2. Geometry Optimizations

2.1. The Potential Energy Surface

The potential energy surface (PES) is a central concept in computational chemistry. It specifies how the energy of a molecular system varies with small changes in its structure. A potential energy surface is a mathematical relationship linking molecular structure and the resultant energy.

For a diatomic molecule, it is a two-dimensional plot with the internuclear separation on the X-axis (the only way that the structure of such a molecule can vary), and the potential energy at that bond distance on the Y-axis, producing a curve.

For larger systems, the surface has as many dimensions as there are degrees of freedom within the molecule.

Generally, a non-linear $N$ atomic molecule, has $3N - 6$ degrees of freedom, or internal coordinates. This is because all $N$ atoms can move in 3 dimensions ($x$, $y$ and $z$) giving $3N$ degrees of freedom. However, 6 of those: three translations - in $x$, $y$, $z$ directions, and three rotations - along $x$, $y$ and $z$ axes of the molecule as a whole - do not produce any change in energy. There are also referred to as external coordinates.

For a linear molecule there are only two rotations and the number of internal coordinates (degrees of freedom) is $3N - 5$.

A potential energy surface (PES) is often represented by illustrations like the one below. This sort of drawing considers only two of the degrees of freedom within the molecule, and plots the energy above the plane defined by them, creating a literal surface. Each point corresponds to the specific values of the two structural variables-and thus represents a particular molecular structure-with the height of the surface at that point corresponding to the energy of that structure.
2.2 Stationary points of PES

There are two minima on this potential surface. A minimum is the bottom of a valley on the potential surface. From such a point, motion in any direction—a physical metaphor corresponding to changing the structure slightly—leads to a higher energy.

A minimum can be a local minimum, meaning that it is the lowest point in some limited region of the potential surface, or it can be the global minimum, the lowest energy point anywhere on the potential surface. Minima occur at equilibrium structures for the system, with different minima corresponding to different conformations or structural isomers in the case of single molecules, or reactant and product molecules in the case of multi-component systems.

Peaks and ridges correspond to maxima on the potential energy surface. A peak is a maximum in all directions (i.e., both along and across the ridge). A low point along a ridge—a mountain pass in our topographical metaphor—is a local minimum in one direction (along the ridge), and a maximum in the other. A point which is a maximum in one direction and a minimum in the other
(or in all others in the case of a larger dimensional potential surface) is called a saddle point (based on its shape).

For example, the saddle point in the diagram is a minimum along its ridge and a maximum along the path connecting minima on either side of the ridge. A saddle point corresponds to a transition structure connecting the two equilibrium structures.

The slice along the “lowest-energy” path connecting the global minimum, the local minimum and the transition state, that is, along the “reaction coordinate” gives a one-dimensional energy surface. Here the transition state (saddle point) looks like a maximum. Note that the horizontal axis (i.e. the reaction coordinate) is a combination of the two coordinates (bond length and angle) from the original plot.

Together, minima, maxima and saddle points are called stationary points. They have in common that at any stationary point (minimum, maximum) the surface is flat, i.e. parallel to the horizontal line corresponding to the one geometric parameter (or to the plane corresponding to two geometric parameters etc). A marble placed on a stationary point will remain balanced, i.e. stationary, while at any other point on a potential surface the marble will roll toward a region of lower potential energy.

Mathematically, the stationary points are characterized by zero first derivatives with respect to the geometrical parameters:

$$\frac{\partial V}{\partial q_1} = \frac{\partial V}{\partial q_2} = ... = \frac{\partial V}{\partial q_n} = 0$$

(2.1)

where the potential energy is denoted by $V$ and:

$n = 3N - 6$ (3N – 5) if we work in internal coordinates
$n = 3N$ for Cartesian coordinates
$n > 3N - 6$ for redundant internal coordinates, i.e. internal coordinates, such as bond lengths, angles and torsions, but more of them than we actually need.

The derivatives can be arranged into a vector, which is called gradient of energy

$$\text{grad}V = \nabla V = \begin{pmatrix} \frac{\partial V}{\partial q_1} \\ \frac{\partial V}{\partial q_2} \\ ... \\ \frac{\partial V}{\partial q_n} \end{pmatrix}$$

(2.2)

as we know from basic physics, gradient of energy is closely related to force:

$$\mathbf{F} = -\text{grad}V = -\nabla V$$

(2.3)
Therefore at a stationary point, the forces on the atoms in the molecule in all directions are all zero, which is another reason why they are stationary points.

