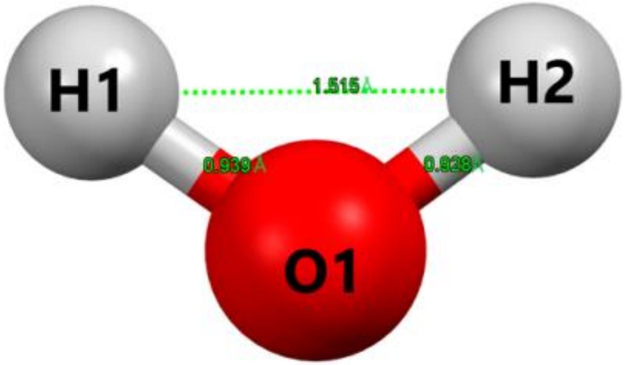
Topic : Geo Opt

Ionic Optimization

Why we need Geometric Optimization?

Structure : Reasonable

molecule



CSD: CIWKEY01

To describe the structure, we need to know the coordination of atoms in space

Cartesian coordinates

1. choose the xyz direction
2. write down the coordination

O	0.000	0.000	0.000
H	0.906	0.641	0.000
H	-0.906	0.641	0.000

3. Display the structure with software

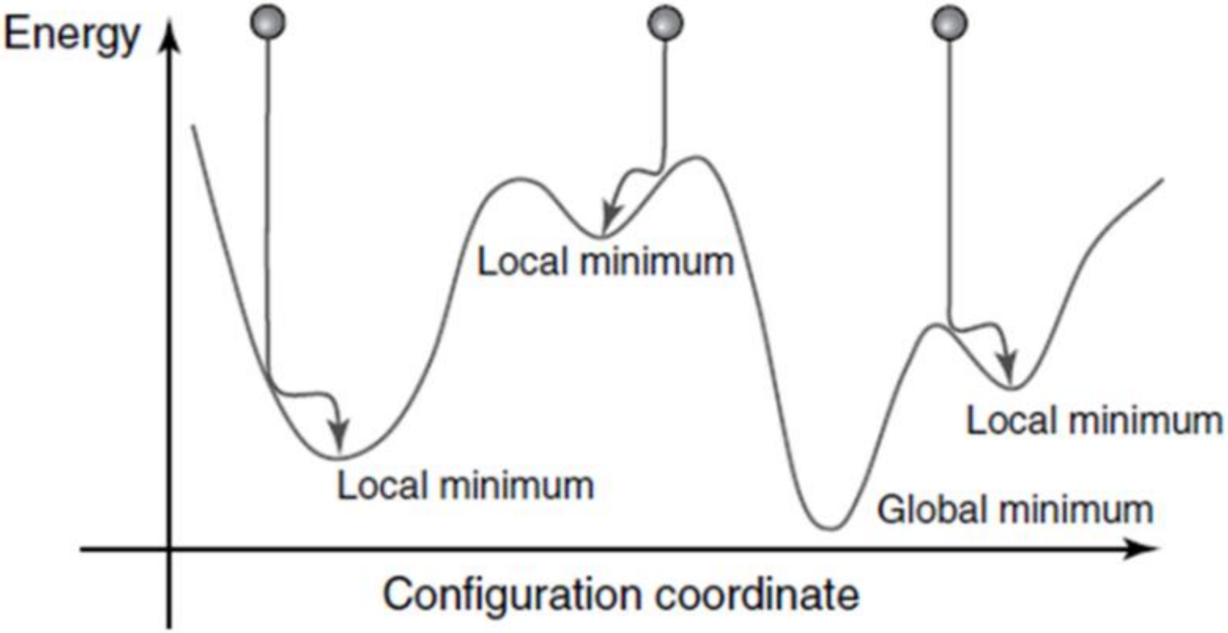
Materials Studio, Gauss View,

Question:

