Automated handling of complex chemical structures in Z-matrix coordinates – the chemcoord library

Oskar Weser,[*,†,‡] Björn Hein Hanke,[‡] and Ricardo A. Mata[‡]

[†]*Max-Planck-Institute for Solid State Research, Stuttgart, Germany*
[‡]*Institute of Physical Chemistry, University of Goettingen, Germany*

E-mail: oskar.weser@gmail.com

Abstract

In this work, we present a fully automated method for the construction of chemically meaningful sets of non-redundant internal coordinates (also commonly denoted as Z-matrices) from the cartesian coordinates of a molecular system. Particular focus is placed on avoiding ill-definitions of angles and dihedrals due to linear arrangements of atoms, to consistently guarantee a well-defined transformation to cartesian coordinates, even after structural changes. The representations thus obtained are particularly well suited for pathway construction in double-ended methods for transition state search and optimisations with non-linear constraints. Analytical gradients for the transformation between the coordinate systems were derived for the first time, which allows analytical geometry optimizations purely in Z-matrix coordinates. The geometry optimisation was coupled with a Symbolic Algebra package to support arbitrary non-linear constraints in Z-matrix coordinates, while retaining analytical energy gradient conversion. Sample applications are provided for a number of common chemical reactions and illustrative examples where these new algorithms can be used to automatically produce chemically reasonable structure interpolations, or to perform non-linearly constrained optimisations of molecules.

Introduction

The choice of atomic coordinate systems and the transformation between different representations has been a matter of debate since the first molecular potential energy surfaces were explored. Many of these representations are based on our own, however biased, way of thinking about molecules in terms of atoms and bonds. Internal coordinates span the potential hypersurface on the basis of bond distances, angles and dihedrals,[1–3] but for a non-redundant description there is no unique set available. The coupling between different coordinates strongly depends on the criteria applied in their construction as well as the specific point in chemical space being described.[4] This matter is particularly complicated when following the course of a chemical reaction, whereby the number or type of bonds will necessarily change. Defining a set of coordinates for start, end points and everything in between is difficult and will usually require user input. In addition structural changes along a reaction may lead to an ill-defined (back-)transformation (from) to internal coordinates.[5] An illustrative example of the problems faced is included in the Supporting Information.

These issues become even more pertinent with the growing number of projects relying on machine learning (ML), large scale structure encoding,[6] and automated, robust processes for chemical structure conversion and comparison.[7,8] It should be noted that as a general use descriptor, the Z-matrix representation is not very useful. Although it is translational and rotational invariant, it is not permutational invariant. It is, however, still a powerful tool in structure optimisation and related applications.[9] A recent study by Meyer and Hauser highlights the advantages of non-redundant internal coordinate systems in Gaussian process regression minima optimisation,[10] and is equally helpful for transition state search.[11] It also finds applications in several pre- and post-processing tasks. For example, in the computation of phase corrections for molecular photodynamics calculations, one requires an automated algorithm for the detection of a phase shift. Interpolation of two structures through a Z-matrix can be used for pathological cases where the geometric overlap is close to zero.[12] However, its greatest relevance is in day to day applications, as many codes struggle to handle exceptions when atoms adopt linear conformation and the (back-)conversion (from) to internal coordinates becomes ill-defined. These are regular occurrences when using standard quantum chemistry packages in geometry optimisations under the application of constraints or in more general searches.

We propose a black-box algorithm that generates chemically meaningful Z-matrices and automatically avoids the conditions under which the coordinate transformation between Z-matrices and cartesian coordinates becomes ill-defined. This new algorithm also works in a hybrid way, i.e. the user can define certain internal coordinates and dependencies while the rest of the Z-matrix is automatically constructed. The automatically generated Z-matrices agree with human chemical intuition, have a set of coordinates that simplify the exploration of potential hyper surfaces (PHS) of a molecule, and can be reliably transformed to cartesian coordinates even after structural changes.

Most electronic structure algorithms and codes yield energy gradients in cartesian coordinates. For the first time we derived fully analytical expressions for the Jacobian matrices of the coordinate transformations between cartesian and Z-matrix coordinates which allow the analytical transformation of energy gradients to Z-matrix coordinates. The transformed energy gradients allow the structure optimization purely in Z-matrix coordinates. So far this transformation had to be performed numerically either by using a self-consistent equation that is solved iteratively,[13] or by "learning" the gradient transformation in the context of machine learning applications.[9] Since we coupled our code to the symbolic algebra package `sympy`[14] it is even possible to perform structure optimizations with respect to arbitrary differentiable expressions in the Z-matrix. These non-linearly constrained optimizations can be used to reduce the degrees of freedom, enforce symmetries, or guide the search for transition states. Especially with a chemically meaningful Z-matrix these constraints can easily match chemical intuition and are easy to define for the user which makes Z-matrices superior compared to *redundant internal coordinates* in that regard. On the other hand we also have to note that *unconstrained* geometry optimization in *redundant internal coordinates*[15,16] still outperforms Z-matrices. One example of a constrained optimization in this work is the highly symmetric cubane where the cubic symmetry can be imposed in Z-matrix coordinates and by optimizing only r_{CC}, r_{CH}, and α the problem size is reduced to three degrees of freedom (from $3 \cdot 16 - 6 = 42$).

One of the advantages of the chemically meaningful and reliable Z-matrix is the possibility to interpolate internal coordinates (IC) in an automated fashion. Using the same construction table for both minimum structures, their Z-matrices can be interpolated just as easily as their cartesian coordinates and if the Z-matrices were defined in a chemically meaningful way, the interpolated structures will be usually much more chemically reasonable. We demonstrate the superior automated interpolation by generating starting guesses for the Nudged Elastic Band (NEB) method. NEB approximates minimum energy paths (MEPs) by connecting minimum structures as a string of discrete intermediate structures ("images"). At the time of writing, most NEB implementations generate the initial pathway by interpolation in cartesian coordinates. These interpolations will often generate structures with compressed bonds or overlapping atoms when bond angles change or molecule fragments rotate against each other. There has been development of methods that obviate this problem, such as the Image-Dependent Pair Potential (IDPP),[17] which performs a pre-optimization of the initial cartesian interpolation, reducing atom overlaps and bond deformations. In this work we demonstrate that starting guesses from interpolation in internal coordinates, that use *black-box-like automatically generated* Z-matrices, performs similar or better compared to cartesian interpolation + IDPP.

All discussed algorithms were implemented in an open source python library `chemcoord` that is publicly available at `https://github.com/mcocdawc/chemcoord` and is actively used by other chemists around the world.

Methods

Chemist Z-matrix definition

It is the aim of this section to analyse in depth the linearity pitfalls that may appear in the use of Z-matrices and how to avoid them. A further aim is to define heuristics for the selection of references that lead to a Z-matrix as a human chemist would draw it.

In order to precisely analyse the possible pitfalls when transforming from Z-matrix to cartesian coordinates, it is advantageous to adopt the view of the *Self-Normalizing Natural Extension Reference Frame* (SN-NeRF) as described by Parsons et al.[5] The Z-matrix entries for an atom are interpreted as spherical coordinates and an atom is put into the canonical reference frame. This frame is then rotated and translated into the orthonormalised, righthanded reference frame $\mathbf{B}$ which is spanned locally by the three reference atoms. In the following the variables r, α, δ should canonically stand for distance, angle and dihedral. The index variables i, b, a, d should canonically stand for the index of the atom to be inserted and its bond, angle, and dihedral defining atom. $\mathbf{v}_i, \mathbf{v}_b, ...$ should be the respective positions. By defining b, a, and d we implicitly assumed that there are three reference atoms, which is not the case for the first three atoms. Practically this implies that the upper right triangle of a Z-matrix is left empty and that there exist conventions for putting the first atom into the origin, align the second atom along the z-axis etc. To describe the translational and rotational degrees of a molecule in Z-matrix coordinates we will instead use absolute reference positions for the first three atoms.[18]

The conversion to cartesian coordinates is now elegantly factored, because the first positioning into the canonical reference frame is a function of only r, α, and δ and the affine-linear transformation into the local reference frame $\mathbf{B}$ is a function of only $\mathbf{v}_b$, $\mathbf{v}_a$, and $\mathbf{v}_d$ as can be seen in equation 1.

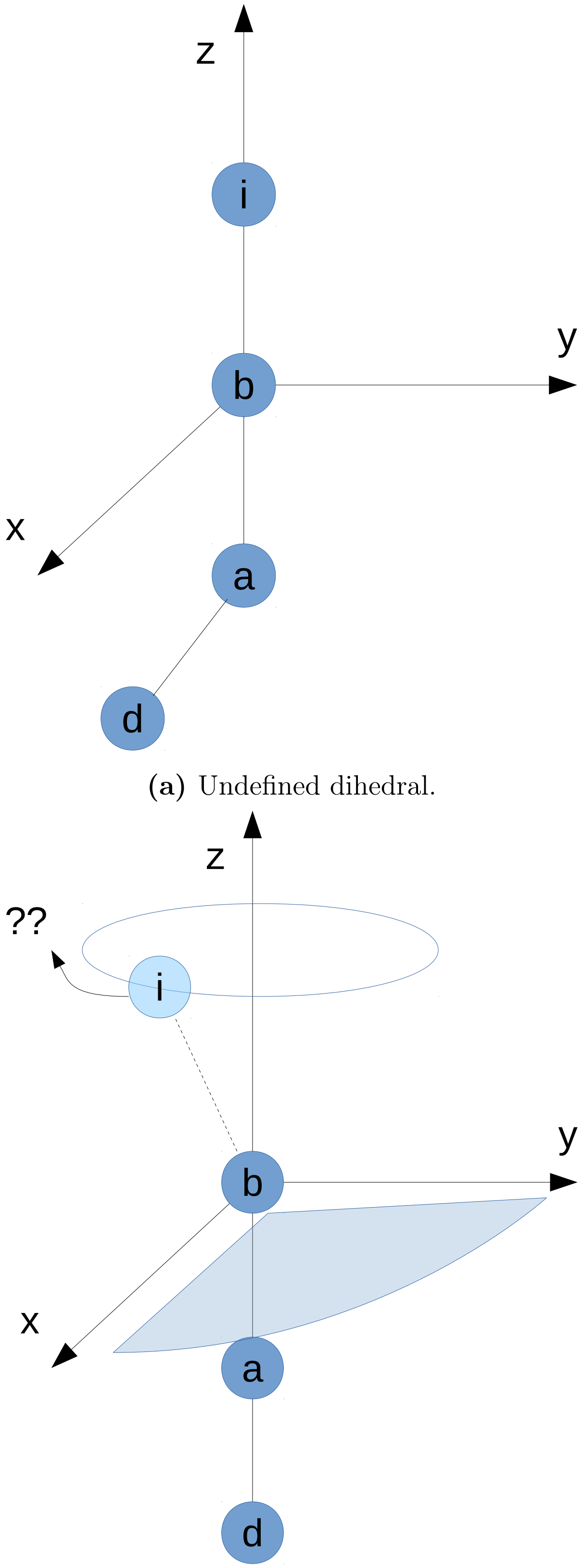
$$\mathbf{v}_i = \mathbf{B}(\mathbf{v}_b, \mathbf{v}_a, \mathbf{v}_d) \cdot \mathbf{s}(r, \alpha, \delta) + \mathbf{v}_b \tag{1}$$

The transformation from cartesian to Z-matrix coordinates is then given by inverting equation 1:

$$\begin{aligned}(r_i, \alpha_i, \delta_i) &= \mathbf{s}^{-1}(\mathbf{B}^{-1}(\mathbf{v}_b, \mathbf{v}_a, \mathbf{v}_d) \cdot (\mathbf{v}_i - \mathbf{v}_b)) \\ &= \mathbf{s}^{-1}(\mathbf{B}^{\mathsf{T}}(\mathbf{v}_b, \mathbf{v}_a, \mathbf{v}_d) \cdot (\mathbf{v}_i - \mathbf{v}_b)),\end{aligned} \tag{2}$$

with $\mathbf{B}$ orthogonal. A Z-matrix that can be converted to cartesian coordinates will be referred to as defined or valid Z-matrix. With the given working equations it is now possible to isolate the two possible pitfalls that prevent a defined transformation between the coordinate systems as illustrated in Figure 1.