### 2.3. Locating minima

Minima on potential energy structures are important because they correspond to the optimum molecular geometries, also called equilibrium structures. Geometry optimization is therefore more or less synonymous with *energy minimization*. From a mathematical point of view, it is a problem of finding a minimum of a function of many variables.

The basic idea of a minimization algorithm is to vary the geometry of the molecule (i.e. atomic positions) in some systematic fashion so that the energy keeps decreasing; once the energy stops decreasing and only increases with further geometry variations, the minimum is found.

Important point to remember is that *all minimization algorithms only find local minima: you are never guaranteed that the minimum found is a global one!*  

There are several classes of algorithms that have been developed for this purpose. They can be divided into three basic classes:

- only function values are used (no derivatives)
- first derivatives (gradients) are used
- second derivatives (Hessians) are used.

The first class represents the most *robust*, but least *efficient* way to find the minimum. The example is the Simplex algorithm (also known as “amoeba”). It is robust, because it can always find a minimum of pretty much any function, no matter how crazy. Least *efficient* means that for well behaved functions, the other two classes could do it ways faster! Because molecular PES are usually smooth and well behaved functions of atomic coordinates, these algorithms are usually not the first choice for optimizations.

If the gradients can be computed (note that the function has to be differentiable - i.e. continuous), they indicate in which direction the energy decreases. Remember the gradient is the negative of force, and the force pushes the atoms to their minimum energy position (= equilibrium) position. The simplest gradient based algorithm is the so-called “steepest descent”, where steps are made in the direction of the gradient until the minimum is found (in that direction), then the gradient is recomputed, steps are made in the new direction etc. This is simple but in practice very inefficient.

With gradients available the direction toward the potential minimum is known, but not how far one has to step to get there. Imagine the surface could be approximated by a quadratic function (parabola). Assuming one dimension for illustration we could write a Taylor expansion around some point $q_0$:  

\[
\begin{align*}
E(q) & = E(q_0) + \nabla E(q_0) \cdot (q - q_0) + \frac{1}{2} \nabla^2 E(q_0) \cdot (q - q_0)^2 \\
& = E(q_0) + \langle \nabla E(q_0), q - q_0 \rangle + \frac{1}{2} \langle \nabla^2 E(q_0), (q - q_0)^2 \rangle.
\end{align*}
\]
\[
V(q) \approx V(q_0) + \left( \frac{dV}{dq} \right)_{q_0} (q - q_0) + \frac{1}{2} \left( \frac{d^2V}{dq^2} \right)_{q_0} (q - q_0)^2
\]  

(2.4)

taking a derivative with respect to \( x \):

\[
\left( \frac{dV}{dq} \right)_q \approx \left( \frac{dV}{dq} \right)_{q_0} + \left( \frac{d^2V}{dq^2} \right)_{q_0} (q - q_0)
\]

(2.5)

Assuming the point \( q_0 \) is our starting point and we are looking for the next point “q” on our PES where the energy is minimum, we know that at “q” (the minimum) the gradient has to be zero. Setting the gradient at point “q”, which is the left-hand side of the last equation, equal to zero:

\[
0 = \left( \frac{dV}{dq} \right)_{q_0} + \left( \frac{d^2V}{dq^2} \right)_{q_0} (q - q_0)
\]

(2.6)

and solving for q:

\[
q = q_0 - \left( \frac{dV}{dq} \right)_{q_0} \left( \frac{d^2V}{dq^2} \right)_{q_0}^{-1} \left( \frac{dV}{dq} \right)_{q_0} = q_0 - \left[ \left( \frac{d^2V}{dq^2} \right)_{q_0} \right]^{-1} \left( \frac{dV}{dq} \right)_{q_0}
\]

(2.7)

