JCTC Journal of Chemical Theory and Computation

Article

Computational Implementation of Nudged Elastic Band, Rigid Rotation, and Corresponding Force Optimization

Henry Herbol, James Stevenson, and Paulette Clancy

J. Chem. Theory Comput., **Just Accepted Manuscript** • DOI: 10.1021/acs.jctc.7b00360 • Publication Date (Web): 16 Jun 2017

ACS Publications

Journal of Chemical Theory and Computation is published by the American Chemical Society. 1155 Sixteenth Street N.W., Washington, DC 20036

Computational Implementation of Nudged Elastic Band, Rigid Rotation, and Corresponding Force Optimization

Henry Herbol,*,† James Stevenson,‡ and Paulette Clancy‡

†*Dept. of Materials Science and Engineering, Cornell University*
‡*Robert F. Smith School of Chemical and Biomolecular Engineering, Cornell University*

E-mail: hch54@cornell.edu

Abstract

The Nudged Elastic Band (NEB) algorithm is the leading method of calculating transition states in chemical systems. However, the current literature lacks adequate guidance for users wishing to implement a key part of NEB, namely, the optimization method. Here, we provide details of this implementation for the following six gradient descent algorithms: Steepest Descent, Quick-Min Verlet, FIRE, Conjugate Gradient, Broyden-Fletcher-Goldfarb-Shanno (BFGS) and Limited-memory BFGS (LBFGS). We also construct and implement a new, accelerated backtracking line search method in concert with a partial Procrustes superimposition to improve upon existing methods. Validation is achieved through benchmark calculations of two test cases, the isomerization of CNX and BOX (where X$\in \{H, Li, Na\}$) and the study of a conformational change within an alanine dipeptide. We also make direct comparisons to the well established codebase known as the Atomic Simulation Environment.

1 Introduction

Computational approaches that offer the ability to calculate reaction barriers and hence shed light on the underlying molecular mechanisms are powerful tools for computational material scientists. Among the methods available to provide reaction rates and barriers, the most popular technique used to calculate the Minimum Energy Pathway (MEP) of a reaction coordinate is known as the "Nudged Elastic Band" method (NEB). NEB was first described in a 1994 paper by Greg Mills and Hannes Jónsson in a study on the adsorption of H_2 on the surface of Cu(110).[1] However, it was not until 1998 that the method we now know as NEB was named and described by Hannes Jónsson, Greg Mills, and Karsten W. Jacobsen.[2] Later, in 2000, Jónsson and Henkelman published two papers updating the method. The first improved the base algorithm, while the latter developed the now well-used Climbing Image (CI-NEB) variation.[3,4] In 2002, these two authors wrote one of the best reviews on the subject as a chapter in a textbook.[5] In 2008, Sheppard and Henkelman wrote a review of gradient-based optimization methods applicable to NEB, and provided tests which showed the ones that worked best for the applications they considered. This proved to be the most concise review of the NEB method and optimization so far.

At this point, it may seem as though this field of study has been well established. And indeed, the combined efforts of Henkelman, Jónsson, Mills, Jacobsen, and Sheppard have laid the foundation of a method that, to this day, offers the forefront of reaction pathway calculations. Unfortunately, the literature on the

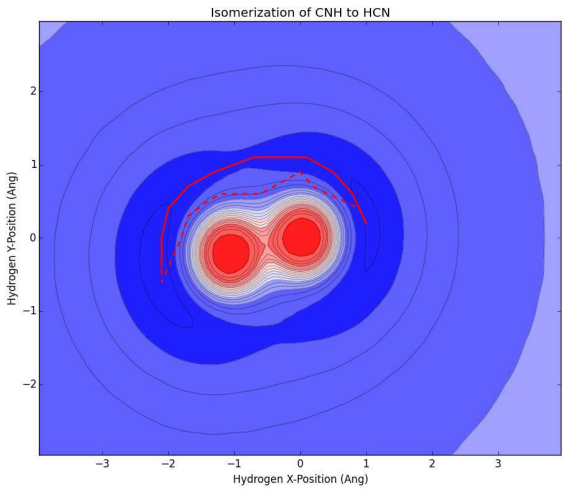


Figure 1: The motion of hydrogen during the isomerization of CNH to HCN, optimized using Nudged Elastic Band.

subject does not lend itself to an easy application of the method due to a lack of elaboration on the merging of NEB and optimization. That is to say, to date, no review has laid out, in a programmatic manner, how to implement gradient-based descent methods. For instance, a literature search by readers looking for a clear and concise description of applying optimization algorithms, such as the Broyden-Fletcher-Goldfarb-Shanno (BFGS) method, will prove unfruitful. Further, absent the existence of such a background, this area is prone to poorly constructed results, such as the paper by Quapp and Bofill that was rebutted by Henkelman and Sheppard in 2011, in which the error lay within their optimization algorithm.[6]

This review only briefly covers the NEB method itself, since we believe that Henkelman *et al.* have more than adequately covered this over the course of recent decades. Instead, our aim is to provide a series of algorithms by which gradient-based optimization may be implemented within a code to interface with existing computational methods. In addition, we discuss accurate and computationally efficient improvements to the NEB method such as the use of the Procrustes method to remove rotation across the reaction coordinate, as well as the effects of line search methods on some Quasi-Newton methods. These methods have been written for implementation in conjunction with electronic structure codes such as Orca[7] and Gaussian09[8]. However, they are broadly described so as to be applicable to any code. We validate this new method through benchmark comparisons for two test cases. The first, simpler case calculates the MEP for the isomerization of CNX and BOX (where X$\in \{H, Li, Na\}$). The second, more complex test case involves the energy barrier associated with a conformational change in alanine dipeptide. These test cases are compared to an already well-established codebase, the Atomic Simulation Environment (ASE).[9] Our focus here is placed on reducing the number of force calculation steps in NEB using quantum mechanical-based methods, since each step comes at a nontrivial cost in computer time.

2 Notation Used

When discussing the NEB method, we will use the term "image" to describe an atomic configuration of a reaction coordinate. That is, a reaction coordinate is made up of N images, each describing the positions of atoms as they evolve during a reaction. Further, we will use the notation R_i to represent the i^{th} image of a reaction coordinate.
Throughout this review, the bra-ket notation is used. This notation is:

$$\begin{aligned} |x\rangle &= \text{column vector} \\ \langle x| &= \text{row vector} \\ \langle x|x\rangle &= \text{scalar} \\ |x\rangle\langle x| &= \text{matrix} \end{aligned}$$

In either the bra or ket, we will assume that the vectors are represented as 1-D arrays. Thus, if referring to the forces on atoms in the i^{th} image of an NEB system as $|F_i\rangle$, this implies:

$$\begin{aligned} &|F_i\rangle = \\ &\langle F_{i,0,x}, F_{i,0,y}, F_{i,0,z}, F_{i,1,x}, F_{i,1,y}, F_{i,1,z}, \\ &..., F_{i,n,x}, F_{i,n,y}, F_{i,n,z}\rangle \end{aligned} \quad (1)$$

where the subscripts indicate the image, atom, and component of the force respectively.

At times during optimization it is computationally beneficial to accumulate all variables into a single array. Further, it is useful to identify this array by its iteration in the optimization cycles. As such, we define $|F^i\rangle$ to be all the forces on all the atoms for all the images of iteration i as:

$$|F^i\rangle = \langle |F_0^i\rangle, |F_1^i\rangle, |F_2^i\rangle, ..., |F_n^i\rangle\rangle \quad (2)$$

Within the Nudged Elastic Band Method section of this paper, R_i is a 1-D array holding atomic coordinates and $-\nabla V(R_i)$, which is the negative of the potential gradient, defines the real forces on atoms. V_i is a scalar denoting the energy of the i$^{\text{th}}$ image. Finally, within the optimization section we use the following notation:

$$|x^i\rangle = \langle R_0^i, R_1^i, ..., R_n^i\rangle \quad (3)$$
$$|g^i\rangle = \langle \nabla V(R_0^i), \nabla V(R_1^i), ..., \nabla V(R_n^i)\rangle \quad (4)$$
$$|F^i\rangle = -|g^i\rangle \quad (5)$$

3 Nudged Elastic Band Method

The NEB method uses a sequence of atomic positions (R_i), called "images" or "beads," to describe a reaction coordinate. This reaction coordinate is generated either by hand, or linear/spline interpolation, and is thus inaccurate by nature (hence the need for NEB optimization). Each atom in each image is joined to the equivalent atom in the two adjacent images by harmonic constraints with a harmonic constant, k_s, simulating an elastic band.