The undefined dihedral in Figure 1a appears as a definition problem when converting from cartesian to Z-matrix coordinates. If the angle $\alpha = \angle(i, b, a)$ equals 180° there are infinitely many possible values for the dihedral and the result of $\mathbf{s}^{-1}$ is undefined. It is assumed that energy and all other quantities of interest may be formulated solely using cartesian coordinates, so the surjectivity of $\mathbf{s}$ is sufficient and the dihedral is set to 0° in this case. When an angle α of 180° is encountered upon converting from



(a) Undefined dihedral.

(b) Undefined local coordinate system.

Figure 1: Schematic drawing of the two corner cases.

Z-matrix to cartesian coordinates, the atom i is simply set to

$$\mathbf{v}_i = \mathbf{v}_b + r\frac{\mathbf{v}_b - \mathbf{v}_a}{\|\mathbf{v}_b - \mathbf{v}_a\|} \tag{3}$$

independent of the dihedral value.

The undefined local coordinate system in Figure 1b appears if the reference atoms are linearly aligned. In this case the matrix $\mathbf{B}$ in equation 1 does not have full rank anymore and the matrix $\mathbf{B}^{-1}$ in equation 2 is undefined. It has to be stressed that this pitfall appears independently of Z-matrix entries r, α, and δ and depends solely on the position of the reference atoms in cartesian coordinates. For this reason it is not possible to test for this pathological case without converting the Z-matrix to cartesian coordinates. To prevent this error it is necessary to assert validity of the Z-matrix upon creation by properly selecting the reference atoms. If any bond, angle, or dihedral is modified later on, it is necessary to convert to cartesian coordinates. If an invalid reference for atom i is found, a dummy atom has to be inserted in the plane spanned by $\mathbf{v}_b, \mathbf{v}_a, \mathbf{v}_d$ *before* the modification. We would like to emphasize that both cases that cause undefined Z-matrices can appear in usual molecular systems. An illustrative example of the problems faced is included in the Supporting Information, where we compare with OpenBabel, the de-facto standard tool for coordinate transformations, in the community.[19]

Up to this point the Z-matrix was just treated as a coordinate system for arbitrary points in space. It is now the aim to obtain a Z-matrix suited for chemical needs. For this purpose the construction table is defined as a Z-matrix with only the b, a, and d columns containing only the indices of reference atoms. The problem of defining a chemical Z-matrix reduces to finding a "good" construction table. A specific construction table could also be created manually while the calculation of Z-matrix entries bond, angle, and dihedral is performed by the computer. It has to be emphasized that there is not one single correct solution for choosing a construction table, but the following rules of thumb should give similar results to a human chemist constructing a Z-matrix manually:

- The bond defining atom should lie in the the first coordination sphere of i.
- The angle defining atom should lie in the first coordination sphere of the bond defining atom and in the second coordination sphere of atom i.
- The dihedral defining atom should lie in the first coordination sphere of the angle defining atom, in the second coordination sphere of the bond defining atom, and in the third coordination sphere of atom i.
- The Z-matrix definition should start in the geometric center of the molecule i. e. start with the atom nearest to the geometric center.
- High valency atoms should appear higher up in the Z-matrix and are preferred reference atoms. Valency in this context is simply the number of connected atoms. There is no distinction between single, double, and triple bonds.
- The variation in the b, a, and d columns should be kept minimal. If it is possible for atom i to use the same references as $i-1$, without violating previous constraints, it should do so.
- The used references should propagate through, which means that the bond defining atom of the bond defining atom of i should be the angle defining atom of i and so on.

It is worth noting that apart from starting near the molecule's geometric center, all other constraints are solely dependent on the connectivity graph and not on specific positions in cartesian space. Hence it is advantageous to use a suitable data structure representing the connectivity graph. Since the following algorithm involves several tests for membership and set intersections, it is advantageous to use hashtable based data structures.

It is important to note that a single connected structure is assumed for algorithm 1, but in certain situations it is necessary to construct a Z-matrix for an ensemble of molecules/fragments. In this case algorithm 1 is applied onto each fragment/substructure separately. The resulting Z-matrices are then concatenated, which is performed by substituting the absolute references in the first three rows with atom labels from other Z-matrices.

Algorithm 1 Finding a chemical construction table.

```
function GET_CONSTRUCTION_TABLE($\mathbf{X}$)
    $B$ := GET_CONNECTIVITY_GRAPH($\mathbf{X}$)
    $i$ := GET_ATOM_LABEL_CLOSEST_TO_CENTROID($\mathbf{X}$)
    visited := {$i$}
    $R[i]$ := {$i$: {'b': 'origin', 'a': 'e_z', 'd': 'e_x'}}        ▷ This is the construction table.
    parent := {$j$ : $i$ for $j$ in $B[i]$}
    work_bond_dict := {$j$ : ($B[j] \setminus$ visited) for $j$ in $B[i]$}
    while not empty work_bond_dict do
        for $i$ in SortByValency(Keys(work_bond_dict)) do
            if $i \in$ visited then
                continue
            else
                $b$ := parent[$i$]
                if $b$ in Index($R$)[:3] then
                    if Length($R$) = 1 then
                        $R[i]$ := {'b': $b$, 'a': 'e_z', 'd': 'e_x'}
                    else if Length(R) = 2 then
                        $R[i]$ := {'b': $b$, 'a': MaxV($B[b] \cap$ Keys($R$)) , 'd': 'e_x'}
                                ▷ MaxV returns the atom label with the highest valency.
                    else
                        if Exists(parent[b]) then
                            $a$ := parent[$b$]
                        else
                            $a$ := MaxV($B[b] \cap$ Keys($R$))
                        end if
                        if Exists(parent[$a$]) and parent[$a$] not in $\{b, a\}$ then
                            $d$ := parent[$a$]
                        else
                            if not empty ($B[a] \cap$ Keys($R$)) $\setminus \{b, a\}$ then
                                $d$ := MaxV(($B[a] \cap$ Keys($R$)) $\setminus \{b, a\}$)
                            else
                                $d$ := MaxV(($B[b] \cap$ Keys($R$)) $\setminus \{b, a\}$)
                            end if
                        end if
                        $R[i]$ :={'b': $b$, 'a': $a$, 'd': $d$}
                    end if
                else
                    $R[i]$ := {'b': $b$, 'a': $R[b]$['b'], 'd': $R[b]$['a']}
                end if
            end if
            visited := visited $\cap\{i\}$
            for $j$ in SortByValency(Keys(work_bond_dict[$i$])) do
                new_work_bond_dict[$j$] := $B[j] \setminus$ visited
                parent[$j$] := $i$
            end for
        end for
        work_bond_dict := new_work_bond_dict
    end while
    return R
end function
```

A test to prevent a locally undefined coordinate system (Figure 1b) is still missing in algorithm 1. This test is performed afterwards by checking if there exists an atom i whose reference atoms are linearly aligned. If such an atom is found, its dihedral defining atom d is changed (preferably) to an atom from the same coordination sphere relative to i. Instead of testing afterwards, it is possible to include additional conditionals in algorithm 1 to reject a selection for $d(i)$, if the reference of i is linear. This inline testing could be faster than the additional test afterwards, but it would further increase the amount of nested conditionals up to a point where the code becomes very hard to understand.

Equations for the coordinate transformation

In this section the working equations for the transformation between Z-matrix and cartesian coordinates are given and the gradients are derived. It will be assumed that the corner case of an undefined coordinate system as discussed in the previous section and shown in Figure 1a does not appear. Due to mathematical elegance and superior perfomance the *Self-Normalizing Natural Extension Reference Frame* (SN-NeRF) algorithm by Parsons et al. is used.[5]

In order to store the information about the absolute positions in cartesian space an approach similar to Hoft et al. was implemented.[18] In the first three rows the origin and canonical unit vectors can be treated as if they were reference atoms, to specify translational and rotational degrees of freedom. These fixed points in cartesian space that are used to specify the absolute position will be called absolute references.

The section will heavily rely on tensor formulation. This might seem hyperbolic for the transformation functions but it is necessary for deriving the gradients. A further advantage of the tensor formulation lies in the similarity between derived equations and actual implementation using vectorized calls.

The following typographic rules for the notation of variables will be used:

- Scalars will be denoted with italic letters
- Tensors of order one (vectors) will be denoted with lowercase, bold letters: $\mathbf{x}$
- Tensors of order higher than one will be denoted with uppercase, bold letters: $\mathbf{X}$

The typography of functions depends on the tensorial order of their image.
In the case of tensors the colon notation of array slices as in Fortran is used. So the i-th column vector of the matrix $\mathbf{X}$ may be written as $\mathbf{X}_{:,i}$ and the j-th row vector may be written as $\mathbf{X}_{j,:}$.

Another important convention is the layout of coordinates. The row wise alignment of XYZ files makes sense for these CSV like files. But in the following section it is mathematically advantageous to use a column wise alignment of coordinates. This implies that the cartesian positions and Z-matrix entries of a molecule with n atoms are an element of $\mathbb{R}^{3,n}$.

Transformation from Z-matrix to cartesian coordinates

Let n be the number of atoms in a molecule. Let $I := [1...n]$ be the index set of the molecule, and $I^R := [-2...n]$ the index set of the molecule with three indices of absolute references. The b, a, and d can also be interpreted as functions that return for a given atom index i the respective bond, angle, and dihedral defining atom of i. In this case, we know that

$$i > b(i), a(i), d(i) \quad . \tag{4}$$

$\mathbf{C} \in ([0, \infty) \times [0, \pi) \times [0, 2\pi))^n$ is defined to be the matrix of bond length r, angle α and dihedral δ of the i-th atom:

$$\mathbf{C}_{:,i} = (r_i, \alpha_i, \delta_i)^\mathsf{T}$$

The distance $r_i = \mathbf{C}_{1,i}$ is the distance between atom i and $b(i)$, the angle $\alpha_i = \mathbf{C}_{2,i}$ is the angle between atom i, $b(i)$, and $a(i)$ and the dihedral $\delta_i = \mathbf{C}_{3,i}$ is the dihedral between atom i, $b(i)$, $a(i)$, and $d(i)$. Let $\mathbf{X} \in \mathbb{R}^{3,n}$ be the matrix of all atom positions. Let $\mathbf{X}^R \in \mathbb{R}^{n+3,3}$ be the matrix of all atom positions plus the three absolute references. (This means that the row index of $\mathbf{X}^R$ starts at -2.) So $\mathbf{X}_{:,i}$ and $\mathbf{X}^R_{:,i}$ are the position of the i-th atom. The position of the i-th atom is given by:

$$\begin{aligned}\mathbf{X}_{:,i}\colon [0,\infty) \times [0,\pi) \times [0, 2\pi) \times \mathbb{R}^{3,3} &\to \mathbb{R}^3 \\ (\mathbf{C}_{:,i}, \mathbf{X}^R_{:,\mathbf{I}(i)}) &\mapsto \mathbf{X}_{:,i}(\mathbf{C}_{:,i}, \mathbf{X}^R_{:,\mathbf{I}(i)})\end{aligned}$$