In more dimensions (remember we have \( n \) coordinates, which can be \( 3N - 6 \) or more for \( N \) atoms depending on the coordinates), \( q \) has \( n \) components, gradient is a vector with \( n \) components (eqn. 2.2), which we denote \( g \), and the second derivatives are so called force constants, which form a \( n \)-by-\( n \) matrix called Hessian, which we will denote \( H \):

\[
H_{ij} = \frac{\partial^2 V}{\partial q_i \partial q_j} \quad \mathbf{H} = \begin{pmatrix}
\frac{\partial^2 V}{\partial q_1 \partial q_1} & \frac{\partial^2 V}{\partial q_1 \partial q_2} & \cdots & \frac{\partial^2 V}{\partial q_1 \partial q_n} \\
\frac{\partial^2 V}{\partial q_2 \partial q_1} & \frac{\partial^2 V}{\partial q_2 \partial q_2} & \cdots & \frac{\partial^2 V}{\partial q_2 \partial q_n} \\
\vdots & \vdots & \ddots & \vdots \\
\frac{\partial^2 V}{\partial q_n \partial q_1} & \frac{\partial^2 V}{\partial q_n \partial q_2} & \cdots & \frac{\partial^2 V}{\partial q_n \partial q_n}
\end{pmatrix}
\]

(2.8)

The equation then becomes:

\[
q = q_0 - \mathbf{H}^{(0)}^{-1} \mathbf{g}^{(0)}
\]

(2.9)

where the superscript “0” symbolizes the starting point and the “-1” is matrix inverse.
For the quadratic (parabolic) surface, this formula will take us from any starting point straight to the minimum. If the surface is not exactly parabolic, this formula can be iterated, i.e. compute gradient and (inverted) Hessian at point \( q_0 \), move to the new point \( q \) using formula (2.9), recalculate gradient and Hessian etc.

This algorithm is known as Newton-Raphson method.

The closer to the minimum one gets the better the quadratic approximation becomes: eventually this algorithm will converge to the minimum.

The disadvantages of Newton-Raphson method are the following:

- Hessian matrix is large and computationally expensive to construct at every iteration step
- Hessian must be inverted (inverting a matrix is also computationally expensive)
- Second derivatives are even more sensitive to discontinuities than gradients: for surfaces that are not “well behaved” the method may become unstable and diverge or crash.

Do avoid that, a class of methods known as Quazi-Newton (also variable metric methods) instead of using the true Hessian estimate it (actually its inverse) approximately, e.g. from the gradients at the successive steps (there are many schemes for estimating and updating Hessians, see e.g. Numerical Recipes: The Art of Scientific Computing by William H. Press, Brian P. Flannery, Saul A. Teukolsky and William T. Vetterling, www.nr.com). The exact Hessian can be calculated once in a while, if needed, typically when the optimization gets close to the minimum.

While this may sound like a step back - using an approximation instead of the true \( H \), quazi-Newton methods are actually more reliable and efficient than the old Newton-Raphson. You can imagine more sophisticated algorithms for finding minima, using e.g. higher order (cubic) approximations to the surface etc. Some details of the algorithm used by Gaussian can be found on the Gaussian manual pages.

2.4. Analytical vs. numerical gradients and Hessians

Gaussian uses both gradients and approximate Hessians or true Hessians when available to find the minima. It is important to know how the computations of gradients and Hessians are implemented for each method.

For example, under MP & Double Hybrid DFT methods in Gaussian manual you will find this:

**AVAILABILITY**

**MP2, B2PLYP[D], mPW2PLYP[D]**: Energies, analytic gradients, and analytic frequencies.

**MP3, MP4(DQ) and MP4(SDQ)**: Energies, analytic gradients, and numerical frequencies.

**MP4(SDTQ) and MP5**: Analytic energies, numerical gradients, and numerical frequencies.

Here “frequencies” as we will see later, refer essentially to Hessians, because vibrational frequencies are calculated directly from them. The important distinction here is analytical vs.
numerical. While there is no such thing as rigorously “analytical” in Gaussian calculations (everything is done numerically),

- analytical gradients (frequencies) mean that essentially exact derivatives are calculated by solving the corresponding derivative Shrodinger equation.

- numerical gradients are calculated by finite differencing from energies at two points:
  \[
  \left( \frac{dV}{dq} \right)_{q_0} \approx \frac{V(q_0 + \Delta q) - V(q_0)}{\Delta q}
  \]  
  (2.10)
  this is not only less accurate, but also more computationally costly (especially the larger the molecules get) because for each coordinate two energies have to be evaluated at two different points (i.e. the Shrodinger equation solved twice).