The "nudging" referred to in NEB involves the method of adjusting the spring forces and the atomic forces such that they are orthogonal and cannot interfere with one another. Taking Fig. 2 as an example, the second image would ideally relax towards the first; however, by nudging with intervening springs and only taking the perpendicular real force, we can maintain the "distance" on the reaction coordinate. This allows for a relaxation of the entire band towards the MEP. The overall force that acts on each atom in NEB is defined as the sum of the spring force projected onto the path ($F_{||}^s$)

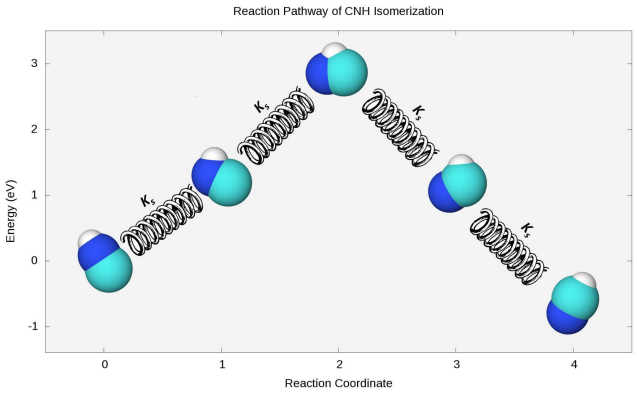


Figure 2: The energy barrier for a 5-bead CNH isomerization elastic band for NEB. Note that connections between beads indicate the presence of springs between them, defined by a spring constant k_s. Carbon is shown in cyan, nitrogen as dark blue, and hydrogen as white.

and the atomic force perpendicular to the path ($F_\perp^b = -\nabla V(R_i)_\perp$):

$$F_i = F_{||}^s + F_\perp^b = F_{||}^s - \nabla V(R_i)_\perp \quad (6)$$

where $\nabla V(R_i)$ is the potential energy gradient, or, simply put, the (negative) force on the atomic positions, R_i. Such atomic forces are standard outputs when using either a Density Functional Theory (DFT) or a Molecular Dynamics (MD) code. To calculate the perpendicular part of these forces to the band, one can simply subtract out the parallel part:

$$\begin{aligned}\nabla V(R_i)_\perp &= \nabla V(R_i) - \nabla V(R_i)_{||} \quad (7)\\ &= \nabla V(R_i) - \nabla V(R_i)\cdot\widehat{\tau}_i\end{aligned}$$

where $\widehat{\tau}_i$ describes the normalized tangential direction of the spring at image i. Minimizing this force allows the method to locate the MEP closest to the initial guessed states. In 2000, an improvement was introduced by Henkelman and Jónsson in which the derivation of $\widehat{\tau}_i$ was chosen to minimize kinks along the elastic band typically seen in NEB calculations (see Fig. 1 of Henkelman and Jónsson[3]). Instead of choosing a linear tangent between images R_{i-1} and R_{i+1}, as was the original method, the tangent was defined to favor the highest energy ($V_{i+1/i-1}$) im-

age neighboring R_i:

$$\tau_i = \begin{cases} R_{i+1} - R_i & \text{if} \quad V_{i+1} > V_i > V_{i-1} \\ R_i - R_{i-1} & \text{if} \quad V_{i+1} < V_i < V_{i-1} \end{cases} \quad (8)$$

Recall that, as R_i is a 1-D vector, so too is τ_i. In the case of an extremum, Henkelman and Jónsson used a weighted method to produce a smooth transition between tangents. If $V_{i+1} > V_{i-1}$, then:

$$\tau_i = (R_{i+1} - R_i)\Delta V_i^{max} + (R_i - R_{i-1})\Delta V_i^{min} \quad (9)$$

else, if $V_{i+1} < V_{i-1}$, then:

$$\tau_i = (R_{i+1} - R_i)\Delta V_i^{min} + (R_i - R_{i-1})\Delta V_i^{max} \quad (10)$$

where

$$\begin{aligned} \Delta V_i^{max} &= max(|V_{i+1} - V_i|, |V_{i-1} - V_i|) \\ \Delta V_i^{min} &= min(|V_{i+1} - V_i|, |V_{i-1} - V_i|) \end{aligned} \quad (11)$$

The parallel spring force is defined by:

$$F_{||}^s = k_s(|R_{i+1} - R_i| - |R_{i-1} - R_i|)\widehat{\tau}_i \quad (12)$$

In this equation, $|x|$ defines the magnitude of the vector x and $\widehat{\tau}_i$ the normalized tangent τ_i. To improve on this even further, Climbing Image NEB (CI-NEB) can be implemented on the highest energy image $R_{i,max}$ (typically chosen after some NEB iterations are run). This can be accomplished by, instead, calculating the force on $R_{i,max}$ as follows[4]:

$$F_{i,max} = F^b - 2F^b \cdot \widehat{\tau}_{i,max} \cdot \widehat{\tau}_{i,max} \quad (13)$$

where F^b is the real force on the atoms (neither parallel nor perpendicular). Note, the springs do not affect this image because $R_{i,max}$ was chosen to converge towards the transition state.

A final update by Jónsson *et al.* recommended that the method should minimize the translation and rotation of the system during the optimization.[2] In that spirit, our new implementation uses the Kabsch algorithm, also known as a partial Procrustes superimposition,[10] to take a reference image (that being the first in our band), and propagate translations and rigid rotations down the band (excluding inversions) in order to minimize translation and rotation between images. It is important to note that this is only applicable to non-periodic systems.

Algorithm 1 Nudged Elastic Band

$|x^i\rangle \leftarrow$ input_args()
$|g^i\rangle, |V^i\rangle \leftarrow$DFT_Calc($|x^i\rangle$)
for j in $range(1, N_{images} - 1)$ **do**
  if $V_{j+1}^i > V_j^i > V_{j-1}^i$ **then**
    $|\tau_j^i\rangle = R_{j+1}^i - R_j^i$
  else if $V_{j+1}^i < V_j^i < V_{j-1}^i$ **then**
    $|\tau_j^i\rangle = R_j^i - R_{j-1}^i$
  else
    $\Delta V_j^{i,max} = max(|V_{j+1}^i - V_j^i|, |V_{j-1}^i - V_j^i|)$
    $\Delta V_j^{i,min} = min(|V_{j+1}^i - V_j^i|, |V_{j-1}^i - V_j^i|)$
    $dR_{left} = R_{j+1}^i - R_j^i$
    $dR_{right} = R_j^i - R_{j-1}^i$
    if $V_{j+1}^i > V_{j-1}^i$ **then**
      $|\tau_j^i\rangle = dR_{left}\Delta V_j^{i,max} + dR_{right}\Delta V_j^{i,min}$
    else
      $|\tau_j^i\rangle = dR_{left}\Delta V_j^{i,min} + dR_{right}\Delta V_j^{i,max}$
    end if
  end if
  $|\widehat{\tau}_j^i\rangle = \frac{|\tau_j^i\rangle}{\sqrt{\langle\tau_j^i|\tau_j^i\rangle}}$
  $|F_{j,||}^{i,s}\rangle = k_s(|R_{j+1}^i - R_j^i| - |R_{j-1}^i - R_j^i|)|\widehat{\tau}_j^i\rangle$
  $\nabla V(R_j^i)_\perp = \nabla V(R_j^i) - \nabla V(R_j^i) \cdot |\widehat{\tau}_j^i\rangle|\widehat{\tau}_j^i\rangle$
  if $i > N_{CI-NEB}$ **and** $V_j^i = max(V^i)$ **then**
    $|F_j^i\rangle = -|g_j^i\rangle + 2.0 * \langle g_j^i|\widehat{\tau}_j^i\rangle|\widehat{\tau}_j^i\rangle$
  else
    $|F_j^i\rangle = |F_{j,||}^{i,s}\rangle - \nabla V(R_j^i)_\perp$
  end if
end for
return $|F^i\rangle$

4 Rigid Rotation

Given a starting reaction coordinate in which the goal is to optimize towards the MEP, erroneous iterations in an optimizer can occur due to a rigid rotation and/or translation between sequential images. For instance, taking the example of Fig. 2, imagine if, instead, Fig. 3 were the starting reaction coordinate. The optimizer now has to remove this excess motion, leading to extra steps taken during force minimization,

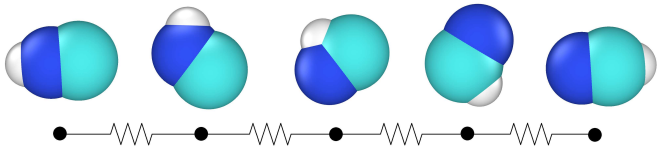


Figure 3: A 5-bead CNH isomerization elastic band for NEB illustrating excessive rigid rotation. Connections indicate the presence of a spring. Color key for atoms as in Fig. 2.

where a simple rotation could be performed to remove the excess motion instead. There are many ways to minimize these rigid rotations and translations; however, a robust, hands-off method would be optimal. A naïve approach involves specifying a subset of atoms in the system to align to an arbitrary coordinate system. This is easily accomplished by choosing three atoms: one for the origin, one to lie along the positive x-axis, and the final to lie on the positive $x-y$ plane. This method would require this subset as an additional input. However, incorrectly choosing atoms could end up exacerbating the rotation/translation problem even more.