Let $\mathbf{B}$ be the third rank tensor, which contains the orthonormal bases spanned by the references of the i-th atom.

$$\begin{aligned}\mathbf{B}\colon \mathbb{R}^{(n+3),3} &\to \mathbb{R}^{n,3,3} \\ \mathbf{ba}_i &\coloneqq \frac{\mathbf{X}^R_{:,a(i)} - \mathbf{X}^R_{:,b(i)}}{\left\|\mathbf{X}^R_{:,a(i)} - \mathbf{X}^R_{:,b(i)}\right\|} \\ \mathbf{ad}_i &\coloneqq \frac{\mathbf{X}^R_{:,d(i)} - \mathbf{X}^R_{:,a(i)}}{\left\|\mathbf{X}^R_{:,d(i)} - \mathbf{X}^R_{:,a(i)}\right\|} \\ \mathbf{B}_{:,3,i} &\coloneqq -\mathbf{ba}_i \\ \mathbf{B}_{:,2,i} &\coloneqq \frac{\mathbf{ad}_i \times \mathbf{ba}_i}{\|\mathbf{ad}_i \times \mathbf{ba}_i\|} \\ \mathbf{B}_{:,1,i} &\coloneqq \frac{\mathbf{B}_{:,2,i} \times \mathbf{B}_{:,3,i}}{\|\mathbf{B}_{:,2,i} \times \mathbf{B}_{:,3,i}\|}\end{aligned} \tag{5}$$

Using the IUPAC defined rotation directions the function for transforming spherical to cartesian coordinates is defined as:[20]

$$\begin{aligned}\mathbf{S}\colon ([0,\infty) \times [0,\pi) \times [0,2\pi))^n &\to \mathbb{R}^{3,n} \\ \begin{pmatrix} r \\ \alpha \\ \delta \end{pmatrix} \coloneqq \mathbf{C}_{:,i} \mapsto \mathbf{S}_{:,i} &\coloneqq \begin{pmatrix} r\sin(\alpha)\cos(\delta) \\ -r\sin(\alpha)\sin(\delta) \\ -r\cos(\alpha) \end{pmatrix}\end{aligned} \tag{6}$$

If we collect $(b(i), a(i), d(i))$ into $\mathbf{I}(i)$ we can combine the previous equations to yield a more explicit version of equation 1 as

$$\mathbf{X}_{i,j}(\mathbf{C}_{:,j}, \mathbf{X}^R_{:,\mathbf{I}(j)}) = \mathbf{B}_{i,:,j}(\mathbf{X}^R_{:,\mathbf{I}(j)}) \cdot \mathbf{S}_{:,j}(\mathbf{C}_{:,j}) + \mathbf{X}^R_{i,b(j)} \tag{7}$$

$$\mathbf{X}^R_{:,i}(\mathbf{C}_{:,i}, \mathbf{X}^R_{:,\mathbf{I}(i)}) = \begin{cases} \mathbf{X}_{:,i}(\mathbf{C}_{:,i}, \mathbf{X}^R_{:,\mathbf{I}(i)}) & i \in I \\ \mathbf{X}^R_{:,i} & i \in I^R \setminus I. \end{cases} \tag{8}$$

Since $\mathbf{X}^R$ itself depends on $\mathbf{C}$, the function $\mathbf{X}$ can be nested until the references are one of the three absolute references.

Gradient for the transformation from Z-matrix to cartesian coordinates

The derivatives of cartesian coordinates with respect to bonds, angles, and dihedrals in the Z-matrix shall be evaluated. This gradient is especially important because it will allow us to transform energy gradients analytically from cartesian to Z-matrix coordinates. Then it is possible to perform geometry optimizations purely in Z-matrix coordinates although most electronic structure methods yield energy gradients in cartesian coordinates.

Using the defined variables of the previous subsection the derivative of equation 7 can be written as:

$$\begin{aligned}\frac{\partial \mathbf{X}_{i,j}(\mathbf{C})}{\partial \mathbf{C}_{k,l}}(\mathbf{C}) &= \frac{\partial \mathbf{B}_{i,:,j}(\mathbf{X}^R_{:,\mathbf{I}(j)}) \cdot \mathbf{S}_{:,j}(\mathbf{C}) + \mathbf{X}^R_{i,b(j)}(\mathbf{C})}{\partial \mathbf{C}_{k,l}}(\mathbf{C}) \\ &= \mathbf{B}_{i,:,j}(\mathbf{X}^R_{:,\mathbf{I}(j)}) \frac{\partial \mathbf{S}_{:,j}(\mathbf{C})}{\partial \mathbf{C}_{k,l}} \\ &\quad + \frac{\partial \mathbf{B}_{i,:,j}(\mathbf{X}^R_{:,\mathbf{I}(j)})}{\partial \mathbf{X}^R_{:,\mathbf{I}(j)}}(\mathbf{X}^R_{:,\mathbf{I}(j)}(\mathbf{C})) \\ &\quad \cdot \frac{\partial \mathbf{X}^R_{:,\mathbf{I}(j)}(\mathbf{C})}{\partial \mathbf{C}_{k,l}}(\mathbf{C}) \cdot \mathbf{S}_{:,j}(\mathbf{C}) \\ &\quad + \frac{\partial \mathbf{X}^R_{i,b(j)}(\mathbf{C})}{\partial \mathbf{C}_{k,l}}(\mathbf{C})\end{aligned} \tag{9}$$

This leads to three different cases depending on $(j < l)$, $(j = l)$, or $(j > l)$. Each case is discussed individually in Appendix A (equations 20, 22, and 23). The symbolically evaluated derivatives $\frac{\partial \mathbf{B}(\mathbf{X}^R)}{\partial \mathbf{X}^R}$ are given in the supporting information. They were calculated using the symbolic algebra package SymPy.[14] With all equations at hand we can formulate the combined algorithm 2.

Algorithm 2 Calculation of the analytical gradient for the coordinate transformation from Z-matrix to cartesian coordinates

```
for $j = 1 \dots n$ do
    for $l = (j+1) \dots n$ do
                                        ▷ Alternatively the whole array may be initialized with zeros.
        $\left(\frac{\partial \mathbf{X}(\mathbf{C})}{\partial \mathbf{C}}\right)_{:,j,l,:}(\mathbf{C}, \mathbf{X}^R) \coloneqq \frac{\partial \mathbf{X}_{:,j}(\mathbf{C})}{\partial \mathbf{C}_{:,l}}(\mathbf{C}, \mathbf{X}^R) \coloneqq 0$
    end for
end for
for $j = 1 \dots n$ do
    $\left(\frac{\partial \mathbf{X}(\mathbf{C})}{\partial \mathbf{C}}\right)_{:,j,j,:}(\mathbf{C}, \mathbf{X}^R) \coloneqq \frac{\partial \mathbf{X}_{:,j}(\mathbf{C})}{\partial \mathbf{C}_{:,j}}(\mathbf{C}, \mathbf{X}^R) = \mathbf{B}_{:,:,j}(\mathbf{X}^R_{:,\mathbf{I}(j)}) \frac{\partial \mathbf{S}_{:,j}(\mathbf{C})}{\partial \mathbf{C}_{:,j}}(\mathbf{C})$
end for
for $j = 1 \dots n$ do
    for $l = 1 \dots j-1$ do
        $\left(\frac{\partial \mathbf{X}(\mathbf{C})}{\partial \mathbf{C}}\right)_{:,j,l,:}(\mathbf{C}, \mathbf{X}^R) \coloneqq \frac{\partial \mathbf{X}_{:,j}(\mathbf{C})}{\partial \mathbf{C}_{:,l}}(\mathbf{C}, \mathbf{X}^R)$
            $= \frac{\partial \mathbf{B}_{:,:,j}(\mathbf{X}^R)}{\partial \mathbf{X}^R_{:,\mathbf{I}(j)}}(\mathbf{X}^R) \frac{\partial \mathbf{X}^R_{:,\mathbf{I}(j)}(\mathbf{C})}{\partial \mathbf{C}_{:,l}}(\mathbf{C}, \mathbf{X}^R) \cdot \mathbf{S}_{:,j}(\mathbf{C}) + \frac{\partial \mathbf{X}^R_{:,b(j)}(\mathbf{C})}{\partial \mathbf{C}_{:,l}}(\mathbf{C}, \mathbf{X}^R)$
            $= \left(\sum_{m_1=1}^{3} \sum_{k \in \{b(j),a(j),d(j)\}} \left(\frac{\partial \mathbf{B}(\mathbf{X}^R)}{\partial \mathbf{X}^R}(\mathbf{X}^R)\right)_{:,:,j,k,m_1} \left(\frac{\partial \mathbf{X}^R(\mathbf{C})}{\partial \mathbf{C}}(\mathbf{C}, \mathbf{X}^R)\right)_{m_1,k,l,:}\right) \cdot \mathbf{S}_{:,j}(\mathbf{C})$
                $+ \left(\frac{\partial \mathbf{X}^R(\mathbf{C})}{\partial \mathbf{C}}(\mathbf{C}, \mathbf{X}^R)\right)_{:,b(j),l,:}$
            $= \sum_{m_1=1}^{3} \sum_{k \in \{b(j),a(j),d(j)\}} \sum_{m_2=1}^{3} \left(\frac{\partial \mathbf{B}(\mathbf{X}^R)}{\partial \mathbf{X}^R}(\mathbf{X}^R)\right)_{:,m_2,j,k,m_1} \mathbf{S}_{m_2,j}(\mathbf{C}) \left(\frac{\partial \mathbf{X}^R(\mathbf{C})}{\partial \mathbf{C}}(\mathbf{C}, \mathbf{X}^R)\right)_{m_1,k,l,:}$
                $+ \left(\frac{\partial \mathbf{X}^R(\mathbf{C})}{\partial \mathbf{C}}(\mathbf{C}, \mathbf{X}^R)\right)_{:,b(j),l,:}$
    end for
end for
```

Transformation from cartesian to Z-matrix coordinates

The inverse function of $\mathbf{S}$ (equation 6) is given by:

$$\begin{aligned}
&\mathbf{S}^{-1}\colon \mathbb{R}^{3,n} \to ([0,\infty) \times [0,\pi) \times [-\pi,\pi))^n \\
&\mathbf{V}_{:,i} := \begin{pmatrix} x \\ y \\ z \end{pmatrix} \mapsto \mathbf{S}_{:,i} := \begin{pmatrix} \sqrt{x^2+y^2+z^2} \\ \operatorname{acos}\left(\frac{-z}{r}\right) \\ \operatorname{atan2}\left(\frac{-y}{r}, \frac{x}{r}\right) \end{pmatrix}
\end{aligned} \tag{10}$$

The affine-linear transformation $\mathbf{T}$ of atom i into the coordinate system spanned by its references is given by:

$$\begin{aligned}
&\mathbf{T}\colon \mathbb{R}^{3,n} \to \mathbb{R}^{3,n} \\
&\quad \mathbf{X}_{:,i} \mapsto \mathbf{T}_{:,i} = \mathbf{B}^{-1}_{:,:,i}\big(\mathbf{X^R}_{:,\mathbf{I}(i)}\big) \cdot \big(\mathbf{X}_{:,i} - \mathbf{X^R}_{:,b(i)}\big) \\
&\qquad\qquad = \mathbf{B}^{\mathsf{T}}_{:,:,i}\big(\mathbf{X^R}_{:,\mathbf{I}(i)}\big) \cdot \big(\mathbf{X}_{:,i} - \mathbf{X^R}_{:,b(i)}\big) \\
&\qquad\qquad = \mathbf{V}_{:,i}
\end{aligned} \tag{11}$$

The second equality is due to the orthogonality of $\mathbf{B}$. This gives for the bond, angle and dihedral of the i-th atom the following equation:

$$\begin{aligned}
\mathbf{C}_{:,i} &= \mathbf{S}^{-1}_{:,i}(\mathbf{T}_{:,:,i}(\mathbf{X}_{:,i})) \\
&= \mathbf{S}^{-1}_{:,i}\left(\mathbf{B}^{\mathsf{T}}_{:,:,i}\big(\mathbf{X^R}_{:,\mathbf{I}(i)}\big) \cdot \big(\mathbf{X}_{:,i} - \mathbf{X^R}_{:,b(i)}\big)\right)
\end{aligned} \tag{12}$$

Note that the use of the atan2 function in equation 10 automatically solves the pitfall of an undefined dihedral discussed in the previous section.

Gradient for the transformation from cartesian to Z-matrix coordinates

The aim is to calculate the derivative of bonds, angles, and dihedrals in the Z-matrix with respect to x, y, and z coordinates in cartesian space. Using the defined variables of the previous subsection the derivative of equation 12 can be written as:

$$\frac{\partial \mathbf{C}_{:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,l}}(\mathbf{X}) = \sum_{m_1=1}^{n} \left(\frac{\partial \mathbf{S}^{-1}_{:,j}(\mathbf{V}_{:,m_1})}{\partial \mathbf{V}_{:,m_1}}(\mathbf{T}_{:,:,m_1}(\mathbf{X}_{:,m_1})) \cdot \frac{\partial \mathbf{T}_{:,m_1}}{\partial \mathbf{X}_{:,l}}(\mathbf{X}_{:,l}) \right)$$

A closer inspection of equation 10 reveals that the j-th column of $\mathbf{S}^{-1}$ solely depends on the j-th column of the argument matrix $\mathbf{V}$:

$$\begin{aligned}
\frac{\partial \mathbf{C}_{:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,l}}(\mathbf{X}) = {}& \frac{\partial \mathbf{S}^{-1}_{:,j}(\mathbf{V}_{:,j})}{\partial \mathbf{V}_{:,j}}(\mathbf{T}_{:,:,j}(\mathbf{X}_{:,j})) \\
&\cdot \frac{\partial \mathbf{T}_{:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,l}}(\mathbf{X}_{:,l})
\end{aligned} \tag{13}$$

The first factor in equation 13 is obtained by straightforwardly deriving equation 10. For a succinct notation it is necessary to define the following variables:

$$\begin{aligned}
(x,y,z)^{\mathsf{T}} &:= \mathbf{v} := \mathbf{V}_{:,j} \\
r &:= \sqrt{x^2+y^2+z^2}
\end{aligned}$$

This gives:

$$\begin{aligned}
\frac{\partial \mathbf{S}^{-1}_{:,j}(\mathbf{V}_{:,j})}{\partial \mathbf{V}_{:,j}}(\mathbf{T}_{:,:,j}(\mathbf{X}_{:,j})) &= \frac{\partial \mathbf{S}^{-1}_{:,j}(\mathbf{v})}{\partial \mathbf{v}}(\mathbf{v}) \\
&= \begin{pmatrix} \frac{x}{r} & \frac{y}{r} & \frac{z}{r} \\ \frac{-xz}{r^2\sqrt{x^2+y^2}} & \frac{-yz}{r^2\sqrt{x^2+y^2}} & \frac{\sqrt{x^2+y^2}}{r^2} \\ \frac{y}{x^2+y^2} & \frac{-x}{x^2+y^2} & 0 \end{pmatrix}
\end{aligned} \tag{14}$$

For $x = y = 0$ a removable singularity exists, which is due to the undefined dihedral case from Figure 1a. Removing this singularity yields for $x = y = 0$:

$$\begin{pmatrix} 0 & 0 & 1 \\ -\frac{1}{z} & -\frac{1}{z} & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

The second factor in equation 13 may be derived by applying the product rule on equation 11.

$$
\begin{aligned}
&\frac{\partial \mathbf{T}_{:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,l}}(\mathbf{X}_{:,l}) \\
&= \frac{\partial \mathbf{B}^{\mathsf{T}}_{:,:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,l}}\left(\mathbf{X^R}_{:,\mathbf{I}(j)}\right) \cdot \left(\mathbf{X}_{:,j} - \mathbf{X^R}_{:,b(j)}\right) \\
&\quad + \mathbf{B}^{\mathsf{T}}_{:,:,j}\left(\mathbf{X^R}_{:,\mathbf{I}(j)}\right) \\
&\qquad \cdot \left(\frac{\partial \mathbf{X}_{:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,l}}(\mathbf{X}_{:,l}) - \frac{\partial \mathbf{X^R}_{:,b(j)}(\mathbf{X})}{\partial \mathbf{X}_{:,l}}(\mathbf{X}_{:,l})\right)
\end{aligned}
\tag{15}
$$

One can then insert equations 14 and 15 into 13. Again this leads to three different cases depending on $(j < l)$, $(j = l)$, or $(j > l)$. Each case is discussed individually in Appendix B and yields four working equations (equations 24, 25, 26, 27, and 28). The derivative $\frac{\partial \mathbf{S}^{-1}(\mathbf{v})}{\partial \mathbf{v}}$ is given by equation 14. The symbolically evaluated derivatives $\frac{\partial \mathbf{B}^{\mathsf{T}}(\mathbf{X}^R)}{\partial \mathbf{X}^R}$ are given in the supporting information if the transposition of the first two indices is taken into account. With all equations at hand we can formulate the combined algorithm 3.

Non-linearly constrained optimisation

In order to find the equilibrium geometry of a molecule, its potential hyper surface (PHS) is optimized in Z-matrix coordinates. This requires a potential energy gradient expressed in internal coordinates. To do so the potential energy is written as V_X if it is a function of cartesian coordinates and as V_C if it is a function of Z-matrix coordinates. Furthermore denoting the Hadamard product with $\odot$ and the sum of all matrix elements with $\sum^{\text{all}}$ allows to write:

$$
\begin{aligned}
&\left(\frac{\partial V_C(\mathbf{C})}{\partial \mathbf{C}}(\mathbf{C}_0)\right)_{i,j} \\
&= \sum^{\text{all}} \left(\left(\frac{\partial V_X(\mathbf{X})}{\partial \mathbf{X}}(\mathbf{X}(\mathbf{C}_0))\right)^{\mathsf{T}} \right. \\
&\qquad \left. \odot \left(\frac{\partial \mathbf{X}(\mathbf{C})}{\partial \mathbf{C}}(\mathbf{C}_0)\right)_{:,:,i,j} \right)
\end{aligned}
\tag{16}
$$

For the purpose of abstracting equation 16 to include non-linear constraints, the vector $\mathbf{p}$ is defined as an arbitrary length vector containing only differentiable expressions. A Z-matrix $\mathbf{C}$ containing symbolic expressions can then be written as function $\mathbf{C}(\mathbf{p})$. The Z-matrix in Table 2 is for example a function of the two-dimensional vector $(t, r_{CH})^{\mathsf{T}}$. The potential function V_p expressed in terms of the parameter vector $\mathbf{p}$ is then:

$$
\begin{aligned}
V_p &:= V_C \circ C = V_X \circ \mathbf{X} \circ \mathbf{C} \\
V_p(\mathbf{p}) &:= V_X(\mathbf{X}(\mathbf{C}(\mathbf{p})))
\end{aligned}
\tag{17}
$$

Deriving this function for the m-th entry in $\mathbf{p}$ yields:

$$
\begin{aligned}
&\frac{\partial V_p(\mathbf{p})}{\partial \mathbf{p}_m}(\mathbf{p}_0) \\
&= \sum_{i=1}^{3}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{3} \left(\left(\frac{\partial V_X(\mathbf{X})}{\partial \mathbf{X}}(\mathbf{X}(\mathbf{C}(\mathbf{p}_0)))\right)_{j,i} \cdot \left(\frac{\partial \mathbf{X}(\mathbf{C})}{\partial \mathbf{C}}(\mathbf{C}(\mathbf{p}_0))\right)_{i,j,k,l} \cdot \left(\frac{\partial \mathbf{C}(\mathbf{p})}{\partial \mathbf{p}}(\mathbf{p}_0)\right)_{l,k,m} \right) \\
&= \sum_{i=1}^{3}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{3} \left(\left(\frac{\partial V_X(\mathbf{X})}{\partial \mathbf{X}_{i,j}}(\mathbf{X}(\mathbf{C}(\mathbf{p}_0)))\right) \cdot \left(\frac{\partial \mathbf{X}_{i,j}(\mathbf{C})}{\partial \mathbf{C}_{l,k}}(\mathbf{C}(\mathbf{p}_0))\right) \cdot \left(\frac{\partial \mathbf{C}_{l,k}(\mathbf{p})}{\partial \mathbf{p}_m}(\mathbf{p}_0)\right) \right)
\end{aligned}
\tag{18}
$$

The first factor $\frac{\partial V_X(\mathbf{X})}{\partial \mathbf{X}}$ is returned from electronic calculations and the second factor $\frac{\partial \mathbf{X}(\mathbf{C})}{\partial \mathbf{C}}$ was derived in this paper. For the factor $\frac{\partial \mathbf{C}(\mathbf{p})}{\partial \mathbf{p}}$, we note the previous restriction that only dif-

Algorithm 3 Calculation of the analytical gradient for the coordinate transformation from cartesian to Z-matrix coordinates

```
for $j = 1 \dots n$ do
    for $l = 1 \dots n$ do
        if $l \notin \{j, b(j), a(j), d(j)\}$ then
                        ▷ Alternatively the whole array may be initialized with zeros.
            $\left(\frac{\partial \mathbf{C}(\mathbf{X})}{\partial \mathbf{X}}\right)_{:,j,l,:}(\mathbf{X}) \coloneqq 0$
        end if
    end for
end for
for $j = 1 \dots n$ do
    $\frac{\partial \mathbf{C}_{:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,j}}(\mathbf{X}) \coloneqq \frac{\partial \mathbf{S}^{-1}_{:,j}(\mathbf{V}_{:,j})}{\partial \mathbf{V}_{:,j}}(\mathbf{T}_{:,:,j}(\mathbf{X}_{:,j})) \cdot \mathbf{B}^{\mathsf{T}}_{:,:,j}\left(\mathbf{X^R}_{:,\mathbf{I}(j)}\right)$

    $\frac{\partial \mathbf{C}_{:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,b(j)}}(\mathbf{X}) \coloneqq \frac{\partial \mathbf{S}^{-1}_{:,j}(\mathbf{V}_{:,j})}{\partial \mathbf{V}_{:,j}}(\mathbf{T}_{:,:,j}(\mathbf{X}_{:,j}))$
        $\cdot \left(\frac{\partial \mathbf{B}^{\mathsf{T}}_{:,:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,b(j)}}\left(\mathbf{X^R}_{:,\mathbf{I}(j)}\right)\left(\mathbf{X}_{:,j} - \mathbf{X^R}_{:,b(j)}\right) - \mathbf{B}^{\mathsf{T}}_{:,:,j}\left(\mathbf{X^R}_{:,\mathbf{I}(j)}\right)\right)$
    $= \frac{\partial \mathbf{S}^{-1}_{:,j}(\mathbf{V}_{:,j})}{\partial \mathbf{V}_{:,j}}(\mathbf{T}_{:,:,j}(\mathbf{X}_{:,j}))$
        $\cdot \left(\sum_{m_1=1}^{3}\left(\left(\frac{\partial \mathbf{B}(\mathbf{X})}{\partial \mathbf{X}}\left(\mathbf{X^R}_{:,\mathbf{I}(j)}\right)\right)_{m_1,:,j,b(j),:}\left(\mathbf{X}_{m_1,j} - \mathbf{X^R}_{m_1,b(j)}\right)\right)\right.$
        $\left. - \mathbf{B}^{\mathsf{T}}_{:,:,j}\left(\mathbf{X^R}_{:,\mathbf{I}(j)}\right)\right)$

    $\frac{\partial \mathbf{C}_{:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,a(j)}}(\mathbf{X}) \coloneqq \frac{\partial \mathbf{S}^{-1}_{:,j}(\mathbf{V}_{:,j})}{\partial \mathbf{V}_{:,j}}(\mathbf{T}_{:,:,j}(\mathbf{X}_{:,j})) \cdot \left(\frac{\partial \mathbf{B}^{\mathsf{T}}_{:,:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,a(j)}}\left(\mathbf{X^R}_{:,\mathbf{I}(j)}\right)\left(\mathbf{X}_{:,j} - \mathbf{X^R}_{:,b(j)}\right)\right)$
    $= \frac{\partial \mathbf{S}^{-1}_{:,j}(\mathbf{V}_{:,j})}{\partial \mathbf{V}_{:,j}}(\mathbf{T}_{:,:,j}(\mathbf{X}_{:,j}))$
        $\cdot \left(\sum_{m_1=1}^{3}\left(\left(\frac{\partial \mathbf{B}(\mathbf{X})}{\partial \mathbf{X}}\left(\mathbf{X^R}_{:,\mathbf{I}(j)}\right)\right)_{m_1,:,j,a(j),:}\left(\mathbf{X}_{m_1,j} - \mathbf{X^R}_{m_1,b(j)}\right)\right)\right)$

    $\frac{\partial \mathbf{C}_{:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,d(j)}}(\mathbf{X}) \coloneqq \frac{\partial \mathbf{S}^{-1}_{:,j}(\mathbf{V}_{:,j})}{\partial \mathbf{V}_{:,j}}(\mathbf{T}_{:,:,j}(\mathbf{X}_{:,j})) \cdot \left(\frac{\partial \mathbf{B}^{\mathsf{T}}_{:,:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,d(j)}}\left(\mathbf{X^R}_{:,\mathbf{I}(j)}\right)\left(\mathbf{X}_{:,j} - \mathbf{X^R}_{:,b(j)}\right)\right)$
    $= \frac{\partial \mathbf{S}^{-1}_{:,j}(\mathbf{V}_{:,j})}{\partial \mathbf{V}_{:,j}}(\mathbf{T}_{:,:,j}(\mathbf{X}_{:,j}))$
        $\cdot \left(\sum_{m_1=1}^{3}\left(\left(\frac{\partial \mathbf{B}(\mathbf{X})}{\partial \mathbf{X}}\left(\mathbf{X^R}_{:,\mathbf{I}(j)}\right)\right)_{m_1,:,j,d(j),:}\left(\mathbf{X}_{m_1,j} - \mathbf{X^R}_{m_1,b(j)}\right)\right)\right)$
end for
```

ferentiable expressions are allowed as Z-matrix entries and use a symbolic algebra package. For the usual applications in chemistry the restriction of differentiability should be of no concern.

With the potential function from equation 17 and its gradient from equation 18 the BFGS algorithm or other quasi-Newton methods can be used to optimize a Z-matrix with non-linear contraints to reduce the degrees of freedom.[21]

Application examples

Starting guesses for NEB calculations

The Nudged Elastic Band (NEB) method approximates minimum energy paths (MEPs) connecting minimum structures as a string of discrete intermediate structures ("images"). To perform an NEB calculation, starting guesses for the entire string of structures are required. At the time of writing, most NEB implementations generate per default the initial pathway by interpolation in cartesian atomic coordinates. An interpolation is performed between the two minima structures, potentially aided through a lower level theory or added potential. However, when trying to model changes in bond angles or rotations of molecule parts, cartesian interpolations will often generate structures with compressed bonds or overlapping atoms. Since such molecular movements are common in reaction pathways, these shortcomings of the cartesian approach will often delay NEB convergence, or even prevent it altogether, when overlapping atoms cause the calculations of one or more structures to fail. Methods have been developed to reduce this problem, such as Image-Dependent Pair Potential (IDPP),[17] which performs a pre-optimization of the initial cartesian interpolation, reducing atom overlaps and bond deformations.

One of the advantages we observe in the method outlined in this paper is the possibility to create structure interpolations in internal coordinates (IC) in an automated fashion. Using the same construction table for both minimum structures, their Z-matrices can be interpolated just as easily as their cartesian coordinates. Interpolations generated this way can effortlessly model changes in bond angles and rotations of large molecule parts, resulting in interpolated structures that are much more chemically reasonable.

If two Z-matrices use the same construction table their changes of bond lengths, angles, and dihedrals can be interpolated almost as in cartesian coordinates and give chemically reasonable starting structures. The only difference to cartesian coordinates are the equivalence relations ($\sim$) for certain combinations of angles (α) and dihedrals (δ)

$$\begin{aligned}(\alpha, \delta) &\sim (-\alpha, \delta + \pi) \\ \alpha &\sim \alpha + n2\pi \quad n \in \mathbb{N} \\ \delta &\sim \delta + n2\pi \quad n \in \mathbb{N}\end{aligned} \tag{19}$$

the interpolation has to be adjusted to select a chemical representation of these equivalence classes. For example a change of $-5°$ is preferred over $355°$ and a movement from (α, δ) to $(-\alpha, \delta)$ happens usually along the dihedral coordinate. For this reason it is necessary before the interpolation to select combinations of angles and dihedrals such that $0 \leq \alpha \leq \pi$ and $-\pi \leq \delta \leq \pi$. After interpolation the angle and dihedral with the smallest absolute value should be chosen (yielding a change of $-5°$ over $355°$). These steps can be completely automatized.

To compare this new approach with the existing cartesian-based initial guesses, NEB calculations were performed on several chemical reactions, each once starting with an IC interpolation, a simple cartesian interpolation, and an IDPP-preoptimized cartesian interpolation, as described in the paper of Smidstrup et al.[17] The reactions are shown in Figure 3. Reactions **R1**–**R3** provide classical examples of organic chemistry reactivity, with the dimerization of cyclopentadiene (**R1**),[22] the Alder-ene reaction between ethene and propene (**R2**)[23] and the cleavage of sulfolene (**R3**).[24] Reactions **R4**–**R5** are two selected nucleophilic reactions of carbocations.[25] The last three reactions **R6**–**R8** are rearrangement reactions in molecular dimers without bond breaking/formation events. One should note that **R6** corresponds to a conversion from an O-bonding to a π-bonding dimer of furan and methanol, while **R7** should describe the translation of methanol from one side of the furan molecule to the other.[26,27] Albeit not a thorough benchmark set, it does provide some common tasks one faces in reaction path optimisations. The theory level used for the calculation of the **R7** reaction (see Figure 2) was PBEh-3c,[28] and for all other calculations, it was HF-3c.[29] Climbing Image[30] was used in conjunction with variable spring constants,[31] as described by Henkelman and Jónsson. In all cases we made use of eleven intermediate points (or “images”).

The optimization algorithm used was a modified version of Momentum Gradient Descent.[32] The related equations, as well as the definitions of its parameters can be found in appendix C. The parameterisation used in this work is provided in the Supporting Information. Our NEB implementation was interfaced to Orca,[33] with the minima also provided by the software package. Further computational details are provided in the Supporting Information.

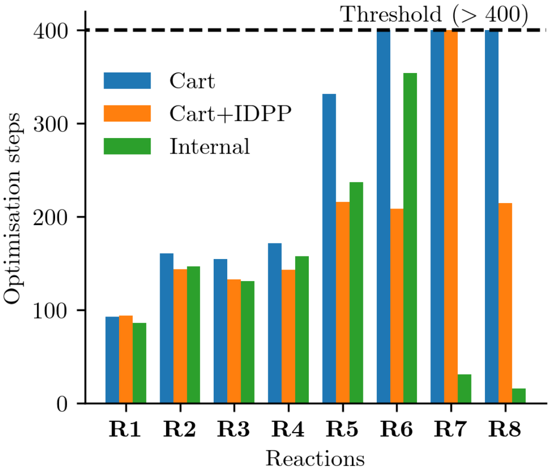


Figure 2: Comparison of individual optimisation steps for the three different choices of initial NEB interpolation: cartesian (Cart), cartesian+IDPP (Cart+IDPP) pre-optimisation and our suggested Z-matrix interpolation (Internal). In order to keep the figure easily readable, we set as maximum threshold for optimisation 400 iterations.

The optimal choice of interpolation method will, of course, strongly vary on the reaction under study. Some paths will be easier to represent in cartesian space (such as the carbocation reactions **R4**–**R5**), others are more assessible with an adequate selection of internal coordinates. As previously noted, there were three sets of initial pathways: simple cartesian space interpolation (Cartesian), the same definition as in the latter case but with a preoptimisation of the path with IDPP (Cartesian+IDPP) and finally with our own Z-matrix definition (Internal). The number of individual optimisation steps (gradient+energy calculations) required to converge the NEB paths in dependence of the method used for the initial pathway are provided in Figure 2. In some cases convergence could not be achieved when setting the limit to 400 individual steps. For many reactions, the performance of cartesian and internal coordinates is rather similar. It should be noted, nonetheless, that there is not a single reaction whereby the cartesian coordinate interpolation is superior (less iterations). The use of IDPP closes the gap rather efficiently, producing numbers very close or even lower than the internal coordinates result (4 out of 8). The most noticeable differences are observed in reactions whereby the main atomic movements can be best described through rotations, angles or dihedral torsions. This is the case for the water dimer and the two reactions involving the methanol-furan dimer (Mefur), whereby cartesian coordinates produce a nonsensical interpolation. The reason for this difference is visible when comparing the initial paths from both sets (see Figures 4 and 5). In the cartesian case one can easily encounter overlapping atoms as the algorithm cannot distinguish bonded from non-bonded atoms. In the case of **R7** and **R8** the internal coordinates interpolation is in fact very close to the converged final pathway. One should note that we have not in any way influenced the Z-matrix definitions. The procedure is completely black box, as it defines a Z-matrix for the starting structure and applies the same construct to the product state automatically. Given that these IC-interpolations are already competitive with IDPP corrected pathways, one can expect that this performance can be still significantly improved.

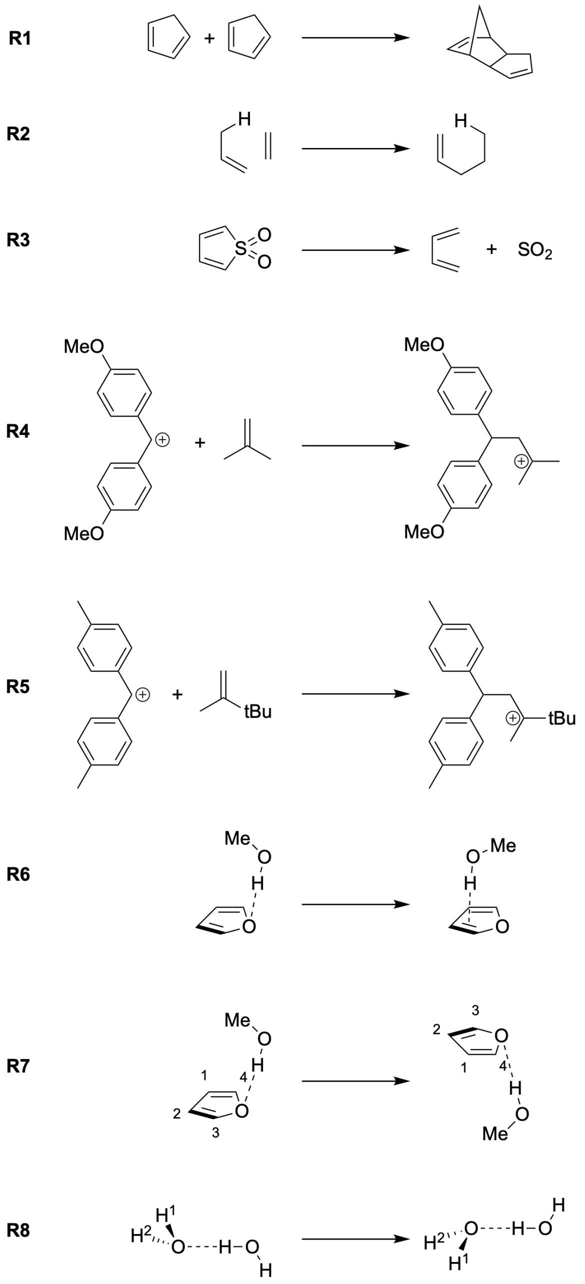


Figure 3: Reactions featured in the NEB calculations, with the corresponding short designation.

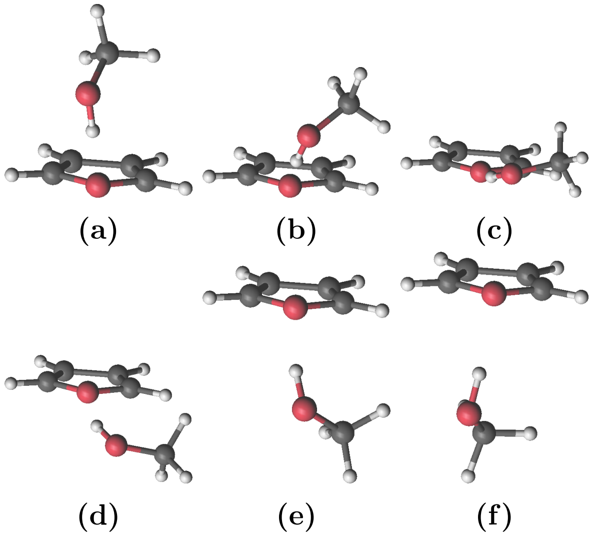


Figure 4: Movement of Methanol relative to Furan interpolated in Z-matrix coordinates (reaction **R7**).

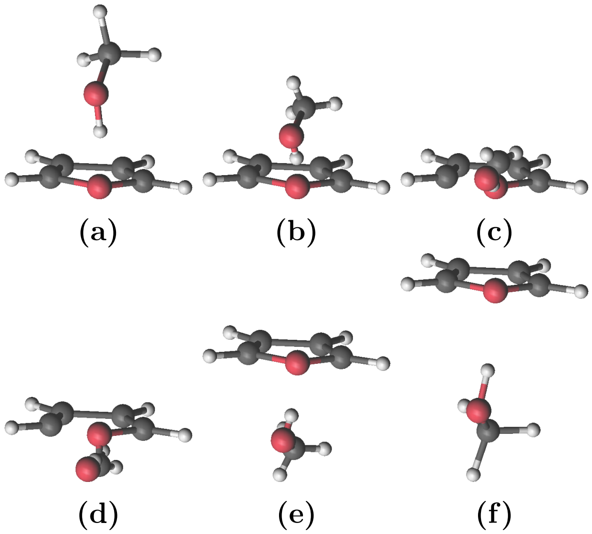


Figure 5: Movement of Methanol relative to Furan interpolated in cartesian coordinates (reaction **R7**).

Geometry optimisation

In the following section we want to apply equation 18 which yields gradients of arbitrary differentiable symbolic expressions to perform non-linearly constrained optimisation in internal coordinates.

One example is the highly symmetric cubane where the cubic symmetry can be imposed as in the example Z-matrix depicted in Table 1. By optimizing for r_{CC}, r_{CH}, and α the problem size is reduced to three degrees of freedom and cubic symmetry is automatically preserved.

Table 1: Z-matrix of cubane with symmetry constraints. Bond lengths in Å, angles in degrees.

	atom	b	bond	a	angle	d	dihedral
7	C	$\vec{0}$	0.0	$\vec{e_z}$	0.0	$\vec{e_x}$	0.0
9	C	7	r_{CC}	$\vec{e_z}$	0.0	$\vec{e_x}$	0.0
2	C	7	r_{CC}	9	90.0	$\vec{e_x}$	0.0
5	C	7	r_{CC}	9	90.0	2	-90.0
3	C	2	r_{CC}	7	90.0	9	-0.0
12	C	5	r_{CC}	7	90.0	9	0.0
1	C	5	r_{CC}	7	90.0	9	90.0
4	C	1	r_{CC}	5	90.0	7	-90.0
6	H	1	r_{CH}	5	$\frac{180\alpha}{\pi}$	7	135.0
8	H	2	r_{CH}	7	$\frac{180\alpha}{\pi}$	9	135.0
10	H	7	r_{CH}	2	$\frac{180\alpha}{\pi}$	9	-135.0
11	H	3	r_{CH}	2	$\frac{180\alpha}{\pi}$	7	135.0
16	H	9	r_{CH}	7	$\frac{180\alpha}{\pi}$	2	-135.0
13	H	5	r_{CH}	7	$\frac{180\alpha}{\pi}$	9	-135.0
14	H	4	r_{CH}	1	$\frac{180\alpha}{\pi}$	5	-135.0
15	H	12	r_{CH}	5	$\frac{180\alpha}{\pi}$	7	-135.0

The electronic structure calculations were Hartree Fock in a STO-3G basis set and were performed with OpenMolcas.[34] The analytic gradients in cartesian coordinates were transformed using equation 18 and the BFGS algorithm implementation from SciPy was used to guess the structure for the next iteration.[21] The geometry was considered to be converged, if the maximum norm of the gradient was less than $5 \cdot 10^{-4}$ and the absolute change of energy was less than $1 \cdot 10^{-6}\,E_{\mathrm{h}}$. With starting values of $r_{\mathrm{CC}} = 1.4\,\text{Å}$, $r_{\mathrm{CH}} = 1\,\text{Å}$, and $\alpha = \frac{120°}{180°}\pi$ the optimisation of cubane converged after 9 iterations to $r_{\mathrm{CC}} = 1.618\,\text{Å}$, $r_{\mathrm{CH}} = 1.159\,\text{Å}$, and $\alpha = \frac{125.27°}{180°}\pi$.

Another example is given by the Z-matrix (Table 2) of 2-methylpropane. The parameter t denotes a coupled movement of C−C stretches and C−C−C bends with one C−C bond growing exponentially. The electronic structure calculation uses the same parameters as before and with starting values of $t = 0$ and $r_{\mathrm{CH}} = 1\,\text{Å}$ the calculation converges after 10 iterations to $t = -0.054$ and $r_{\mathrm{CH}} = 1.089\,\text{Å}$. It has to be stressed that it would be very complicated to define similar constraints in cartesian coordinates. The parameter t appears both in the bond lengths and angles; subsequent hydrogens in Table 2 depend on the previous carbon atoms. Both facts imply that in cartesian coordinates the constraints would have to appear in the x, y, and z coordinates of nearly every atom and would involve complicated trigonometric expressions, while the same constraints in Z-matrix coordinates (Table 2) are user-friendly to define and read.

Constraints as in Table 2 can be used to enforce e.g. symmetry or certain paths in the search of a transition state. If there are no constraints and all internal degrees of freedom are to be optimized, our method is unfortunately inferior to the unconstrained optimization in redundant internal coordinates. One reason is that usual approximations and perfomance enhancements such as a model Hessian or step restrictions based on chemical knowledge cannot be applied.[15,16] There is no “one size fits all” approximation if arbitrary symbolic constraints are to be optimized as in the previous examples. One would require guess force constants for the computed internal coordinates. In the future these values could potentially be tabulated and interpolated and/or partially be derived from the aforementioned model Hessians. Instead of the traditional quasi-Newton algorithms the constrained optimization could be also coupled with new techniques like Kriging.[35,36]

Table 2: Z-matrix of 2-methylpropane with non-linear constraints. Bond lengths in Å, angles in degrees.

	atom	b	bond	a	angle	d	dihedral
5	C	$\vec{0}$	0.3728	$\vec{e_z}$	66.392	$\vec{e_x}$	-100.576
2	C	5	$e^t + 1.5$	$\vec{e_z}$	95.608	$\vec{e_x}$	-10.577
11	C	5	$t + 1.5$	2	$180/\pi \arcsin{(t)} + 110$	$\vec{e_x}$	-99.902
7	C	5	$t + 1.5$	2	$180/\pi \arcsin{(t)} + 110$	11	122.638
6	H	5	r_{CH}	2	108.405	11	-118.668
1	H	2	r_{CH}	5	111.344	11	178.701
3	H	2	r_{CH}	5	111.575	11	-61.305
4	H	2	r_{CH}	5	111.338	11	58.672
12	H	11	r_{CH}	5	111.582	2	61.294
13	H	11	r_{CH}	5	111.348	2	-58.695
14	H	11	r_{CH}	5	111.348	2	-178.720
8	H	7	r_{CH}	5	111.343	2	58.688
9	H	7	r_{CH}	5	111.567	2	-61.303
10	H	7	r_{CH}	5	111.344	2	178.716

Conclusions

With the growing interest in automated procedures for the exploration of chemical space, it becomes more than ever pertinent to develop a robust, chemically meaningful ansatz for the definition of internal coordinates. In this work, we provide such an algorithm, overcoming the pitfalls commonly encountered when converting from the cartesian to the internal coordinate space and backwards. Analytic transformations of gradients from cartesian to Z-matrix coordinates are derived for the first time, improving on existing numerical algorithms for this task. In combination with symbolic algebra packages, the analytic gradients can be used for non-linearly constrained geometry optimisations in Z-matrix space.

In order to illustrate the reliability of our procedure, we use it to generate starting pathways for nudged elastic band calculations. Compared to cartesian space interpolation, our Z-matrix formulation is clearly superior. A comparison is also provided to refined pathways making use of the Image-Dependent Pair Potential (IDPP) method. The Z-matrix results, even without any type of post-processing, are still competitive and offer clear advantages for conversion processes in molecular dimers. This is all carried out without any previous knowledge of the chemical step in question, as the algorithm follows a priority derived from the connectivity matrix.

We also performed non-linearly constrained geometry optimisations on cubane and 2-methylpropane, to show that this new method allows contraints with arbitrary differentiable symbolic expressions. This can be used to enforce symmetries, to reduce degrees of freedom, or to apply chemical knowledge for searching transition states.

We believe this set of tools will be of great interest to the community. It also provides a step-by-step algorithm (open for discussion) of how we chemists interpret connectivity and envisage internal coordinates. So far the algorithm has reliably provided Z-matrices for a plethora of chemical systems, from single molecules, to non-covalent and coordination complexes. Input from the community will be highly valued going into the future.

Acknowledgement The authors thank Dr. Thorsten Stolper (Universität Göttingen), Dr. Rainer Oswald (Universität Göttingen), and Dr. Michael Willatt (MPI for Solid State Research) for helpful scientific discussions. OW acknowledges funding from the Max-Planck so-

ciety.