- numerical frequencies (or more precisely the force constants) are calculated either by numerical differentiation of the analytical gradients, or in the worst case scenario, by numerical differentiation of numerical gradients (which is equivalent to calculating finite difference second derivatives from the energies).
  \[
  \left( \frac{d^2V}{dq^2} \right)_{q_0} \approx \frac{V(q_0 + \Delta q) + V(q_0 - \Delta q) - 2V(q_0)}{\Delta q^2}
  \]  
  (2.10)
  Obviously, this is very slow and very inaccurate.

Fortunately, most methods, even the very high level ones nowadays have implemented at least analytical gradients.

2.5. Geometry Optimization in Gaussian

Geometry optimizations are done by using the keyword Opt in the route section of the Gaussian input file. The molecule specification for a geometry optimization can be given in any format desired: Cartesian coordinates, Z-matrix, mixed coordinates. By default the optimization will be done in redundant internal coordinates, no matter what input format you use. We will discuss different options and tricks that can be done with the redundant internal coordinates below.

The basic optimization input looks like this:

```plaintext
%chk=formaldehyde.chk
# HF/6-31G(d) Opt Test

Formaldehyde Optimization
0 1
C 0.03300000000 -0.00000000000 0.00000000000 0.00000000000
O -1.18700000000 -0.00000000000 0.00000000000 0.00000000000
H 0.57700000000 0.94300000000 0.00000000000 0.00000000000
H 0.57700000000 -0.94300000000 0.00000000000 0.00000000000
```
Remember: if you generate your input using Gabe dit, get rid of keywords that request full printing of the basis set and orbital parameters, ie. Gfinput Pop=Full Density IOP(6/7=3).

We'll now look at the output from the formaldehyde optimization. The actual optimization starts like this:

```
GradGradGradGradGradGradGradGradGradGradGradGradGradGradGradGradGradGrad
Berny optimization.
Initialization pass.
```

```
!   Initial Parameters    !
! (Angstroms and Degrees)  !
--------------------------
! Name  Definition        Value          Derivative Info. 
! ------------------------
! R1    R(1,2)             1.22           estimate D2E/DX2
! R2    R(1,3)             1.0887         estimate D2E/DX2
! R3    R(1,4)             1.0887         estimate D2E/DX2
! A1    A(2,1,3)           119.9799       estimate D2E/DX2
! A2    A(2,1,4)           119.9799       estimate D2E/DX2
! A3    A(3,1,4)           120.0402       estimate D2E/DX2
! D1    D(2,1,4,3)         180.0          estimate D2E/DX2
```

```
Trust Radius=3.00D-01 FncErr=1.00D-07 GrdErr=1.00D-07
Number of steps in this run= 20 maximum allowed number of steps= 100.
GradGradGradGradGradGradGradGradGradGradGradGradGradGradGradGradGradGrad

This is the list of (redundant) internal coordinates: bond lengths - R, angles - A and dihedrals - D that will be optimized. For example, R1 \ = \ R(1,2) \ is the length of the bond between atoms 1 and 2, \ D(2,1,4,3) \ is the dihedral (torsion) angle 2-1-4-3.

Note that these coordinates are assigned automatically. We will discuss below how to modify them.

After each step of the optimization, the following is printed (with less important parts omitted):

```
GradGradGradGradGradGradGradGradGradGradGradGradGradGradGradGradGradGrad
Berny optimization.
... 
Search for a local minimum.
Step number 2 out of a maximum of 20
All quantities printed in internal units (Hartrees-Bohrs-Radians)
...
The meaning of this output should be fairly obvious. Important is the table which tells you how the optimization is progressing. The items on the left are the convergence criteria, the numbers under Threshold are the limits for each criterion: if the actual value is less than the threshold, the criterion is satisfied. If all are satisfied the geometry is considered optimized.

Note that energy is not an optimization criterion. However, after each step is taken, a single point energy calculation follows at the new point on the potential energy surface, producing the normal output for such a calculation.

The final optimized structure appears immediately after the final convergence tests:
When the optimization converges, it knows that the current structure is the final one, and accordingly ends the calculation at that point. Therefore, the energy for the optimized structure is found in the single point energy computation for the previous step in other words, it appears before the successful convergence test in the output.

SCF Done:  E(RHF) =  -113.866330648     A.U. after    9 cycles

It also appears in the archive entry at the end of the output file: