Exploring the Potential Energy Surfaces Using an Incremental Description

F. Mohamed

February 4, 2011

1 Potential energy surface

A large class of problems in chemistry can be studied looking at one or more potential energy surface (PES) $E(r)$. One can define this energy in various ways: classical models, semiempirical force fields, density functional theory, MP2, CI,..., and any combination of those (for example the minimal energy of various surfaces). What is important is that for every configuration r there is only one value, and that the surface is continuos everywhere, and differentiable almost everywhere, which assumes some kind of separation between the atomic and electronic states, as for example in the Born-Oppenheimer approximation.

The importance of the PES is well known, unfortunately it is almost always an hyper-surface in many dimensions, and thus difficult to analyze and understand.

A way to understand it is to look at special points on it, critical points, i.e. places where the derivative is 0: minima, saddle points, maxima, and local approximations close to them. Thus it is a common strategy to try to build a harmonic approximation close to minima and saddle-points. Local minimization has a very long history, but also the idea of exploring automatically the surface is not new [1], and is an active field of research [2].

Another way to explore the PES is to use some sampling strategy, that normally weights things using $e^{-\beta E}$, and thus allows one to calculate thermodynamical averages. If MD is used both dynamical properties or thermodynamical averages can be calculated.

In this article we concern ourselves with a method to describe the PES in a global and incremental way. Global fitting with easier to evaluate functions is a good strategy to be able to perform better sampling, but is not our aim, what we want is having a map of what we have explored that can be used to find new places to explore, or to gather information depending on the places that we have already explored.

2 Basic exploration strategies

To have a better idea of the kind of representation that is needed, we begin to describe some basic exploration strategies that we want to be able to perform.

MinE strategy: explore the place around the point with minimal energy, but never explore an already explored region. Such a strategy first finds the local minimum reachable from the current point, then fills up the current basin of attraction until the lowest saddle-point is found. Then it goes down into that minimum and fills it up, until it reaches the lowest saddle point. If this is the original one then it explores the combined basin of attraction of the minima until it finds the next lowest saddle-point, and continues from there. This exploration strategy is useful to explore for example chemical reaction networks when one knows the starting point, and wants to know what will happen. The idea is similar to the one presented in [3], even if the scanning order is closer to what the local elevator [4] or Metadynamics [5] do, but those methods potentially visit several times the same place of the configurational space (using the free energy they don't have an easily computable PES, only the forces are easily computable).

GoUp strategy: just like with *MinE* explore the place around the "free" point with lowest energy, but never go down in energy (i.e. along the forces). Explore at the same energy (perpendicular to the forces) only if there are point lower in energy close-by, otherwise explore only toward higher energies. This way one never goes down from a saddle-point in the next basin of attraction. Thus if started from a point, it explores mostly in the direction of the point going up in energy finding all saddle points. If at the beginning one explores directions at the same energy than the initial point (or starts after having explored a bit the minimum) this strategy explores the whole basin of attraction, and not just the region on the side of the starting point. This is useful to understand all possible reactions that can happen starting from a compound up to a given maximal energy.

3 Reducing the degrees of freedom

It is clear that the sketched exploration strategies can be very useful, but one should note that both of them do not cope well with flat PES, or not relevant degrees of freedom: as the whole basin is explored, all "flat" degrees of freedom are explored before going to higher energies. On one side this is good, because then one is sure that the PES is really fully explored, and no special configuration of these degrees of freedom gives a lower saddle point, a different reaction. The exploration is exhaustive, and gives some guarantee that what has been explored is really the relevant part. On the other side if the degrees of freedom really have little or no influence, then one can waste lot of resources, in the worst case never progress (for a translational invariant system if the translation degrees of freedom are explored one begins to explore different translations of the system).

A way to simplify the PES and at the same time get rid of this problem is to

try to reduce the number of coordinates to few "relevant" degrees of freedom. Once one has defined the relevant coordinates the question is what to do with the "non-relevant" ones. There are a few possibilities:

1. *freeze* them, this is the cheapest choice, and is correct for translational degrees of freedom, and in some cases for distant parts of a rigid system.

2. *minimimze* them, this is more expensive, and might give problems if several minima are present, but is a better choice in some cases.

3. *average* them, if the number of degrees of freedom is large, and their contribution is small (for example far away molecules of a solvent) it might be much more efficient to calculate their average contribution than to really explore them. In this case a thermodynamic average, that using $e^{-\beta E}$ weights heavily the lowest energy configurations, and has a well defined physical interpretation, and is well studied, is particularly attractive. A good method is for example [6]. The averaging process gives a noise on the gathered results, one can think about methods decide how much noise is acceptable in various places, and could thereby potentially save much computational effort, but in this article we just assume that the noise is small enough.

4 General coordinates

Chemists use internal coordinates, as these remove the invariant degrees of freedom (translation, rotation) and are well suited for small molecules. For larger molecules the generation of good internal coordinates is not simple, thus redundant coordinates are often used. Also the definition good relevant degrees of freedom often requires the definition of general coordinates $c_i(r)$. To treat general coordinates there are several ways, we use an approach close to the differential geometry approach, and discuss the issues that can arise and should be handled by a robust algorithm.

At each point $P = (p_1, p_2, \dots) = (p_1 = c_1, p_2 = c_2, \dots) = r_P$ we have a tangential (or derivative) space $T(P) \approx \mathbb{R}^n$ which is a vector space of dimension n, in which gradients/forces are defined and a dual space $dT(P)$ on which movements are defined. A positive semi definite matrix M transfers vectors from $dT(P)$ to $T(P)$, whereas M^{-1} (generalized inverse if some eigenvalues are 0) transfers from $T(P)$ to $dT(P)$, and defines a metric (and scalar product) on this space. Normally M is the metric induced by the cartesian coordinates

To be more explicit, if r are the cartesian coordinates, and $c_i(r)$ are general coordinates then one can define M as

$$M_{ij} = \frac{\partial c_i}{\partial r} \cdot \frac{\partial c_j}{\partial r}. \tag{1}$$

If M is not diagonal it means that some coordinates are dependent from each other, M can be seen as an overlap, and the various directions are not orthogonal.

Thus a change along c_i might change also c_j, to change only c_i one has to move along the dual direction with respect to c_i (thus we called $dT(P)$ movement space).

The gradient of a function $E(r)$ t in the c_i space is

$$\frac{\partial E(c_i)}{\partial c_i} = \frac{\partial E(r)}{\partial r} \cdot \frac{\partial r(c_i)}{\partial c_i} = \sum_k \frac{\partial E(r)}{\partial r} \cdot \frac{\partial c_k(r)}{\partial r} M^{-1}_{ki}, \tag{2}$$

where M^{-1} is the generalized inverse if M is singular. The gradient can be seen as $\partial E(r)/\partial r \cdot \partial c_k(r)/\partial r$ that is in the gradient space $T(P)$, and the final result that we get going to the dual movement space.

To go from the normal space to the tangential space in the previous example one should use P as reference point: the vector v in the gradient space and v^* in the movement space corresponds to the point U with $U_r = P_r + \sum_k v^*_k \partial c_k/\partial r$, that has (to the first order) coordinates $p_i + v_i$. Vice-versa a point U is mapped to the vector v of the gradient space with coordinates $v_i = u_i - p_i$.

Note that in our notation (and unlike what is often assumed) going back and fort from the real space to the tangential space is not necessarily the identity, it has to be close to the identity (to the first order) only close to P. On the other hand we assume that any point can be transferred to the tangential space without error (even if it will end up in another place when coming back). In other words we don't assume that uniform speed geodesic are used to map to/from the tangential space.

In the previous example the dimension n of the tangential space is equal to the number of coordinates, but n might also be smaller, for example a direction could be represented with a normalized 3D vector, then the derivatives have only two independent directions that are tangential to the sphere, similarly one can use quaternions to represent rotations, and they have only 3 independent derivative directions.

Normalized vectors and rotations are a good example of the problems that can arise using generalized coordinates: they can be parametrized also using angles, but angles can become singular (the latitude is ill defined at the poles). Thus if possible one should prefer general coordinates like normalized vectors and quaternion that have an uniform derivative space everywhere (as the underlying space is) unlike angles that introduce singularities. The derivatives of normalized vectors are regular, but show that the mapping from the tangential space to the real space might be incomplete, and the transfer from the tangential space at one point to the tangential space in another point is not necessarily the unity matrix. Indeed it is impossible to define a continuous gradient field that is non zero everywhere on the sphere, so that somewhere there must be a discontinuity in the mapping between tangential spaces.

Finally a last thing that a robust algorithm should be able to cope with is that some directions might be in the null space of M (have 0 eigenvalue), or might be blocked by constraints.

5 PES Description

On the potential surface we can get the following information:

1. energy, normally it is the cheapest to calculate and storing it occupies just one real. The free energy is an exception, because it cannot be directly evaluated. It is still possible to calculate the average potential energy for example, but then has to take care than even if related, the energy refers to a different surface than the gradients.

2. gradient/forces, often are relatively cheap to calculate (less than n times the energy), in some cases it is worth to always evaluate them, but not always. Their storage cost is $ndim$ reals, the number of dimensions.

3. hessian, second derivatives are analytical only in few programs, often they are calculated using finite differences of the gradients. Their storage cost is $ndim^2$ reals.

4. higher derivatives, are even more expensive to calculate, and store, and normally calculated from energy and gradients.

Most algorithms to explore the potential energy surface use the first and second derivative. The use of second derivatives has some drawbacks: the storage of the full hessian is quadratic in the number of dimensions, and very often they are actually calculated using gradients. Still having the hessian often allows for much better algorithms, for this reason many program build an approximate hessian incrementally. There are several update strategies, and for when looking for a minimum or a saddle-point such an approach seem to be close to the best thing that can be done, but from a global PES view it is unclear to which part of the PES to associate the various approximate hessians.

The most efficient methods to go uphill to find saddle points without knowing the other basin of attraction actually need information about the third derivative. A smart way to evaluate this information already in a way that makes the directions to follow easy to find is the scaled hypersphere search [7], but also in this case what is really calculated by the underlying program are energies and gradients (on a surface that depends on the hessian).

To build an incremental description of the PES that is as light weight as possible and can be used by all kinds of algorithms (and can store the evaluations of all kind of explorations) restricting to store the position of the evaluation, energy and possibly gradient seems to be a good compromise between generality and efficiency. The approximated hessian can be left as something built by the exploration algorithm and not directly associated with the PES, in fact it is relatively easy to build an hessian from a series of evaluations (either by least squares, or by a series of updates), thus also algorithms that use approximate hessians can take advantage of the information stored.

Storing exactly what is calculated, and not derived quantities helps keeping the footprint low (all these quantities have a storage size for each evaluation that grows as $O(ndim)$), and usable by different high-level algorithms, but our

goal is not to just build a big database of what has been calculated. To be really useful we have to endow this data with some topological information. For example it is clear that an approximated hessian can be built only from the evaluations that are close to the point of interest. Thus building a neighboring list is a first step in that direction.

To decide which points a neighbors one has to decide some distance cutoff. Also using just the hessian is not enough to do this, because we want to consider neighbors points for which the harmonic approximation is still mostly valid, so what we would need is actually the third derivative. The maximum distance that can be used is related to the minimum size of the basin of attraction, and so to the trust radius of optimization algorithms. One can think at adapting this size to the direction, but this is difficult to control, so for simplicity and robustness at the moment we use a uniform size rDualMax in the movement space $dT(P)$. Thus all direction dependent adaptation has to be done a priori when choosing the coordinates (scaling a coordinate inversely scales the size in $dT(P)$). To cope well with linear dependent or ill-defined coordinates points whose cartesian distance is less than rCartesianMax are also considered neighbors.

A neighboring list is not enough, another very important question that we want to be able to answer is around which points we have explored enough. This allows one to consider some region of the PES as known.

6 Local frame of reference

Looking at the exploration algorithms that we want to be able to perform one can see that not only the the explored points are of interest, but also the the explored directions for the point that are at the border, so that one can extend the exploration in the unexplored directions.

One has to think that even the nearest neighbors are 3^{ndim} and thus grow exponentially with the number of dimensions. Indeed the larger the number of dimensions the larger the "border" becomes. Sampling the border is more and more costly. Here the gradient can help very much, as it gives the direction in which the energy decreases and directions in which it should be constant. We want a method where the storage cost for the single point grows at most linearly, so we take only the $2ndim$ first neighbors, but oriented in the direction of the gradient. To use the gradient g in the movement space as frame of reference one can build an orthogonal basis around it, or in a much simple way rotate the $\epsilon_1 = \delta_i^1$, $\epsilon_2 = \delta_i^2$,... basis aligning it with the direction of the minimum minDir $= -g/||g||$. If M is singular in some cases might be advantageous in some cases to add a vector from its null space to g to improve the length of minDir in the gradient space (adding a such a vector does change the normalization factor of the vector), but as this is not so well defined it is not done. Thus linear dependent vectors are scaled proportionally to their length, and thus they are equivalent to a single variable with a mean length that favors the longest lengths. This has to be kept into account when using redundant coordinates, so if cartesian coordinates are added they should be scaled.

To put a vector in that frame of reference of minDir one can use the rotation matrix $R(\epsilon_1, \text{minDir})$, which is the rotation that maps the vector ϵ_1 to minDir. It is possible to implement this rotation efficiently, as shown in the appendix A.

To find in which direction a neighbor represented by the vector n in movement space one can simply look at the component with the largest absolute value of the vector $v = R(\text{minDir}, \epsilon_1)n$ which is in the frame of reference of the gradient. This assumes an orthogonal space in the movement space, but is much faster and does not need extra memory like orthogonalization based methods.

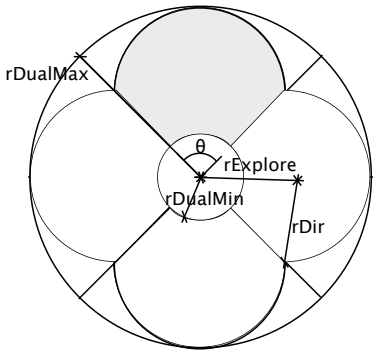

Figure 1. Exploration region: the gray region is the region of the direction that goes up. The region encompass the points that have a minimum distance rDualMin, a maximum distance rDualMax an angle θ, defined by $cos(\theta/2) <$ cosMax, and a maximum distance rDir from the ideal point that is at rExplore from the center. In the cartesian space only the angle and the equivalent of rDir, namely rDirCartesian are checked.

Finding exactly which directions are explored by a point needs some thresholds, both in the movement space and in the cartesian space, we check the geometrical region shown in figure Figure 1 with the following values:

$$\begin{aligned} \text{rDualMin} &= \frac{\text{rExplore}}{2}, & \text{rDualMax} &= 2\,\text{rExplore}, \\ \text{cosMax} &= \frac{1}{\sqrt{ndim}}, & \text{rDir} &= \text{rExplore}, \end{aligned} \tag{3}$$

where rExplore is the "discretization step" of the representation. These values mean that there is no gap, and all values in the correct range are assigned to a direction, which is important for large $ndim$.

To catch degenerate directions space we perform a similar step in the cartesian space with

$$\begin{aligned} \text{rCartesianMin} &= 0, & \text{rCartesianMax} &= \infty, \\ \text{cartesianCosMax} &= \frac{1}{\sqrt{2}}, & \text{rDirCartesian} &= \text{cartesianDiffMin} \end{aligned} \tag{4}$$

which means that all points within cartesianDiffMin $\approx 0.1 Bohr$ of the point in that direction are considered close if their angle with the direction is less than 45 deg. In the movement space one takes only the main direction, whereas in the cartesian space all matching directions are taken. If the distance of the point is smaller than rDualMin or cartesianDiffMin then the points are considered very close. The detailed procedure is explained in the appendix B.

Having a local frame of reference we can check which directions have been explored and which ones still need to be explored around each point, but we can also do more, we can try to assign topological information to each point, if we expect special points to be close to that point:

- if all neighbors have an energy that is larger than the current, and the direction toward the minimum has been taken then it is likely that there is a minimum nearby, if all directions have been taken then it is most likely. It is also possible to check that points in the direction of the minimum have almost antiparallel gradients, which only needs gradients, and not energies.

- if all neighbors have an energy that is smaller than the current, and the direction toward the maximum is taken the it is likely that there is a maximum nearby, if all directions have been taken then it is most likely.

- if a point is close to a minimum then it is in its own attractor, otherwise it is in the same attractor as the point in the direction of the minimum. Doing this one can build approximate basins of attraction of a minimum.

- if all points in the direction of the maximum have a smaller energy than the current point then there is a saddle-point close by (if some neighbors have energies larger than the current, or a maximum otherwise). This is confirmed if it is in a different attractor than the other point. Checking that the points in the direction of the maximum have almost antiparallel gradients is test that needs only the gradients.

The checks sketched here are correct if the discretization size is small enough, and the "gradient only" version can be used if energy and gradients are not consistent, and the gradients are correct, as with free energy.

This topological information gives enough structure to perform many kinds of algorithms directly, the information is not final, in the sense that refinement might change the stored information (for example refining a minimum will probably change the point that is considered close to the minimum), and there might be some false positives, that go away when refined but as long as the discretization size is small enough with respect to the largest third derivative change and to the inverse of the condition number of the hessian the results should be reliable.

With this structure the MinE strategy becomes simply

1. find the point with smallest energy that has still free directions.

2. explore in one of the free directions

3. iterate from 1

The GoUp strategy is similar:

1. find the point with smallest energy in the current basin of attraction that has still free directions.

2. if the direction toward the minimum has not been taken yet then take the direction toward the maximum (if possible) and declare the point as visited

3. otherwise explore in one of the free directions

4. iterate from 1

During the exploration one should continuously check of which points the newly explored points are neighbors, check for special points, and possibly refine the evaluations close to points that are likely to be special.

If the calculation of the gradient/forces is expensive then one can delay it until one needs to explore starting from that point, i.e. until the point is the lowest point in energy not yet fully explored. This can be very effective because the larger *ndim* the more points are on the boundary.

The generation of a point along a direction should be done as follow:

1. find the ideal direction in the movement space:

$$\text{dirDual1} = R(\epsilon_1, \text{minDir})(\pm\epsilon_i)$$

2. project out linear dependent directions:

$$\text{dir1} = M\text{dirDual1}$$

$$\text{dirDual2} = M^{-1}\text{dir1}$$

3. $\text{cartesianNorm} = \sqrt{\text{dir1} \cdot \text{dirDual1}}$
   if $(\text{cartesianNorm} = 0)$ **then** discard the direction

4. scale:
$$\text{dirDual2}* = \max\left(\frac{\text{rExplore}}{||\text{dirDual2}||}, \frac{\text{cartesianChangeMin}}{\text{cartesianNorm}}\right)$$

5. declare the direction as visited

6. apply constraints, and check if the resulting point is a neighbor of this one and in which direction

7. if the point is much too close to the starting point (using a small threshold), or in a direction that was already visited discard it

8. evaluate point

7 MD comparison

It is instructive to compare the proposed exploration strategies with molecular dynamics (MD).

MD has a maximum timestep, that is probably a little bit smaller than the maximum discretization step, because if the quadratic approximation has a large error most likely also the integrator fails. Still MD has an advantage: the evolution is continuos, so one can use extrapolation methods that can speed up the calculation quite substantially. So one can expect that the volume of the phase space that can be explored by the two methods is not very different.

If done in the microcanonical ensemble MD samples points with a probability proportional to $e^{-\beta E}$, which means that lower energies have a relatively high probability, and higher energies are exponentially less likely. Up to a point this is also correct, as these regions are more important for thermodynamical averages, but after a while the PES in that region is well known, and it would be better to simply use the correct weight and explore other regions. This is even more true if one is interested in rare events or chemical reactions.

Avoiding visiting again regions that were already visited is what the proposed methods (that explore only the configuration space) do, as they never explore the same space in the configurational space twice. Thus the "cost" of the method depends on the ratio between the volume of the configurational space that has to be explored, and the discretization size. This is how these methods in the correct settings can reduce the exponential increase of the cost with increasing dimension if in the chosen coordinates the volume to be explored does not increase exponentially with respect to the discretization step that has to be used.

8 A-posteriori checks

The presented methods were developed to be robust and work reasonably well also with ill defined directions/coordinates, but work optimally in coordinates where the hessian has a small condition numbers, and the third derivatives are not so large (basins of attraction not too small or irregular).

Knowledge about the hessian should be used to define good general coordinates, which can improve the efficiency of the algorithm. An interesting property of this PES representation is that one can easily build indicators that can tell how good the coordinates and discretization step are. Badly ill defined directions are easy to detect as will trigger the cartesian detection of visited directions, a more complete picture can then be collected if requested by looking at the eigenvalues of the M matrix.

When two neighboring points have evaluated their energy, and at least one has the gradient, the hessian in one direction can be evaluated through the deviation from the linear extrapolation. If both have the gradient then the hessian can be evaluated starting form either one and compared giving the deviation of the hessian. Also as difference of the gradients can be used (which

give even more information). This can then be used to detect too large changes on the discretization scale. It is also possible to compare several neighbors at once.

One can try to think adaptive algorithms using these indicators and actually the adapt various thresholds to the place. For the moment we refrained from this for two reasons: it is difficult to build them in such a way that they are robust in all cases, and often they are difficult/impossible to express in a way that does depend only on on the points evaluated so far, but not from their actual evaluation sequence. This makes most of them useful to explore, but less useful for a PES based view, still an automatic reduction of the exploration size would probably be useful.

For now just a detection and evaluation of the choosen parameters is done, but the parameters themselves are not adapted. This is not so bad, because our PES based approach has a very nice property: it is possible to restart the evaluations with a different set of coordinates as long as the coordinates that are frozen do not change. Thus one can reload all the evaluations previously done and reinterpret them using the new coordinates (this is also how one can load trajectories performed with other methods).

9 Examples

artificial potentials [8] [9]
    almost linear dependence
    chemical reactions

10 dchem

- robustness of general coords and constraints - adding new evaluators - framework parallelization - smp - distribued objects - mpi - program parallelization - PES representation:distribute points - explorers & observers - active network - external query explorer

11 Conclusion

We presented a flexible representation of the potential energy surface that offers topological informations on the points that have been evaluated so far. The representation is quite method agnostic, and can use $O(ndim)$ storage for each evaluation[1] The information is stored incrementally and builds a non uniform grid of the evaluations. Two exploration strategies that use this information are proposed, they explore the PES exhaustively guaranteeing that there is no "hole" in the coverage larger than twice the the discretization size, something

[1] the attentive reader might argue that we do use the matrix M which has the same dimension of the hessian. Indeed this is true, but M depends on the coordinates and can be calculated on the fly when requested, and is expected to be sparse for large $ndim$.

that can be very valuable in giving some guarantees about the studied surface, and can be useful also for branch and bound global optimization techniques. Topological information on the potential energy surface is collected and can locate critical points on the surface and help building a conceptual model of the potential surface, and can have potentially many applications.

The method has been implemented in a parallel program able to use several drivers to evaluate energy and forces.

A Efficient N-dimensional v_1, v_2 rotations

As noted by Parlett[10] all 3D rotations can be represented by a pair of unit vectors: $R(v_1, v_2)$ is the rotation that maps v_1 in v_2, and that has minimal norm. In n-dimensional spaces this are not any more all rotations, but represent an important subclass. These are the rotations that are different from unity only in the v_1, v_2 plane.

We are not aware of people using this representation explicitly for general vectors. The existence of such a representation is important for the efficiency of our algorithm, and as the procedure to calculate them avoiding numerical instabilities that can arise when v_1 and v_2 are almost collinear, is not so obvious, and is rather beautiful, we give here the pseudocode to apply such a rotation to a vector res:

```
R(v1,v2,res){
  c=dot(v1,v2);
  v1m=dot(v1,res);
  v2Ortho=v2.copy();
  v2Ortho-=c*v1;
  v2OrthoM=dot(v2Ortho,m);
  res+=(v2-v1)*v1m;
  if (c<0){
      vOrtC=-1/(-c+1);
      res+=(vOrtC*v2Ortho+v1)*v2OrthoM;
  } else {
      vOrtC=-1/(c+1);
      res+=(vOrtC*v2Ortho-v1)*v2OrthoM;
  }
}
```

Almost the same code can be used to apply the rotation to a matrix, and the code can be further optimized if v_1 or v_2 are a basis vector δ_i^k, see dchem for the code.

B Neighbor check

In practice to decide which directions of a point P a point U explores keeping into account possible degeneration of the directions the following procedure can

be used

1. transfer U to the Tangential space of P: $u = T(P)(U)$
2. go to the movement space: $du = M^{-1}u$
3. check the distance in the movement space
4. **if** $||du|| \leq$ rDualMin **then**
   (a) declare that U is very close to P
5. **else if** rDualMin $< ||du|| <$ rDualMax **then**
   (a) find the dimension and direction in which the neighbor is:

$$rdu = R(\text{minDir}, \epsilon_1)du$$

$$\text{idim} = argmin(rdu), idir = dir(\text{idim}, sign(rdu[\text{idim}]))$$

   (b) check the angle of the vector: $|rdu[\text{idim}]|/||du|| \leq$ cosMax
   (c) check if it is within rDir of the direction

$$(rdu[\text{idim}] - \text{rExplore})^2 + ||du||^2 - rdu[\text{idim}]^2 \leq \text{rDir}^2$$

   (d) if passes checks declare direction as explored and point U as lying in that direction
6. check the distance in the cartesian space: $nCart = \sqrt{du \cdot u}$
7. cartesian length in the various directions:

$$lens = \text{diag}(R(\text{minDir}, \epsilon_1)MR(\epsilon_1, \text{minDir}))$$

8. **foreach** idim **do**
   (a) find the dimension and direction in which the neighbor is:

$$ru = R(\epsilon_1, \text{minDir})u$$

   (b) check the angle of the vector: $du[\text{idim}]/lens[\text{idim}] \leq$ cartesianCosMax
   (c) check if it is within rDirCartesian of the direction:

$$(ru[\text{idim}] - lens[\text{idim}]\,\text{rExplore})^2 + nCart^2 - ru[\text{idim}]^2 \leq \text{rDirCartesian}^2$$

   (d) if passes checks and $lens[\text{idim}] > 0$ declare direction as explored and point U a lying in that direction
9. **if** (nCartesian $\leq$ rCartesianMin) declare that U is very close to P
10. **if** it is not in any direction
    (a) if $||du|| <$ rDualMax declare that it is in an unprecised direction
    (b) if P is close to U (possible if U threshold are more lax) declare that U is a neighbor of P only because P is neighbor of U.

References and Notes

[1] J. Simons, P. Jorgensen, H. Taylor and J. Ozment, *J Phys Chem-Us*, 1983, **87**, 2745–2753.

[2] H. Schlegel, *Journal of Computational Chemistry*, 2003, **24**, 1514–1527.

[3] G. Bringmann, S. Güssregen and H. Busse, *Journal of Computer-Aided Molecular Design*, 1992, **6**, 505–512.

[4] T. Huber, A. Torda and W. vanGusteren, *J Comput Aided Mol Des*, 1994, **8**, 695–708.

[5] A. Laio and F. L. Gervasio, *Rep Prog Phys*, 2008, **71**, 126601.

[6] G. Ciccotti, R. Kapral and E. Vanden-Eijnden, *Chemphyschem*, 2005, **6**, 1809–1814.

[7] S. Maeda and K. Ohno, *Chem Phys Lett*, 2008, **460**, 55–58.

[8] K. Müller and L. Brown, *Theor Chim Acta*, 1979, **53**, 75–93.

[9] C. Cerian and W. Miller, *J Chem Phys*, 1981, **75**, 2800–2806.

[10] B. N. Parlett, *Classics in Applied Mathematics*, 1998, **20**, year.