Appendix A - Discrimination of transformation cases from Z-matrix to cartesian coordinates

In this subsection we will discuss the three different cases $j < l$, $j = l$, and $j > l$ that appear in the gradient for the transformation from Z-matrix to cartesian coordinates. We repeat the corresponding equation 9

$$\begin{aligned}\frac{\partial \mathbf{X}_{i,j}(\mathbf{C})}{\partial \mathbf{C}_{k,l}}(\mathbf{C}) = {} & \underbrace{\mathbf{B}_{i,:,j}(\mathbf{X}^R_{:,\mathbf{I}(j)}) \frac{\partial \mathbf{S}_{:,j}(\mathbf{C})}{\partial \mathbf{C}_{k,l}}}_{A} \\ & + \underbrace{\frac{\partial \mathbf{B}_{i,:,j}(\mathbf{X}^R_{:,\mathbf{I}(j)})}{\partial \mathbf{X}^R_{:,\mathbf{I}(j)}}(\mathbf{X}^R_{:,\mathbf{I}(j)}(\mathbf{C})) \cdot \frac{\partial \mathbf{X}^R_{:,\mathbf{I}(j)}(\mathbf{C})}{\partial \mathbf{C}_{k,l}}(\mathbf{C}) \cdot \mathbf{S}_{:,j}(\mathbf{C})}_{B} \\ & + \underbrace{\frac{\partial \mathbf{X}^R_{i,b(j)}(\mathbf{C})}{\partial \mathbf{C}_{k,l}}(\mathbf{C})}_{C}\end{aligned}$$

and split it into summands A, B, and C to evaluate them separately.

Derivative with $j < l$

The variation of a Z-matrix entry can never affect the cartesian position of atoms previous to this entry. Therefore, the corresponding derivative has to be zero.

$$\forall\, i, j, k, l \in \{1 \dots n\} \wedge j < l : \frac{\partial \mathbf{X}_{i,j}(\mathbf{C})}{\partial \mathbf{C}_{k,l}} = 0 \qquad (20)$$

It should be noted that the derivative of absolute references after Z-matrix entries is always zero:

$$\forall\, j \in I^R \setminus I : \frac{\partial \mathbf{X}^R_{i,j}(\mathbf{C})}{\partial \mathbf{C}_{k,l}} = 0 \qquad (21)$$

Derivative with $j = l$

In this case we know, because of equation 4, that $l > b(l), a(l), d(l) = b(j), a(j), d(j)$. Together with equation 20 it may be concluded, that

$$\begin{aligned}\frac{\partial \mathbf{X}^R_{:,\mathbf{I}(j)}(\mathbf{C})}{\partial \mathbf{C}_{k,j}} &\equiv 0 \\ \frac{\partial \mathbf{X}^R_{i,b(j)}(\mathbf{C})}{\partial \mathbf{C}_{k,j}} &\equiv 0 \\ \Rightarrow \quad B \equiv C &\equiv 0\end{aligned}$$

which implies:

$$\frac{\partial \mathbf{X}_{i,j}(\mathbf{C})}{\partial \mathbf{C}_{k,j}} \equiv \mathbf{B}_{i,:,j}(\mathbf{X}^R_{:,\mathbf{I}(j)}) \frac{\partial \mathbf{S}_{:,j}(\mathbf{C})}{\partial \mathbf{C}_{k,j}} \quad . \qquad (22)$$

The only missing term is the gradient of $\mathbf{S}$ (equation 6) which is given by:

$$\frac{\partial \mathbf{S}_{:,j}(\mathbf{C})}{\partial \mathbf{C}_{:,j}}(\mathbf{C}_{:,j}) = \begin{pmatrix} \sin(\mathbf{C}_{2,j})\cos(\mathbf{C}_{3,j}) & \mathbf{C}_{1,j}\cos(\mathbf{C}_{2,j})\cos(\mathbf{C}_{3,j}) & -\mathbf{C}_{1,j}\sin(\mathbf{C}_{2,j})\sin(\mathbf{C}_{3,j}) \\ -\sin(\mathbf{C}_{2,j})\sin(\mathbf{C}_{3,j}) & -\mathbf{C}_{1,j}\sin(\mathbf{C}_{3,j})\cos(\mathbf{C}_{2,j}) & -\mathbf{C}_{1,j}\sin(\mathbf{C}_{2,j})\cos(\mathbf{C}_{3,j}) \\ -\cos(\mathbf{C}_{2,j}) & \mathbf{C}_{1,j}\sin(\mathbf{C}_{2,j}) & 0 \end{pmatrix}$$

Derivative with $j > l$

A closer inspection of equation 6 reveals, that the j-th column of $\mathbf{S}$ solely depends on the j-th column of the argument matrix. It may be concluded, that

$$\begin{aligned}\forall\, j \neq l : \frac{\partial \mathbf{S}_{i,j}(\mathbf{C})}{\partial \mathbf{C}_{k,l}} &\equiv 0 \\ \Rightarrow \quad A &\equiv 0\end{aligned}$$

which implies:

$$\begin{aligned}&\frac{\partial \mathbf{X}_{i,j}(\mathbf{C})}{\partial \mathbf{C}_{k,l}}(\mathbf{C}) \\ = &\frac{\partial \mathbf{B}_{i,:,j}(\mathbf{X}^{R}_{:,\mathbf{I}(j)})}{\partial \mathbf{X}^{R}_{:,\mathbf{I}(j)}}(\mathbf{X}^{R}_{:,\mathbf{I}(j)}(\mathbf{C})) \\ &\cdot \frac{\partial \mathbf{X}^{R}_{:,\mathbf{I}(j)}(\mathbf{C})}{\partial \mathbf{C}_{k,l}}(\mathbf{C}) \cdot \mathbf{S}_{:,j}(\mathbf{C}) \\ &+ \frac{\partial \mathbf{X}^{R}_{i,b(j)}(\mathbf{C})}{\partial \mathbf{C}_{k,l}}(\mathbf{C})\end{aligned} \tag{23}$$

Appendix B - Discrimination of transformation cases from cartesian to Z-matrix coordinates

In this subsection we will discuss the three different cases $j < l$, $j = l$, and $j > l$ that appear in the gradient for the transformation from cartesian to Z-matrix coordinates in equation 13.

Derivative with $l \notin \{j, b(j), a(j), d(j)\}$

The Z-matrix entries of the j-th atom can only change if itself or its reference frame moves in cartesian space. For this reason we can write

$$\forall\, l \notin \{j, b(j), a(j), d(j)\}\colon \left(\frac{\partial \mathbf{C}(\mathbf{X})}{\partial \mathbf{X}}\right)_{:,j,l,:} \equiv \mathbf{0} \quad . \tag{24}$$

Derivative with $l = j$

Deriving the reference frame of the j-th atom with respect to $\mathbf{v}_j$ itself, has to be zero

$$\frac{\partial \mathbf{B}^{\mathsf{T}}_{:,:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,j}} \equiv \mathbf{0} \quad .$$

The positions of each atom in cartesian space are independent of each other:

$$\frac{\partial {\mathbf{X}^{\mathbf{R}}}_{:,b(j)}(\mathbf{X})}{\partial \mathbf{X}_{:,j}} \equiv \mathbf{0} \quad .$$

The following equation is trivial:

$$\frac{\partial \mathbf{X}_{:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,j}} \equiv \mathbf{1} \quad .$$

Combining the three previous equations and inserting into 15 yields:

$$\frac{\partial \mathbf{T}_{:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,j}}(\mathbf{X}) = \mathbf{B}^{\mathsf{T}}_{:,:,j}\left({\mathbf{X}^{\mathbf{R}}}_{:,\mathbf{I}(j)}\right)$$

Hence by inserting into 13:

$$\begin{aligned}&\left(\frac{\partial \mathbf{C}(\mathbf{X})}{\partial \mathbf{X}}\right)_{:,j,j,:}(\mathbf{X}) = \frac{\partial \mathbf{C}_{:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,j}}(\mathbf{X}) \\ = &\left(\frac{\partial \mathbf{S}^{-1}_{:,j}(\mathbf{V}_{:,j})}{\partial \mathbf{V}_{:,j}}(\mathbf{T}_{:,:,j}(\mathbf{X}_{:,j}))\right) \cdot \mathbf{B}^{\mathsf{T}}_{:,:,j}\left({\mathbf{X}^{\mathbf{R}}}_{:,\mathbf{I}(j)}\right)\end{aligned} \tag{25}$$

Derivative with $l = b(j)$

With the same argumentation as before the following relationships hold:

$$\begin{aligned}\frac{\partial {\mathbf{X}^{\mathbf{R}}}_{:,b(j)}(\mathbf{X})}{\partial \mathbf{X}_{:,b(j)}} &\equiv \mathbf{1} \\ \frac{\partial \mathbf{X}_{:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,b(j)}} &\equiv \mathbf{0}\end{aligned}$$

By inserting into equation 15 one obtains:

$$\begin{aligned}&\frac{\partial \mathbf{T}_{:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,b(j)}}(\mathbf{X}) \\ = &\frac{\partial \mathbf{B}^{\mathsf{T}}_{:,:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,b(j)}}\left({\mathbf{X}^{\mathbf{R}}}_{:,\mathbf{I}(j)}\right)\left(\mathbf{X}_{:,j} - {\mathbf{X}^{\mathbf{R}}}_{:,b(j)}\right) - \mathbf{B}^{\mathsf{T}}_{:,:,j}\left({\mathbf{X}^{\mathbf{R}}}_{:,\mathbf{I}(j)}\right)\end{aligned}$$

$$\left(\frac{\partial \mathbf{C}(\mathbf{X})}{\partial \mathbf{X}}\right)_{:,j,b(j),:}(\mathbf{X}) = \frac{\partial \mathbf{C}_{:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,b(j)}}(\mathbf{X})$$
$$= \frac{\partial \mathbf{S}^{-1}_{:,j}(\mathbf{V}_{:,j})}{\partial \mathbf{V}_{:,j}}(\mathbf{T}_{:,:,j}(\mathbf{X}_{:,j})) \cdot \left(\frac{\partial \mathbf{B}^{\intercal}_{:,:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,b(j)}}\left(\mathbf{X^R}_{:,\mathbf{I}(j)}\right)\left(\mathbf{X}_{:,j} - \mathbf{X^R}_{:,b(j)}\right) - \mathbf{B}^{\intercal}_{:,:,j}\left(\mathbf{X^R}_{:,\mathbf{I}(j)}\right)\right) \tag{26}$$

Derivative with $l = a(j)$ or $l = d(j)$

With the same argumentation as before the following relationships hold:

$$\frac{\partial \mathbf{X^R}_{:,b(j)}(\mathbf{X})}{\partial \mathbf{X}_{:,a(j)}} \equiv \frac{\partial \mathbf{X^R}_{:,b(j)}(\mathbf{X})}{\partial \mathbf{X}_{:,d(j)}} \equiv \mathbf{0}$$
$$\frac{\partial \mathbf{X}_{:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,a(j)}} \equiv \frac{\partial \mathbf{X}_{:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,d(j)}} \equiv \mathbf{0}$$

It may be concluded by inserting into equation 15:

$$\frac{\partial \mathbf{T}_{:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,a(j)}}(\mathbf{X}) = \left(\frac{\partial \mathbf{B}^{\intercal}_{:,:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,a(j)}}\left(\mathbf{X^R}_{:,\mathbf{I}(j)}\right)\right) \cdot \left(\mathbf{X}_{:,j} - \mathbf{X^R}_{:,b(j)}\right)$$
$$\frac{\partial \mathbf{T}_{:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,d(j)}}(\mathbf{X}) = \left(\frac{\partial \mathbf{B}^{\intercal}_{:,:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,d(j)}}\left(\mathbf{X^R}_{:,\mathbf{I}(j)}\right)\right) \cdot \left(\mathbf{X}_{:,j} - \mathbf{X^R}_{:,b(j)}\right)$$

Which finally gives by inserting into equation 13:

$$\left(\frac{\partial \mathbf{C}(\mathbf{X})}{\partial \mathbf{X}}\right)_{:,j,a(j),:}(\mathbf{X}) = \frac{\partial \mathbf{C}_{:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,a(j)}}(\mathbf{X})$$
$$= \left(\frac{\partial \mathbf{S}^{-1}_{:,j}(\mathbf{V}_{:,j})}{\partial \mathbf{V}_{:,j}}(\mathbf{T}_{:,:,j}(\mathbf{X}_{:,j}))\right) \cdot \left(\frac{\partial \mathbf{B}^{\intercal}_{:,:,j}}{\partial \mathbf{X}_{:,a(j)}}\left(\mathbf{X^R}_{:,\mathbf{I}(j)}\right)\left(\mathbf{X}_{:,j} - \mathbf{X^R}_{:,b(j)}\right)\right) \tag{27}$$

and

$$\left(\frac{\partial \mathbf{C}(\mathbf{X})}{\partial \mathbf{X}}\right)_{:,j,d(j),:}(\mathbf{X}) = \frac{\partial \mathbf{C}_{:,j}(\mathbf{X})}{\partial \mathbf{X}_{:,d(j)}}(\mathbf{X})$$
$$= \left(\frac{\partial \mathbf{S}^{-1}_{:,j}(\mathbf{V}_{:,j})}{\partial \mathbf{V}_{:,j}}(\mathbf{T}_{:,:,j}(\mathbf{X}_{:,j}))\right) \cdot \left(\frac{\partial \mathbf{B}^{\intercal}_{:,:,j}}{\partial \mathbf{X}_{:,d(j)}}\left(\mathbf{X^R}_{:,\mathbf{I}(j)}\right)\left(\mathbf{X}_{:,j} - \mathbf{X^R}_{:,b(j)}\right)\right) \tag{28}$$

Appendix C - Momentum Gradient Descent and Modified Momentum Gradient Descent

In standard Momentum Gradient Descent, the step vector $\mathbf{s}_n$ at the n-th iteration is given as

$$\mathbf{s_n} = \gamma \cdot \mathbf{s_{n-1}} - \alpha \cdot \nabla E \tag{29}$$

where ∇E is the energy gradient, α is a step size (usually between 0 and 1), $\mathbf{s_{n-1}}$ is the step vector of the previous iteration, and γ is the friction coefficient - effectively also a stepsize. In the field of machine learning, where this algorithm was used, it usually has a value around 0.9.[37]

In the modified version of Momentum Gradient Descent used for the NEB calculations in the 'Application Examples'-section, γ is no longer a constant. Rather, it is now given as

$$\gamma(\nabla E,\ \mathbf{s_{n-1}},\ \gamma_{\max}) = \min\left[\gamma_{\max},\ \frac{1}{2}\left(1 + \frac{\langle -\nabla E, \mathbf{s_{n-1}}\rangle}{||\nabla E|| \cdot ||\mathbf{s_n}||}\right)\right], \tag{30}$$

where $\gamma_{\max}$ is the maximum gamma value. The step vector s_n is then calculated via

$$\mathbf{s_n'} = \gamma(\nabla E,\ \mathbf{s_{n-1}},\ \gamma_{\max}) \cdot \mathbf{s_{n-1}} - \alpha \cdot \nabla E \text{ , then}$$

$$\mathbf{s_n} = \begin{cases} \mathbf{s_n'} & \text{if } ||\mathbf{s_n'}|| \leq s_{\max} \\ \mathbf{s_n'} \cdot \frac{s_{\max}}{||\mathbf{s_n'}||} & \text{otherwise} \end{cases} . \quad (31)$$

$s_{\max}$ is the maximum step length.

Supporting Information Available

The Supporting Information contains an example for a failing conversion between Cartesian and Z-matrix coordinates, a description of coordinate and input files (`additional_files.zip`) to reproduce our results, and the analytical derivatives of the basis **B**.

References

(1) Pulay, P.; Fogarasi, G.; Pang, F.; Boggs, J. E. Systematic ab initio gradient calculation of molecular geometries, force constants, and dipole moment derivatives. *J. Am. Chem. Soc.* **1979**, *101*, 2550–2560.

(2) Pulay, P.; Fogarasi, G. Geometry optimization in redundant internal coordinates. *J. Chem. Phys.* **1992**, *96*, 2856–2860.

(3) Vaidehi, N.; Jain, A. Internal Coordinate Molecular Dynamics: A Foundation for Multiscale Dynamics. *J. Phys. Chem. B* **2015**, *119*, 1233–1242, PMID: 25517406.

(4) Schlegel, H. B. Estimating the hessian for gradient-type geometry optimizations. *Theor. Chim. Acta* **1984**, *66*, 333–340.

(5) Parsons, J.; Holmes, J. B.; Rojas, J. M.; Tsai, J.; Strauss, C. E. M. Practical Conversion from Torsion Space to Cartesian Space for in Silico Protein Synthesis. *J Comput Chem* **2005**, *26*, 1063–1068.

(6) AlQuraishi, M. Parallelized Natural Extension Reference Frame: Parallelized Conversion from Internal to Cartesian Coordinates. *J. Comput. Chem.* **2019**, *40*, 885–892.

(7) Jablonka, K. M.; Ongari, D.; Moosavi, S. M.; Smit, B. Big-Data Science in Porous Materials: Materials Genomics and Machine Learning. *Chem. Rev.* **2020**, *120*, 8066–8129.

(8) Friederich, P.; Häse, F.; Proppe, J.; Aspuru-Guzik, A. Machine-learned potentials for next-generation matter simulations. *Nat. Mater.* **2021**, *20*, 750–761.

(9) McConnell, S. R.; Kästner, J. Instanton Rate Constant Calculations Using Interpolated Potential Energy Surfaces in Nonredundant, Rotationally and Translationally Invariant Coordinates. *J. Comput. Chem.* **2019**, *40*, 866–874.

(10) Meyer, R.; Hauser, A. W. Geometry optimization using Gaussian process regression in internal coordinate systems. *J. Chem. Phys.* **2020**, *152*, 084112.

(11) Denzel, A.; Kästner, J. Gaussian Process Regression for Transition State Search. *J. Chem. Theory Comput.* **2018**, *14*, 5777–5786.

(12) Westermayr, J.; Gastegger, M.; Menger, M. F. S. J.; Mai, S.; González, L.; Marquetand, P. Machine learning enables long time scale molecular photodynamics simulations. *Chem. Sci.* **2019**, *10*, 8100–8107.

(13) Baker, J.; Kinghorn, D.; Pulay, P. Geometry Optimization in Delocalized Internal Coordinates: An Efficient Quadratically Scaling Algorithm for Large Molecules. *J. Chem. Phys.* **1999**, *110*, 4986–4991.

(14) Meurer, A. et al. SymPy: symbolic computing in Python. *PeerJ Comput. Sci.* **2017**, *3*, e103.

(15) Lindh, R.; Bernhardsson, A.; Karlström, G.; Malmqvist, P.-Å. On the Use

of a Hessian Model Function in Molecular Geometry Optimizations. *Chem. Phys. Lett.* **1995**, *241*, 423–428.

(16) Bakken, V.; Helgaker, T. The Efficient Optimization of Molecular Geometries Using Redundant Internal Coordinates. *J. Chem. Phys.* **2002**, *117*, 9160–9174.

(17) Smidstrup, S.; Pedersen, A.; Stokbro, K.; Jónsson, H. Improved Initial Guess for Minimum Energy Path Calculations. *J. Chem. Phys.* **2014**, *140*, 214106.

(18) Hoft, R. C.; Gale, J. D.; Ford, M. J. Implementation of a Z-Matrix Approach within the SIESTA Periodic Boundary Conditions Code and Its Application to Surface Adsorption. *Mol. Simul.* **2006**, *32*, 595–600.

(19) O'Boyle, N. M.; Banck, M.; James, C. A.; Morley, C.; Vandermeersch, T.; Hutchison, G. R. Open Babel: An Open Chemical Toolbox. *Journal of Cheminformatics* **2011**, *3*, 33.

(20) IUPAC, In *Compendium of Chemical Terminology ("the gold book")*; A. D. McNaught and A. Wilkinson.,, Ed.; Blackwell Scientific Publications, 1997.

(21) Nocedal, J.; Wright, S. J.; SpringerLink (Online service)., *Numerical Optimization*; 2006.

(22) Gao, S.; Simon, S. L. The reaction kinetics of cyclopentadiene dimerization using differential scanning calorimetry: Experiments and modelling. *Thermochim. Acta* **2014**, *589*, 241–246.

(23) Paderes, G. D.; Jorgensen, W. L. Computer-assisted mechanistic evaluation of organic reactions. 20. Ene and retro-ene chemistry. *J. Org. Chem.* **1992**, *57*, 1904–1916.

(24) Chou, T.-S.; Chang, S.-Y. A convenient method for the extrusion of sulfur dioxide from 2,2,5,5-tetrasubstituted 3-sulfolenes. *J. Chem. Soc., Perkin Trans. 1* **1992**, 1459–1460.

(25) Ammer, J.; Nolte, C.; Mayr, H. Free Energy Relationships for Reactions of Substituted Benzhydrylium Ions: From Enthalpy over Entropy to Diffusion Control. *J. Am. Chem. Soc.* **2012**, *134*, 13902–13911, PMID: 22839678.

(26) Gottschalk, H. C. et al. The furan microsolvation blind challenge for quantum chemical methods: First steps. *J. Chem. Phys.* **2018**, *148*, 014301.

(27) Gottschalk, H. C. et al. The first microsolvation step for furans: New experiments and benchmarking strategies. *J. Chem. Phys.* **2020**, *152*, 164303.

(28) Grimme, S.; Brandenburg, J. G.; Bannwarth, C.; Hansen, A. Consistent structures and interactions by density functional theory with small atomic orbital basis sets. *J. Chem. Phys.* **2015**, *143*, 054107.

(29) Sure, R.; Grimme, S. *J. Comput. Chem.* **2013**, *34*, 1672–1685.

(30) 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*.

(31) 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*.

(32) Qian, N. On the momentum term in gradient descent learning algorithms. *Neural Netw.* **1999**, *12*, 145–151.

(33) Neese, F. The ORCA program system. *Wiley Interdiscip. Rev.: Comput. Mol. Sci.* **2012**, *2*, 73–78.

(34) Fdez. Galván, I. et al. OpenMolcas: From Source Code to Insight. *J. Chem. Theory Comput.* **2019**, *15*, 5925–5964.

(35) Krige, D. G. A Statistical Approach to Some Basic Mine Valuation Problems on the Witwatersrand. *J. South. Afr. Inst. Min. Metall.* **1951**, *52*, 119–139.

(36) Raggi, G.; Galván, I. F.; Ritterhoff, C. L.; Vacher, M.; Lindh, R. Restricted-Variance Molecular Geometry Optimization Based on Gradient-Enhanced Kriging. *J. Chem. Theory Comput.* **2020**, *16*, 3989–4001.

(37) Qian, N. On the momentum term in gradient descent learning algorithms. *Neural Netw.* **1999**, *12*, 145–151.

Graphical TOC Entry