6974–6988.

(46) Hermes, E. D.; Sargsyan, K.; Najm, H. N.; Zádor, J. Geometry optimization speedup through a geodesic approach to internal coordinates. *The Journal of Chemical Physics* **2021**, *155*, 094105.

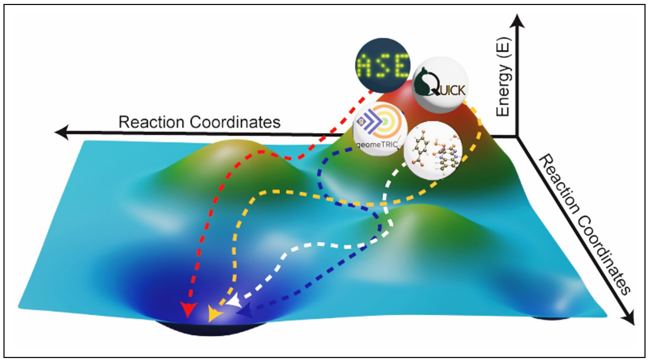
(47) Fischer, T. H.; Almlof, J. General methods for geometry and wave function optimization. *The Journal of Physical Chemistry* **1992**, *96*, 9768–9774.

(48) Baker, J. An algorithm for the location of transition states. *Journal of Computational Chemistry* **1986**, *7*, 385–395.

(49) Császár, P.; Pulay, P. Geometry optimization by direct inversion in the iterative subspace. *Journal of Molecular Structure* **1984**, *114*, 31–34.

(50) Lindh, R.; Bernhardsson, A.; Schütz, M. Force-constant weighted redundant coordinates in molecular geometry optimizations. *Chemical physics letters* **1999**, *303*, 567–575.

TOC Graphic


ASE
QUICK
geomeTRIC
Reaction Coordinates
Energy (E)
Reaction Coordinates