A more robust method involves employing a partial Procrustes superimposition applied alongside translations, in which the goal is to find a rotation matrix, A, such that $A^TA = I$, and the Frobenius norm of $(R_{i+1}A - R_i)$ between consecutive images is minimized. We begin with a translation B such that translating R_i by B positions the center of geometry of image R_i at the origin. Understanding that minimizing the Frobenius norm also minimizes its square, we obtain the following by definition:

$$\begin{aligned} &||R_{i+1}A - R_i||^2 = \\ &= Tr\left[(R_{i+1}A - R_i)^T (R_{i+1}A - R_i)\right] \\ &= Tr\left[\left(A^T R_{i+1}^T - R_i^T\right)(R_{i+1}A - R_i)\right] \\ &= Tr\left[A^T R_{i+1}^T R_{i+1}A - R_i^T R_{i+1}A \right. \\ &\quad \left. -A^T R_{i+1}^T R_i + R_i^T R_i\right] \\ &= Tr\left(A^T R_{i+1}^T R_{i+1}A\right) + Tr\left(R_i^T R_i\right) \\ &\quad - Tr\left(R_i^T R_{i+1}A\right) - Tr\left(A^T R_{i+1}^T R_i\right) \end{aligned} \tag{14}$$

As $Tr(R_{i+1}R_i) = Tr(R_iR_{i+1})$ and $Tr(R_{i+1}) = Tr(R_{i+1}^T)$, we can reorder the relationship above to obtain the following:

$$\begin{aligned} &||R_{i+1}A - R_i||^2 = \\ &= Tr\left(AA^T R_{i+1}^T R_{i+1}\right) + Tr\left(R_i^T R_i\right) \\ &\quad - Tr\left(R_i^T R_{i+1}A\right) - Tr\left(AR_{i+1}R_i^T\right) \\ &= Tr\left(R_{i+1}^T R_{i+1}\right) + Tr\left(R_i^T R_i\right) \\ &\quad - 2Tr\left(R_i^T R_{i+1}A\right) \\ &= ||R_{i+1}||^2 + ||R_i||^2 - 2Tr\left(R_i^T R_{i+1}A\right) \end{aligned} \tag{15}$$

Therefore, since the first two terms are positive, if we want to minimize this expression, we must maximize $Tr\left(R_i^T R_{i+1}A\right)$. From a Singular Value Decomposition (SVD), we get the following:

$$R_i^T R_{i+1} = UWV^T \tag{16}$$

where U and V^T are unitary matrices and W is a diagonalized matrix describing the "weight". Thus, we must maximize $Tr\left(UWV^TA\right)$. As we are seeking out a rotation matrix, and not one to scale coordinates, the "weight" matrix W is ignored by setting $W = I$. We must now maximize:

$$Tr\left(UV^TA\right) \tag{17}$$

As each matrix within this trace is unitary, and subsequently orthonormal, the magnitude of each column must then be unity by definition. The trace is then maximized when this product equals the identity matrix:

$$max\left(Tr\left(UV^TA\right)\right) \implies UV^TA = I \tag{18}$$

Solving:

$$A = VU^T \tag{19}$$

Therefore, starting with $SVD(R_i^T R_{i+1}) = UWV^T$, we can calculate the transpose of V^T and U. From that, the rotation matrix is simply $A = VU^T$.

For computational efficiency, observe the SVD of $R_{i+1}^T R_i$:

$$\begin{aligned} R_{i+1}^T R_i = (R_i^T R_{i+1})^T = UWV^T \\ \implies R_i^T R_{i+1} = VW^TU^T \end{aligned} \tag{20}$$

In this case, if we maximize the following:

$$\begin{aligned} & trace\left(VW^TU^TA\right) \\ \implies & VU^TA = I \\ \implies & A = UV^T \end{aligned} \tag{21}$$

we never need to calculate V itself, and can directly use the transpose. Simply put, given two consecutive images R_i and R_{i+1}, the rotation matrix A that best "matches" R_{i+1} onto R_i can be found *via* the following two-step calculation:

$$\begin{aligned} SVD\left[R_{i+1}^TR_i\right] &= UWV^T \\ \implies A &= UV^T \end{aligned} \tag{22}$$

One small adjustment must be made to the final rotation matrix. This is due to the fact that A can include inversions which, in a reaction coordinate, is not possible. Simply adding a check for a negative determinant and a sign flip of the last column of U corrects for this:

$$\begin{bmatrix} -0.45 & 0.0 & \mathbf{0.89} \\ -0.89 & 0.0 & \mathbf{-0.45} \\ 0.0 & 1.0 & \mathbf{0.0} \end{bmatrix} \rightarrow \begin{bmatrix} -0.45 & 0.0 & \mathbf{-0.89} \\ -0.89 & 0.0 & \mathbf{0.45} \\ 0.0 & 1.0 & \mathbf{0.0} \end{bmatrix}$$

Figure 4: An example U matrix with a negative determinant, implying inversion in the final rotation matrix. Flipping the sign of the last column removes this inversion.

Therefore, recalling that R_i is the i^{th} image of a reaction coordinate, the final algorithm describing how to robustly remove rotation and translation is shown in Alg. 2.

We return the list of applied rotation matrices because, during optimization, certain stored variables would also need to have this change of coordinate system applied. Thus, a generalized function dubbed "fit_rigid()" is developed (shown in Alg. 3).

Algorithm 2 Procrustes Superimposition

$B = COG(R_0)$ ▷ Center of Geometry
$R_0 = R_0 - B$
for i in $range(0, N_{images} - 1)$ **do**
  $B = COG(R_{i+1})$
  $R_{i+1} = R_{i+1} - B$
  $UWV^T = SVD\left[R_{i+1}^TR_i\right]$
  $A_i = UV^T$
  if $det(A_i) < 0$ **then**
    $U[:, -1] = U[:, -1] * -1$
    $A_i = UV^T$
  end if
  $R_{i+1} = R_{i+1}A_i$
end for
return A

Algorithm 3 Rigid Rotation of Coordinate System

$R, [|v_1\rangle, |v_2\rangle, ..., |v_n\rangle], H \leftarrow$ input_args()
$A =$ procrustes(R)
$G = block\_diag(A)$ ▷ Define full rotation
for i in $range(1, n)$ **do**
  $\langle v_i| = \langle v_i|G$
end for
$H = GHG^T$

5 Optimization

In order to converge a reaction pathway towards the MEP, we need to have some quantitative value that we are minimizing. Potential objective functions at hand are the energy and the force. These two metrics are inherently linked

by definition; however, it stands to reason that minimizing E_{max} does not necessarily minimize F_{RMS}. This provides the chance to hide a potential barrier between images of the pathway. Further, in practice, we find that, at times, the MEP increases with decreasing F_{RMS}, exemplifying the flaw of using E_{max} as an objective function. As such, the best choice is to minimize either F_{RMS} or F_{max}. In this work, we choose to use F_{RMS} as our objective function.

There exist a number of algorithms to minimize the NEB. However, the most commonly used ones are (1) the steepest descent (SD)[11], (2) the Fast Inertial Relaxation Engine (FIRE)[11,12] algorithm, (3) the Quick Min (QM)[2,11,13] algorithm, (4) the Conjugate Gradient (CG)[11,14], and (5) the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithms[11,15–18]. Of these, CG and BFGS typically converge both the fastest and most reliably, with BFGS on average converging faster than CG.[11] Finally, the BFGS algorithm requires the inverse Hessian to be represented in full form and, as such, can be memory-intensive on larger systems ($N > 90$ atoms or so). An update was made to reduce this memory requirement in the so-called Limited-BFGS (LBFGS).[11,19–21]

5.1 Steepest Descent

The Steepest Descent method is the application of Newton's Method, lending itself to be a reliable optimizer if the inverse Hessian is available. Newton's Method works by first observing the standard quadratic form in n dimensions:

$$f = f_0 + \sum_{i=1}^{n} a_i x_i + \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} B_{ij} x_i x_j \tag{23}$$

where f_0 is the function evaluated at a point x_0, a_i is some constant, and B_{ij} is a matrix of constants. Using the aforementioned bra-ket notation, we see that:

$$f = f_0 + \langle a|x\rangle + \frac{1}{2}\langle x|B|x\rangle \tag{24}$$

where $\langle x|$ denotes a row vector and $|x\rangle$ denotes a column vector. This is useful because we can see at a glance that $\langle x|x\rangle$ is a scalar, whereas $|x\rangle\langle x|$ is a matrix. To minimize f, we solve for cases in which the gradient is zero:

$$\begin{aligned}\frac{df}{dx} &= \frac{d}{dx}\left(f_0 + \sum_{i=1}^{n} a_i x_i + \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} B_{ij} x_i x_j\right) \\ &= 0 + \sum_{i=1}^{n} a_i + \sum_{i=1}^{n}\sum_{j=1}^{n} B_{ij} x_i \\ \implies |g\rangle &= |a\rangle + B|x\rangle\end{aligned} \tag{25}$$

where B is simply a matrix of constants. Letting the subscript min denote the frame of reference for the minimum, and keeping in mind that $|g_{min}\rangle = 0$ at the minimum, we can solve for a step towards the minimum as:

$$\begin{aligned}|g_{min}\rangle - |g\rangle &= B|x_{min}\rangle - B|x\rangle \\ \implies |x_{min}\rangle - |x\rangle &= -B^{-1}|g\rangle\end{aligned} \tag{26}$$

Defining this as the step direction, we find that $|s\rangle = |x_{min}\rangle - |x\rangle = -B^{-1}|g\rangle$ is the direction in which the system must move in order to minimize. Before continuing, note that $\frac{d^2 f}{dx^2} = B$ is simply the Hessian and thus, $B^{-1} = H$ describes the inverse Hessian. This notation of B and H being the Hessian and inverse Hessian respectively is widespread, and thus maintained for clarity here.

This gives us the Quasi-Newton family of methods, in which the step to take towards the minimum (on iteration i) is:

$$\begin{aligned}|s^i\rangle &= -H^i|g^i\rangle = H^i|F^i\rangle \\ |x^{i+1}\rangle &= |x^i\rangle + \alpha|s^i\rangle\end{aligned} \tag{27}$$

where $|F^i\rangle$ is the force on the atoms (which by definition is equal to the negative of the potential gradient). α describes a step size to take towards the minimum. In theory, a line search is done to find the optimal step size to take along this direction; however, in practice most line search methods involve multiple calculations. When performing complex computational steps, such as those in DFT, this quickly becomes unfeasible. A good compromise is the backtracking line search method, with an even better compromise being an accelerated version. The backtracking line search method in-

volves taking a step, checking it against some criteria, and either reducing α by some γ, or leaving it as is. In the case of an accelerated version, if no backtracking is done after N_{back} iterations, then α is increased by $\frac{1}{\gamma}$. In more advanced methods, a reset can be added when backtracking, so as to forego previous, erroneously made steps. This accelerated backtracking line search can be seen in Alg. 4, in which the criterion for backtracking has been defined as the difference in RMS forces between subsequent iterations being larger than some given ϵ (10^{-3}). Thus, we arrive at the completed steepest descent algorithm Alg. 5 (where N_{max} is the maximum number of iterations the optimizer will take).

Algorithm 4 Accelerated Backtracking Line Search

$F^i_{RMS}, F^{i-1}_{RMS} \leftarrow$input_args()
$chk = \frac{F^i_{RMS} - F^{i-1}_{RMS}}{|F^i_{RMS} + F^{i-1}_{RMS}|}$
skip=False
if $chk > \epsilon$ **then**
  $\alpha = \alpha * \gamma$ ▷ Slow α
  skip=True
  $N_{back} = N_0$ ▷ Reset N_{back}
else
  $N_{back} = N_{back} - 1$
  if $N_{back} < 0$ **then**
    $N_{back} = N_0$
    if $\alpha < \alpha_0$ **then**
      $\alpha = \alpha_0$ ▷ Reset α
      skip=True
    else
      $\alpha = \frac{\alpha}{\gamma}$ ▷ Accelerate α
    end if
  end if
end if
return α, skip

It should be noted that, for every algorithm used, a maximum step size was set to prevent extreme steps from being taken. This was simply done by scaling all steps when the maximum was larger than some pre-defined size. Scaling was done to $|s^i\rangle$ prior to the scaling by α for all algorithms except for the conjugate gradient, which requires a memory of previous step

Algorithm 5 Steepest Descent

$|x^0\rangle \leftarrow$starting_guess()
$H^0 = I$ ▷ Identity matrix
for i in $range(0, N_{max})$ **do**
  $|x^i\rangle \leftarrow$Procrustes($|x^i\rangle$)
  $|F^i\rangle \leftarrow$NEB_Calc($|x^i\rangle$)
  $|s^i\rangle = H^0|F^i\rangle$
  if $i > 0$ **then**
    $\alpha \leftarrow$ backtrack($|F^i_{RMS}\rangle, |F^{i-1}_{RMS}\rangle$)
  end if
  $|s^i\rangle \leftarrow$ scale_by_max_step($|s^i\rangle$)
  $|x^{i+1}\rangle = |x^i\rangle + \alpha|s^i\rangle$
end for

sizes. In this case, α modified $|s^i\rangle$, was scaled, and was then saved for the next iteration. In all other methods, α did not modify $|s^i\rangle$ directly, but was used when calculating $|x^{i+1}\rangle$.

5.2 Quick-Min Verlet

The Quick-Min Verlet algorithm works by calculating velocities from a time step dt and acceleration, which can be found *via* Newton's Second Law as $|a\rangle = \frac{|F\rangle}{|M\rangle}$. As long as the velocity and force align, then, with each subsequent iteration, the atom will accelerate along the direction of the force. However, as soon as it overshoots its minimum, the velocity is set to zero, and the process repeats itself. Letting v^i define the velocities on the i^{th} iteration, and $|M\rangle$ the mass of each atom, we obtain Alg. 6. It should be mentioned that, in order to obtain reasonable units, the force units were converted as:

$$\begin{aligned} |F^i\rangle[=]\frac{Ha}{Ang} * c_1 \frac{\frac{kg\,m^2}{s^2}}{Ha} * c_2 \frac{Ang^2}{m^2} \\ |F^i\rangle[=]kg\frac{Ang}{s^2} * c_1 * c_2 \end{aligned} \tag{28}$$

(where c_1 is the unit conversion from Ha to J, and c_2 is the conversion from m^2 to Ang^2).

An upgrade to this method is the use of a damping factor. In that case, the acceleration is offset by a viscosity. This is achieved by re-

Algorithm 6 Quick-Min Verlet

1: $|x^0\rangle \leftarrow$starting_guess()
2: $|v^0\rangle = |0\rangle$
3: **for** i in $range(0, N_{max})$ **do**
4: fit_rigid($|x^i\rangle, |v^i\rangle$)
5: $|F^i\rangle \leftarrow$NEB_Calc($|x^i\rangle$)
6: $|v'^i\rangle = \begin{cases} |0\rangle & , \quad i = 0 \\ \frac{\langle v^{i-1}|F^i\rangle |F^i\rangle}{\langle F^i|F^i\rangle} & , \quad \langle v^{i-1}|F^i\rangle > 0 \\ |0\rangle & , \quad \langle v^{i-1}|F^i\rangle \leq 0 \end{cases}$
7: $|a^i\rangle = \frac{|F^i\rangle}{|M\rangle}$
8: $|v^i\rangle = |v'^i\rangle + dt * |a^i\rangle$
9: $|s^i\rangle = |v^i\rangle * dt + \frac{1}{2}|a^i\rangle * (dt)^2$
10: $|s^i\rangle \leftarrow$ scale_by_max_step($|s^i\rangle$)
11: $|x^{i+1}\rangle = |x^i\rangle + |s^i\rangle$
12: **end for**

placing line 7 of algorithm 6 with the following:

$$|a^i\rangle = \frac{|F^i\rangle - |v^i\rangle * \eta}{|M\rangle} \tag{29}$$

The Verlet algorithm has many update methods for the velocity, position, and acceleration, with the above being the simplest to implement. Other methods include the leap-frog algorithm and the velocity Verlet scheme.[22–24]

5.3 FIRE

The Fast Inertial Relaxation Engine (FIRE) is reminiscent of the Quick-Min algorithm: acceleration occurs while stepping along a predefined "good" path, whereas resetting velocities to 0 happens when taking "poor" steps. It differs, however, in the implementation of acceleration. Instead of projecting the velocity along the direction of the force, as is the case in Quick-Min, the FIRE method instead perturbs the velocity in such a way as to maintain its momentum along the steepest path. Due to the need to both increase and decrease acceleration and time steps, the FIRE algorithm comes with six parameters to adjust. However, those provided in Alg. 7 are known to be a robust set.[12]

Algorithm 7 FIRE

1: $|x^0\rangle \leftarrow$starting_guess()
2: $|v^0\rangle = |0\rangle$
3:
4: $dt = 0.1$
5: $N_{accelerate} = 5$
6: $f_{inc} = 1.1$
7: $f_{accelerate} = 0.99$
8: $f_{decelerate} = 0.5$
9: $a_{start} = 0.1$
10:
11: $n_{reset} = 0$ ▷ Step since reset
12: $a = a_{start}$
13:
14: **for** i in $range(0, N_{max})$ **do**
15: fit_rigid($|x^i\rangle, |v^i\rangle$)
16: $|F^i\rangle \leftarrow$NEB_Calc($|x^i\rangle$)
17: $|v'\rangle = (1.0 - a)|v^i\rangle + a * \left(\frac{\langle v^i|v^i\rangle}{\langle F^i|F^i\rangle}\right)^{0.5} |F^i\rangle$
18: **if** $i > 0$ **and** $\langle v^{i-1}|F^i\rangle > 0$ **then**
19: **if** $n_{reset} > N_{accelerate}$ **then**
20: $dt = min(dt * f_{inc}, dt_{max})$
21: $a = a * f_{accelerate}$
22: **end if**
23: $n_{reset} = n_{reset} + 1$
24: **else**
25: $|v'\rangle = |0\rangle$
26: $a = a_{start}$
27: $dt = dt * f_{decelerate}$
28: $n_{reset} = 0$
29: **end if**
30: $|v^i\rangle = |v'\rangle + dt|F^i\rangle$ ▷ Euler Step
31: $|s^i\rangle = dt|v^i\rangle$
32: $|s^i\rangle \leftarrow$ scale_by_max_step($|s^i\rangle$)
33: $|x^{i+1}\rangle = |x^i\rangle + |s^i\rangle$
34: **end for**

5.4 Conjugate Gradient Method

It is known that, using the Steepest Descent method, a jagged path is sometimes taken as the process steps towards the minimum (see Fig 5). This is due to subsequent steps being taken along an orthogonal gradient.

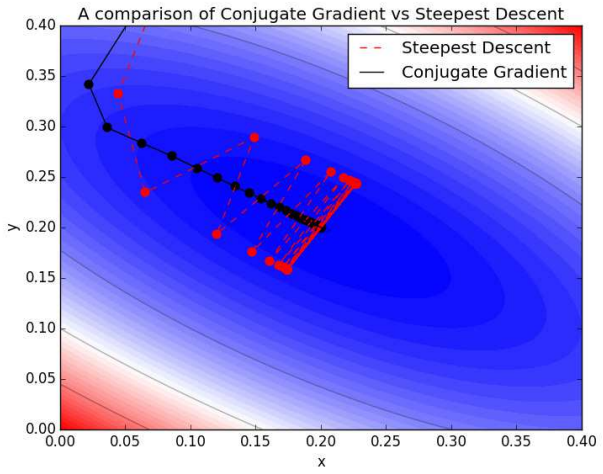


Figure 5: A comparison of the behavior of Conjugate Gradient and Steepest Descent methods when minimizing a function.

The Conjugate Gradient algorithm endeavors to fix this by using information from previous steps to ensure that each step $|s^i\rangle$ is conjugate to all other steps $|s^j\rangle$ where $i \neq j$. This is accomplished by

$$|s^i\rangle = |F^i\rangle + \beta|s^{i-1}\rangle \tag{30}$$

where β is a scaling factor. There are several methods for defining β, but the simplest (and fairly efficient) is that of Fletcher-Reeves[25]:

$$\beta = \frac{\langle F^i|F^i\rangle}{\langle F^{i-1}|F^{i-1}\rangle} \tag{31}$$

Thus, altering Alg. 5 with this new step direction $|s^i\rangle$ gives the Conjugate Gradient method, shown in Alg. 8.

Algorithm 8 Conjugate Gradient

$|x^0\rangle \leftarrow$starting_guess()
$|x^0\rangle \leftarrow$Procrustes($|x^0\rangle$)
$|F^0\rangle \leftarrow$NEB_Calc($|x^0\rangle$)
for i in $range(0, N_{max})$ **do**
  if i > 0 **then**
    $\beta = \frac{\langle F^i|F^i\rangle}{\langle F^{i-1}|F^{i-1}\rangle}$
    if β is inf **then**
      $\beta = 1.0$
    end if
    $|s^i\rangle = |F^i\rangle + \beta|s^{i-1}\rangle$
  else
    $|s^i\rangle = |F^i\rangle$
  end if
  $|s^i\rangle = \alpha|s^i\rangle$
  $|s^i\rangle \leftarrow$ scale_by_max_step($|s^i\rangle$)
  $|x^{i+1}\rangle = |x^i\rangle + |s^i\rangle$
  $|F^{i+1}\rangle \leftarrow$NEB_Calc($|x^{i+1}\rangle$)
  α, skip$\leftarrow$backtrack($|F^{i+1}_{RMS}\rangle, |F^i_{RMS}\rangle$)
  if skip **then**:
    continue
  end if
  fit_rigid($|x^{i+1}\rangle, [|F^{i+1}\rangle, |F^i\rangle, |s^i\rangle]$)
end for

5.5 BFGS

The Broyden-Fletcher-Goldfarb-Shanno (BFGS) method is a Quasi-Newton method that works similarly to the Steepest Descent method. The main difference is that there is an update to H^i (starting as the identity matrix) during each iteration that builds up an approximate inverse hessian. To improve on this, the starting inverse hessian can be scaled by some β. Although a static constant can be used, an improvement is to let $\beta = \frac{\langle y^i|\sigma^i\rangle}{\langle y^i|y^i\rangle}$.[20]

The BFGS method began from the Davidon-Fletcher-Powell (DFP) method,[15,26] whereby H^i is updated during each iteration using the following set of equations:

$$\begin{aligned} |y^i\rangle &= |g^{i+1}\rangle - |g^i\rangle \\ |\sigma^i\rangle &= \alpha^i|s^i\rangle \\ H^{i+1} &= H^i + \frac{|\sigma^i\rangle\langle\sigma^i|}{\langle\sigma^i|y^i\rangle} - \frac{H^i|y^i\rangle\langle y^i|H^i}{\langle y^i|H^i|y^i\rangle} \end{aligned} \tag{32}$$

where $|g^i\rangle$ is the gradient at some iteration (*i.e.*, the negative of the force in our case). Broyden helped improve this by using the following update method:[16–18,20,27]

$$\sigma^i = |x^{i+1}\rangle - |x^i\rangle \tag{33}$$

$$B^{i+1} = B^i + \frac{|y^i\rangle\langle y^i|}{\langle y^i|\sigma^i\rangle} - \frac{B^i|\sigma^i\rangle\langle\sigma^i|B^i}{\langle\sigma^i|B^i|\sigma^i\rangle}$$

$$H^{i+1} = \left(I - \frac{|\sigma^i\rangle\langle y^i|}{\langle\sigma^i|y^i\rangle}\right) H^i \left(I - \frac{\langle y^i|\sigma^i\rangle}{\langle\sigma^i|y^i\rangle}\right) + \frac{\langle\sigma^i|\sigma^i\rangle}{\langle\sigma^i|y^i\rangle}$$

Now σ^i is explicitly taken to be the difference in $|x\rangle$. The update method for the Hessian matrix, B^i, is shown alongside the update method for the inverse Hessian matrix, H^i, to avoid confusion in the notation. To obtain the BFGS H^i update method, the Sherman-Morrison-Woodbury formula is applied to B^i and simplified.[20]

A final update to the BFGS method was made by Sheppard and Henkelman in 2008, in which all images are minimized in a single iteration, instead of an image-by-image method.[3] This method, dubbed the global BFGS, was shown to perform better than the standard BFGS method. The method laid out in Alg. 9 is that of the the global BFGS method.

Algorithm 9 BFGS

$|x^0\rangle \leftarrow$starting_guess()
$H^i = I$ ▷ Identity matrix
$|x^0\rangle \leftarrow$Procrustes($|x^0\rangle$)
$|F^0\rangle \leftarrow$NEB_Calc($|x^0\rangle$)
for i in $range(0, N_{max})$ **do**
  $|s^i\rangle = H^i|F^i\rangle$
  $|s^i\rangle \leftarrow$ scale_by_max_step($|s^i\rangle$)
  $|x^{i+1}\rangle = |x^i\rangle + \alpha|s^i\rangle$
  $|x^i_{hold}\rangle, |F^i_{hold}\rangle = |x^i\rangle, |F^i\rangle$
  fit_rigid($|x^{i+1}\rangle, [|x^i\rangle, |F^i\rangle], H^i$)
  $|F^{i+1}\rangle \leftarrow$NEB_Calc($|x^{i+1}\rangle$)
  α, skip$\leftarrow$backtrack($|F^{i+1}_{RMS}\rangle, |F^i_{RMS}\rangle$)
  if skip **then**
    $H^i = I$ ▷ Reset Inverse Hessian
    $|x^i\rangle, |F^i\rangle = |x^i_{hold}\rangle, |F^i_{hold}\rangle$
    $i = i - 1$
    continue
  else
    $|\sigma^i\rangle = |x^{i+1}\rangle - |x^i\rangle$
    $\rho^i = \frac{1.0}{\langle y^i|\sigma^i\rangle}$
    $|y^i\rangle = (-|F^{i+1}\rangle) - (-|F^i\rangle)$
    if H^i is I **then**
      $H^i = \frac{\langle y^i|\sigma^i\rangle}{\langle y^i|y^i\rangle} I$
    end if
    $A^i = I_{n\times n} - |\sigma^i\rangle\langle y^i|\rho^i$
    $B^i = I_{n\times n} - |y^i\rangle\langle\sigma^i|\rho^i$
    $H^{i+1} = A^i H^i B^i + |\sigma^i\rangle\langle\sigma^i|\rho^i$
  end if
end for

5.6 LBFGS

Storing H^i, a matrix of $(3 * N * M)^2$ entries, where N is the number of atoms per image and M is the number of images, can quickly become memory-intensive. To compensate for this, Jorge Nocedal recognized that the BFGS update algorithm was an iterative method and, as such, memory can be saved by only applying the last N iterations.[19] His update method has been reproduced in Alg. 10. In the LBFGS method, the only difference from the BFGS method is this update for calculating $|s^i\rangle$ iteratively using σ_{hold} and y_{hold} to store previous iterations. The complete global LBFGS method is laid out in Alg. 11.

6 Results

6.1 Computational Techniques

Simulations were run using the Orca DFT package developed by the Max-Planck Institute for

Algorithm 10 BFGS_MULTIPLY

$s, y, |G\rangle, \beta \leftarrow$ input_args()
$|r\rangle \leftarrow \beta|G\rangle$
$N = \text{len}(s)$
$xs =$range(N)
$\alpha =$np.zeros(N) ▷ Initialize Empty Array
for i in $rev(xs)$ **do** ▷ Compute Right Prod
  $\rho_i = \frac{1.0}{\langle y_i|s_i\rangle}$
  $\alpha_i = \rho_i * \langle s_i|r\rangle$
  $|r\rangle = |r\rangle - \alpha_i|y_i\rangle$
end for
if N > 0 **then**
  $|r\rangle = \frac{\langle y_{last}|s_{last}\rangle}{\langle y_{last}|y_{last}\rangle}|r\rangle$
end if
for i in xs **do** ▷ Compute Left Prod
  $\rho_i = \frac{1.0}{\langle y_i|s_i\rangle}$
  $\kappa = \rho_i * \langle y_i|r\rangle$
  $|r\rangle = |r\rangle + (\alpha_i - \kappa)|s_i\rangle$
end for
return $|r\rangle$

Chemical Energy Conversion.[7] For benchmark purposes, Grimme's 3-corrected Hartree-Fock method was used for force and energy calculations.[28–31] A custom patch was written for the ASE library, allowing for the implementation of an Orca "calculator." Benchmark calculations were performed for a test case, the isomerization of CNX and BOX systems, where X was either H, Li, or Na. A final "production" validation was done for the study of a conformational change within an alanine dipeptide molecule. The parameters used by our codebase are listed in Table 1.

Within the ASE codebase, we used the default parameters. Calculations were done without the use of CI-NEB. Finally, all calculations were run until the maximum force on an atom across all NEB images was calculated to be less than 0.03 eV/Å (a very tight convergence), with a maximum number of allowable iterations being 1000.

6.2 Benchmarks

To validate the methods outlined in this paper, we proceeded by benchmarking against an already well-established codebase that imple-

Algorithm 11 LBFGS

$|x^0\rangle \leftarrow$starting_guess()
$\sigma_{hold} = [\,]$
$y_{hold} = [\,]$
$\beta = 1.0$
$|x^0\rangle \leftarrow$Procrustes$(|x^0\rangle)$
$|F^0\rangle \leftarrow$NEB_Calc$(|x^0\rangle)$
for i in $range(0, N_{max})$ **do**
  $|s^i\rangle = -$BFGS_MULTIPLY$(x_{hold}, F_{hold}, |F^i\rangle, \beta)$
  $|s^i\rangle \leftarrow$ scale_by_max_step$(|s^i\rangle)$
  $|x^{i+1}\rangle = |x^i\rangle + \alpha|s^i\rangle$
  $|x^i_{tmp}\rangle, |F^i_{tmp}\rangle = |x^i\rangle, |F^i\rangle$
  fit_rigid$(|x^{i+1}\rangle, [|x^i\rangle, |F^i\rangle, \sigma_{hold}, y_{hold}])$
  $|F^{i+1}\rangle \leftarrow$NEB_Calc$(|x^{i+1}\rangle)$
  α, skip$\leftarrow$backtrack$(|F^{i+1}_{RMS}\rangle, |F^i_{RMS}\rangle)$
  if skip **then**
    $\sigma_{hold}, y_{hold} = [\,], [\,]$ ▷ Reset here
    $|x^i\rangle, |F^i\rangle = |x^i_{tmp}\rangle, |F^i_{tmp}\rangle$
    $i = i - 1$
    continue
  else
    $|\sigma^i\rangle = |x^{i+1}\rangle - |x^i\rangle$
    $|y^i\rangle = (-|F^{i+1}\rangle) - (-|F^i\rangle)$
    if σ_{hold} is empty **then**
      $\beta = \frac{\langle y^i|\sigma^i\rangle}{\langle y^i|y^i\rangle}$
    else
      $\beta = 1.0$
    end if
    $\sigma_{hold} \leftarrow |\sigma^i\rangle$
    $y_{hold} \leftarrow |y^i\rangle$
  end if
end for

Table 1: A list of parameters used during benchmarking using our codebase.

	SD/CG	BFGS/LBFGS	QM	FIRE
α_0	0.1	1.0	-	-
γ	0.5	0.5	-	-
N_{reset}	-	20	-	-
dt	-	-	0.001	0.1
η	-	-	0	0
N_{acc}	-	-	-	5
f_{inc}	-	-	-	1.1
f_{acc}	-	-	-	0.99
f_{dec}	-	-	-	0.5
a_{start}	-	-	-	0.1
$\|s\rangle_{max}$	0.04	0.04	0.04	0.04

ments NEB for electronic state calculations, namely, ASE. The average number of iterations needed to reach convergence was calculated for each method, as shown in Fig. 6. On average, the methods outlined above converge to the MEP faster than the ASE implementations of BFGS, LBFGS, and FIRE (excluding our SD and FIRE implementation). The only instance of failed convergence arose using the ASE_LBFGS method for the isomerization of BOLi.

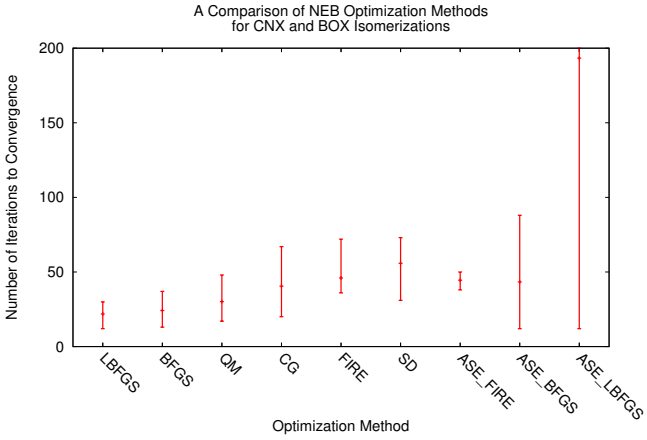


Figure 6: A comparison of NEB optimization methods for the isomerization of CNX and BOX where X is H, Li, and Na.

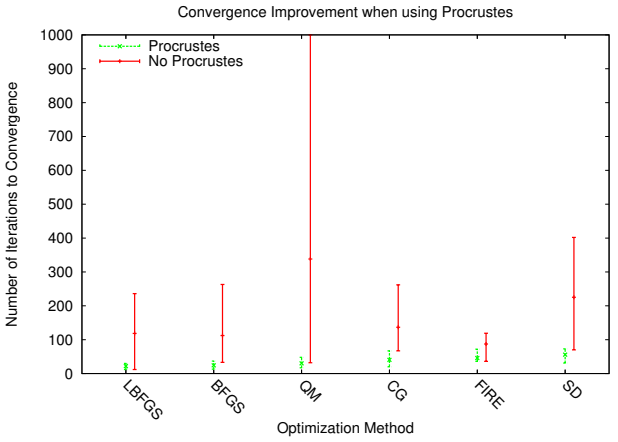


Figure 7: The efficacy of NEB optimization methods when rigid rotation is left alone (No Procrustes) or removed (Procrustes).

To validate that the removal of rigid rotations across images using the Procrustes Superimposition method improves the rate of convergence, a comparison was performed with and without this approach. This is shown in Fig. 7.

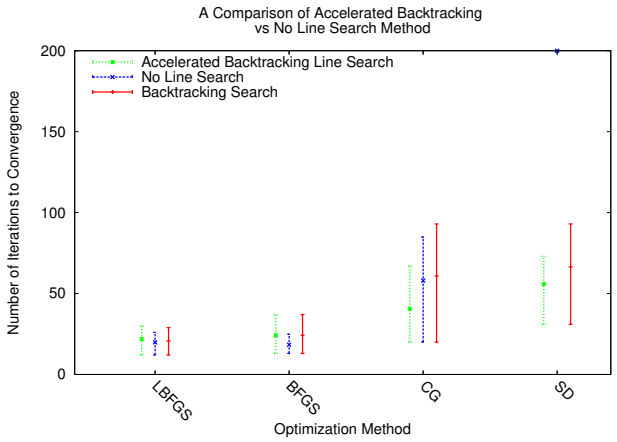


Figure 8: Validation of the efficacy of the accelerated backtracking line search method over one employing a static step size α. Comparison was performed *via* benchmark isomerizations of CNX and BOX, where X is H, Li, and Na. Procrustes was used in all line search benchmarks.

The accelerated backtracking line search method appears to show a slight improvement over that of a backtracking line search in the case of conjugate gradient and steepest descent. However, there is no noticeable improvement using the BFGS/LBFGS methods (on the contrary, it proved detrimental for BFGS). When comparing against steepest descent, a significant improvement can be found using a line search. Further, these improvements may be dependent on the initial step size; as such, starting with a larger step size may show more favorable results. Benchmark results for the line search methods are shown in Fig. 8.

An issue arose when benchmarks were performed over a distributed computer cluster, in which re-running calculations caused slightly different numbers of iterations to reach convergence. This difference is likely due to the propagation of floating-point discrepancies, and is most noticeable in the BFGS and LBFGS methods that take more than about 20 iterations to converge. To minimize this difference, all calculations were run in serial on the same computer. Unfortunately, the error still persisted and propagated over time in the ASE_LBFGS calculation of BOLi, leading to the abnormally high number of iterations to convergence and standard deviation shown in Fig. 6.

Validation was undertaken for a more complex system, that of the conformational changes associated with an alanine dipeptide. Fig. 9 shows the minimum energy pathways determined using our implementation of BFGS (converged in 89 iterations) and that of ASE (converged in 191 iterations). Numerical frequency calculations were performed on the maxima using the Orca package to (1) check if they were transition states (indicated by having one imaginary mode) and (2) verify whether an eigenvector-following method would lead to the same transition state. For both optimization methods, there was exactly one imaginary mode, shown in Table 2. After further eigenvector-following, both transition states converged to the same geometry, with similar vibrational modes (shown as -43.60 cm^{-1} for OptTS starting from the BFGS maxima and -43.68 cm^{-1} from the ASE_BFGS maxima).

Table 2: Transition State associated with an alanine dipeptide conformational change

Method	E_A (eV)	Mode	Freq. (cm^{-1})
BFGS	0.154	6	-43.72
ASE_BFGS	0.135	6	-45.26
OptTS	0.134	6	-43.60/-43.68

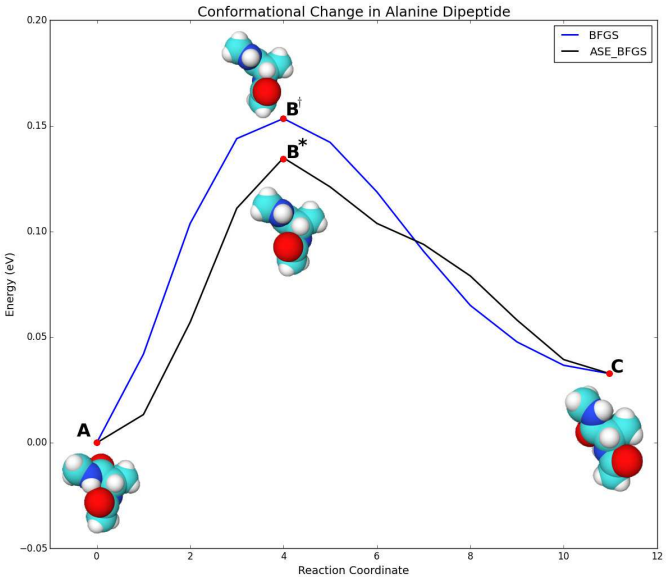


Figure 9: Validation of the NEB method for a more complex system, the minimum energy pathway of a conformational change in alanine dipeptide using the BFGS method (blue line) as compared to that of the ASE_BFGS method (black line). Points (A) and (C) indicate endpoints, while $B^\dagger$ and B^* are the maxima obtained from our method and the ASE codebase, respectively.

7 Conclusions

We have provided algorithms for the steepest descent, Conjugate Gradient, BFGS, LBFGS, FIRE, and Quick Min Verlet optimization techniques, and have elaborated on their implementation alongside the NEB method for non-periodic electronic state calculators such as Gaussian09 and Orca. We have validated their performance against a well-established codebase, the Atomic Simulation Environment, achieving comparable and, at times, improved results. Optimization methods all converge towards the same transition state with varying degrees of efficiency (in terms of iterations to convergence) and indeed, upon final eigenvector following, converge to the same geometry. In the case of steepest descent, convergence is improved by over 90% by using a line search. For steepest descent, we provided a new line search method that we have developed, which provides an accelerated backtracking line search. Using this new method, we found that convergence was improved by over 16% in comparison to the standard backtracking line search method (with further improvements possible for larger starting step sizes). On average, however, when using a line search, no significant improvement is seen for BFGS, LBFGS, and Conjugate Gradient. Finally, our implementation of the partial Procrustes Superimposition improves iterations to convergence, on average, by 74%.

8 Acknowledgments

This work was supported by the Cornell Center for Materials Research with funding from the NSF MRSEC program (DMR-1120296) through a seed grant. HH also thanks Corning Inc. for the provision of a Corning Glass Age

Fellowship which partially funded this work. Dr. Matthias Poloczek (ORIE, Cornell) and Mr. Chase Brisbois (MSE, Northwestern) are thanked for their suggestions which improved the flow and readability of this paper.

References

(1) Mills, G.; Jónsson, H. Quantum and Thermal Effects in H_2 Dissociative Adsorption: Evaluation of Free Energy Barriers in Multidimensional Quantum Systems. *Phys. Rev. Lett.* **1994**, *72*, 1124–1127.

(2) Jónsson, H.; Mills, G.; Jacobson, K. W. Nudged Elastic Band Method for Finding Minimum Emergy Paths of Transitions. In *Classical and Quantum Dynamics in Condensed Phase Simulations*; Berne, B. J., Ciccotti, G., Coker, D. F., Eds.; World Scientific, **1998**, Chapter 16, pp 385–404.

(3) Henkelman, G.; Jónsson, H. Improved tangent estimate in the nudged elastic band method for finding minimum energy paths and saddle points. *J. Chem. Phys.* **2000**, *113*, 9978–9985.

(4) Henkelman, G.; Uberuaga, B. P.; Jónsson, H. A climbing image nudged elastic band method for finding saddle points and minimum energy paths. *J. Chem. Phys.* **2000**, *113*, 9901–9904.

(5) Henkelman, G.; Jóhannesson, G.; Jónsson, H. Methods for Finding Saddle Points and Minimum Energy Paths. In *Theoretical Methods in Condensed Phase Chemistry*; Schwartz, S. D., Ed.; Springer Netherlands: Dordrecht, **2002**, pp 269–302.

(6) Sheppard, D.; Henkelman, G. Paths to which the nudged elastic band converges. *J. Comput. Chem.* **2011**, *32*, 1769–1771.

(7) Neese, F. The ORCA program system. *WIREs Comput. Mol. Sci.* **2012**, *2*, 73–78.

(8) Frisch, M. J.; Trucks, G. W.; Schlegel, H. B.; Scuseria, G. E.; Robb, M. A.; Cheeseman, J. R.; Scalmani, G.; Barone, V.; Mennucci, B.; Petersson, G. A.; Nakatsuji, H.; Caricato, M.; Li, X.; Hratchian, H. P.; Izmaylov, A. F.; Bloino, J.; Zheng, G.; Sonnenberg, J. L.; Hada, M.; Ehara, M.; Toyota, K.; Fukuda, R.; Hasegawa, J.; Ishida, M.; Nakajima, T.; Honda, Y.; Kitao, O.; Nakai, H.; Vreven, T.; Montgomery, J. A., Jr.; Peralta, J. E.; Ogliaro, F.; Bearpark, M.; Heyd, J. J.; Brothers, E.; Kudin, K. N.; Staroverov, V. N.; Kobayashi, R.; Normand, J.; Raghavachari, K.; Rendell, A.; Burant, J. C.; Iyengar, S. S.; Tomasi, J.; Cossi, M.; Rega, N.; Millam, J. M.; Klene, M.; Knox, J. E.; Cross, J. B.; Bakken, V.; Adamo, C.; Jaramillo, J.; Gomperts, R.; Stratmann, R. E.; Yazyev, O.; Austin, A. J.; Cammi, R.; Pomelli, C.; Ochterski, J. W.; Martin, R. L.; Morokuma, K.; Zakrzewski, V. G.; Voth, G. A.; Salvador, P.; Dannenberg, J. J.; Dapprich, S.; Daniels, A. D.; ö. Farkas,; Foresman, J. B.; Ortiz, J. V.; Cioslowski, J.; Fox, D. J. Gaussian09. Gaussian Inc. Wallingford CT 2009.

(9) Bahn, S. R.; Jacobsen, K. W. An object-oriented scripting interface to a legacy electronic structure code. *Comput. Sci. Eng.* **2002**, *4*, 56–66.

(10) Eggert, D. W.; Lorusso, A.; Fisher, R. B. Estimating 3-D rigid body transformations: a comparison of four major algorithms. *Mach Vision Appl* **1997**, *9*, 272–290.

(11) Sheppard, D.; Terrell, R.; Henkelman, G. Optimization methods for finding minimum energy paths. *J. Chem. Phys.* **2008**, *128*, 134106.

(12) Bitzek, E.; Koskinen, P.; Gähler, F.; Moseler, M.; Gumbsch, P. Structural Relaxation Made Simple. *Phys. Rev. Lett.* **2006**, *97*, 170201.

(13) Andersen, H. C. Molecular dynamics simulations at constant pressure and/or temperature. *J. Chem. Phys.* **1980**, *72*, 2384–2393.

(14) Hestenes, M. R.; Stiefel, E. Methods of Conjugate Gradients for Solving Linear Systems. *J. Res. Nat. Bur. Stand.* **1952**, *49*, 409–436.

(15) Fletcher, R.; Powell, M. J. D. A Rapidly Convergent Descent Method for Minimization. *Comput. J* **1963**, *6*, 163–168.

(16) Broyden, C. G. The Convergence of a Class of Double-rank Minimization Algorithms - 1. General Considerations. *J. Inst. Math. Appl.* **1970**, *6*, 76–90.

(17) Broyden, C. G. The Convergence of a Class of Double-rank Minimization Algorithms - 2. The New Algorithm. *J. Inst. Math. Appl.* **1970**, *6*, 222–231.

(18) Broyden, C. G. The Convergence of Single-Rank Quasi-Newton Methods. *Math. Comput.* **1970**, *24*, 365–382.

(19) Nocedal, J. Updating Quasi-Newton Matrices with Limited Storage. *Math. Comput.* **1980**, *35*, 773–782.

(20) Nocedal, J.; Wright, S. J. *Numerical optimization*; Springer Series in Operations Research and Financial Engineering; Springer: Berlin, **2006**.

(21) Liu, D. C.; Nocedal, J. On the Limited Memory BFGS Method for Large Scale Optimization. *Math. Program.* **1989**, *45*, 503–528.

(22) Grønbech-Jensen, N.; Farago, O. A simple and effective Verlet-type algorithm for simulating Langevin dynamics. *Mol. Phys.* **2013**, *111*, 983–991.

(23) Diethelm, K.; Ford, N. J.; Freed, A. D. A Predictor-Corrector Approach for the Numerical Solution of Fractional Differential Equations. *Nonlinear Dyn.* **2002**, *29*, 3–22.

(24) Skeel, R. D. Variable step size destabilizes the Störmer/leapfrog/verlet method. *BIT* **1993**, *33*, 172–175.

(25) Fletcher, R.; Reeves, C. M. Function minimization by conjugate gradients. *Comput. J* **1964**, *7*, 149–154.

(26) Davidon, W. C. Variable Metric Method for Minimization. *A.E.C. Research and Development Report, ANL-5990* **1991**, *1*, 1–17.

(27) Li, D.-H.; Fukushima, M. On the Global Convergence of the BFGS Method for Nonconvex Unconstrained Optimization Problems. *SIAM J. Optim.* **2001**, *11*, 1054–1064.

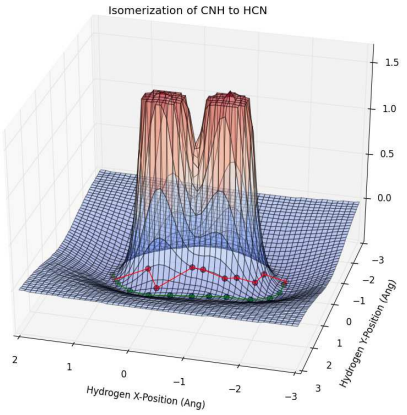
(28) Sure, R.; Grimme, S. Corrected small basis set Hartree-Fock method for large systems. *J. Comput. Chem.* **2013**, *34*, 1672–1685.

(29) Kruse, H.; Grimme, S. A geometrical correction for the inter- and intra-molecular basis set superposition error in Hartree-Fock and density functional theory calculations for large systems. *J. Chem. Phys.* **2012**, *136*, 154101.

(30) Grimme, S.; Ehrlich, S.; Goerigk, L. Effect of the damping function in dispersion corrected density functional theory. *J. Comput. Chem.* **2011**, *32*, 1456–1465.

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

For Table of Contents Only



Computational Implementation of the Nudged Elastic Band Method, Rigid Rotation and Optimization

Henry Herbol, James Stevenson, Paulette Clancy