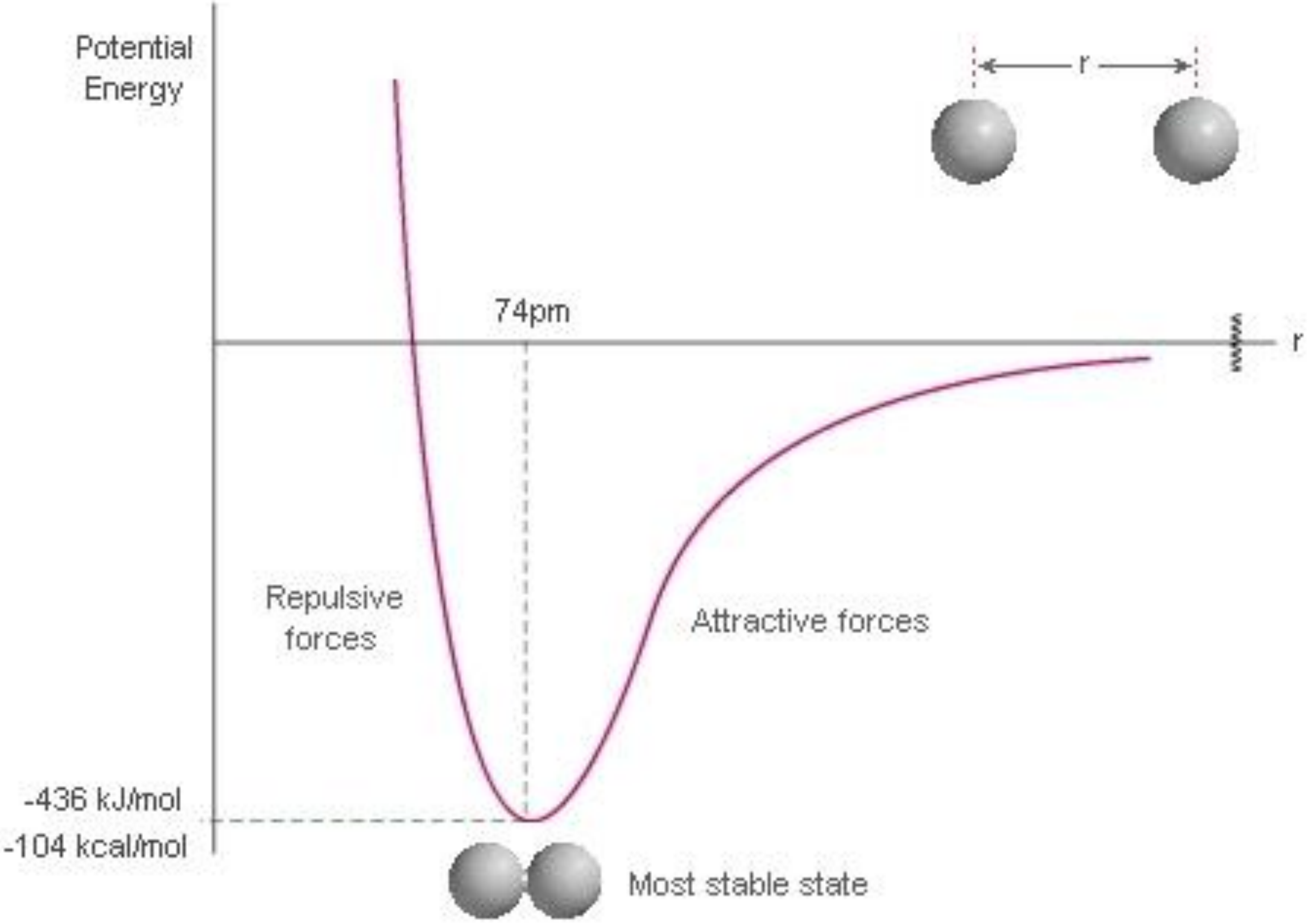
How do we know these parameters?

What is ionic optimization?

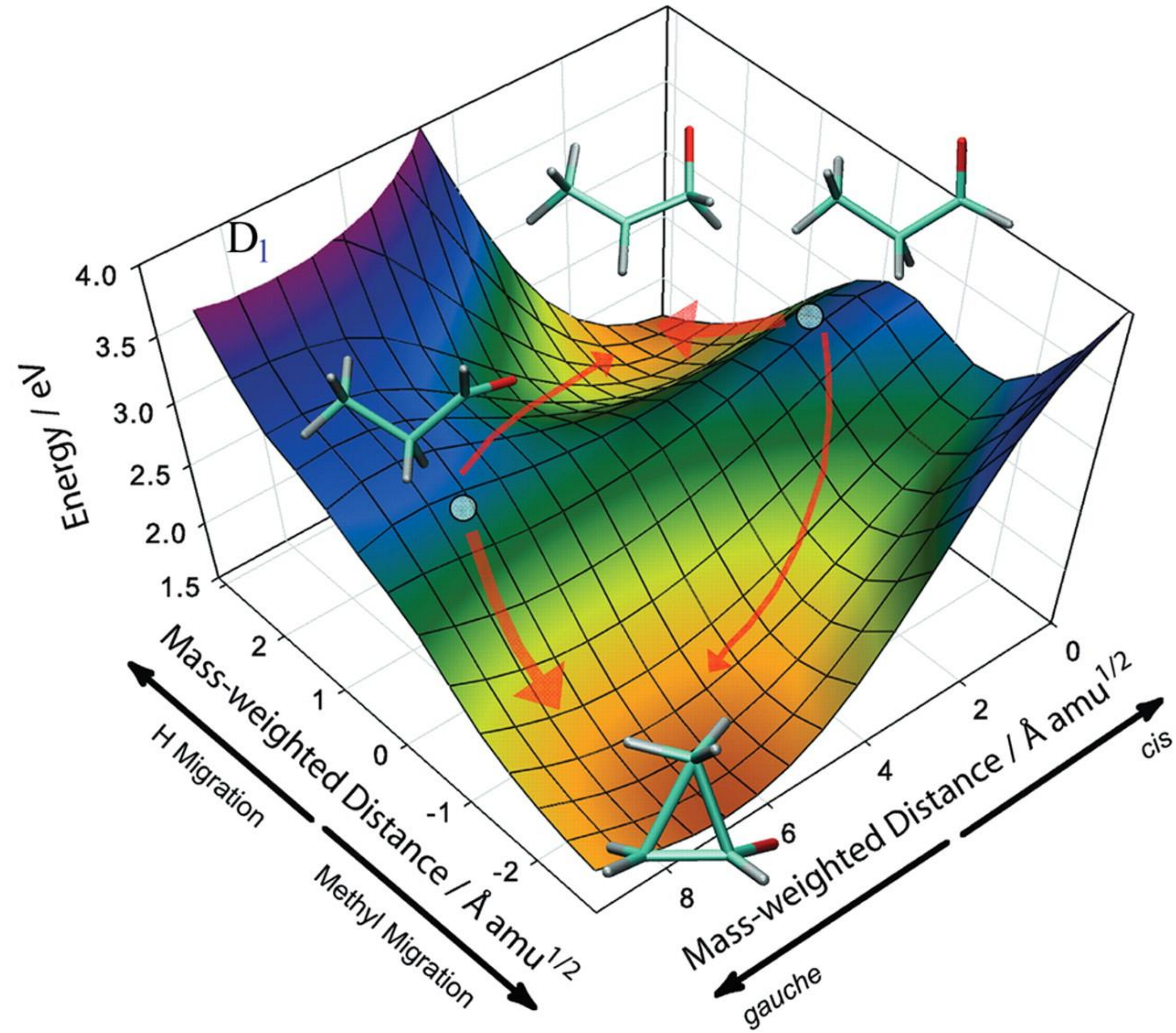
search for the local minimum in the energy potential landscape



A Simple Example of diatom molecule



Complexity in Potential Energy Surface



What is ionic optimization?

search for the local minimum in the energy potential landscape

Two steps:

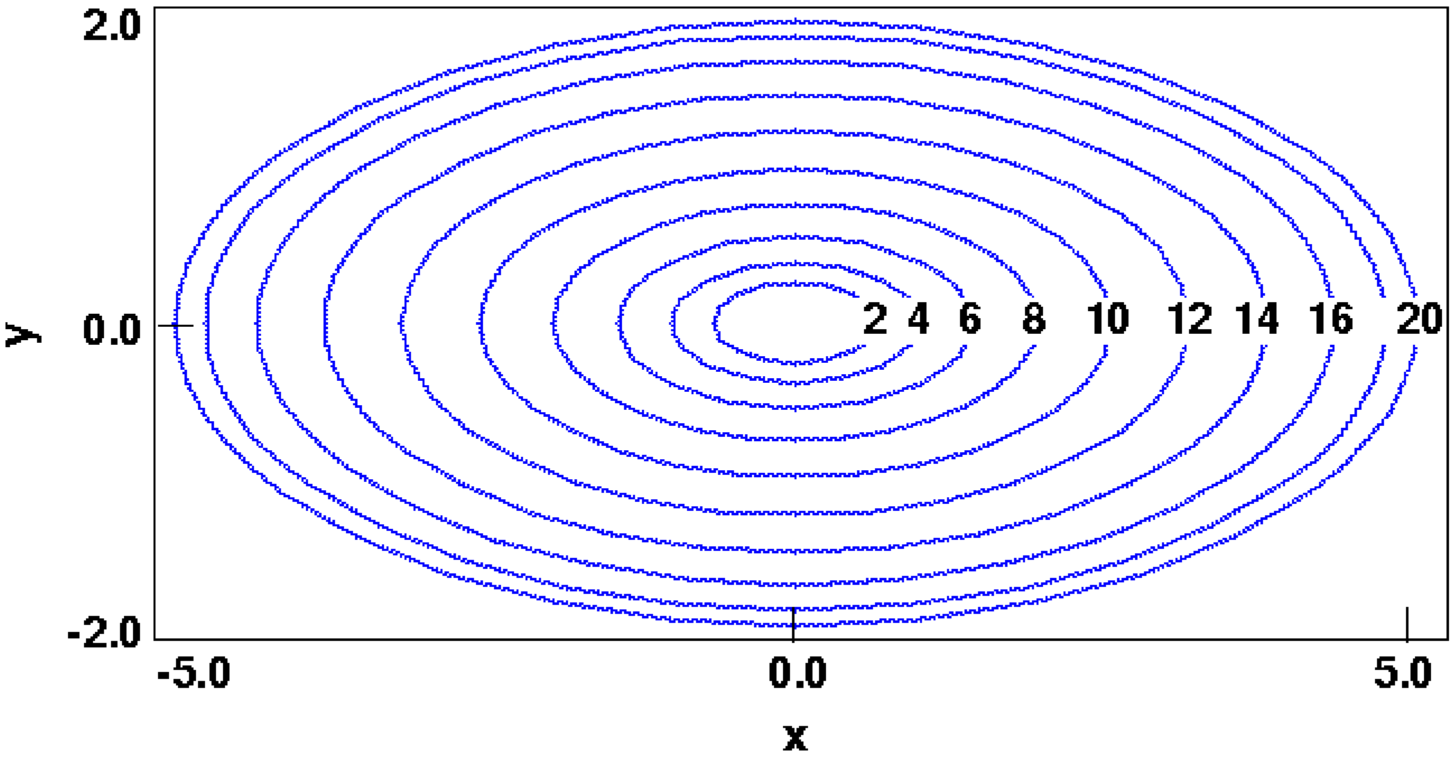
1. **Energy evaluation**
   The energy expression must be defined and evaluated for a given conformation, sometimes including external restraining terms

2. **Conformation adjustment**
   to reduce the value of the energy expression. A minimum may be found, depending on the nature of the algorithm, the form of the energy expression, and the size of the structure.

The efficiency of the optimization is judged by both the time needed to evaluate the energy expression and the number of structural adjustments (iterations) needed to converge to the minimum.

A geometry optimization example

$E(x, y) = x^2 + 5y^2$



a minimizer must determine both the *direction* toward a minimum and the *distance* to the minimum in that direction.

A good initial direction is simply the slope or derivatives of the function at the current point.

The derivatives are a two-dimensional vector: $\nabla E = (2x, 10y)$

ionic optimization algorithms

search for the local minimum in the energy potential landscape

mathematical problem : find the minimum of a function f(x1, x2…)

Simply:

$$f(\vec{x}) = a + \vec{b}\vec{x} + \frac{1}{2}\vec{x}\mathbf{B}\vec{x} = \bar{a} + \frac{1}{2}(\vec{x}-\vec{x}^0)\mathbf{B}(\vec{x}-\vec{x}^0)$$

where **B** is the Hessian matrix

$$\mathbf{B}_{ij} = \frac{\partial^2 f}{\partial x_i \partial x_j}.$$

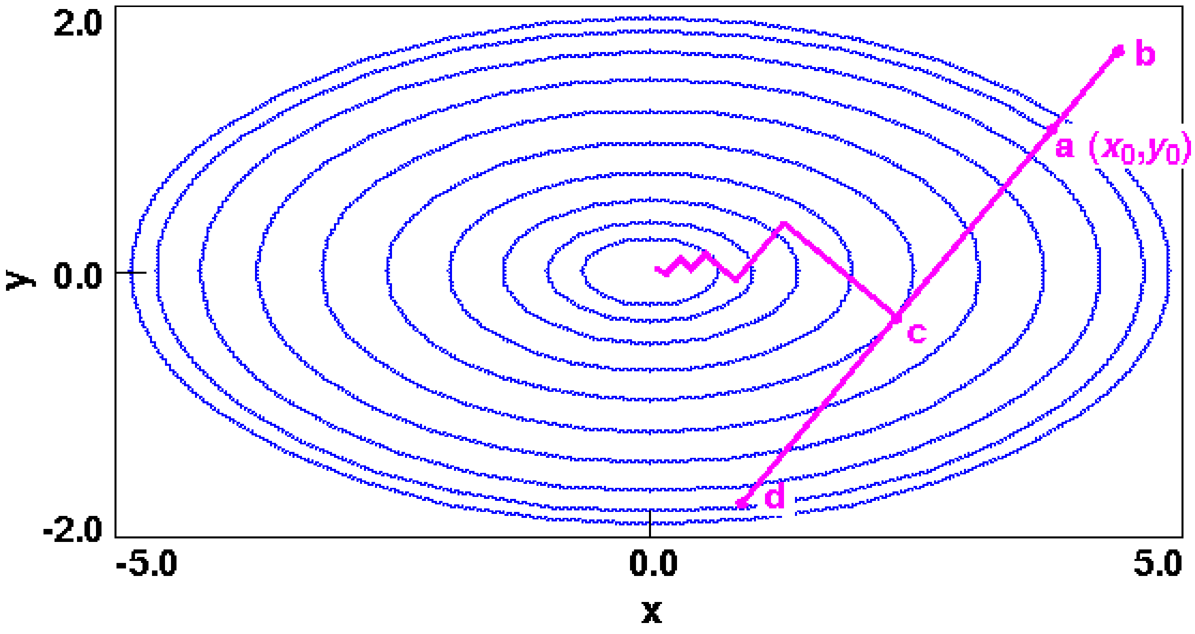
for a stationary point, one requires

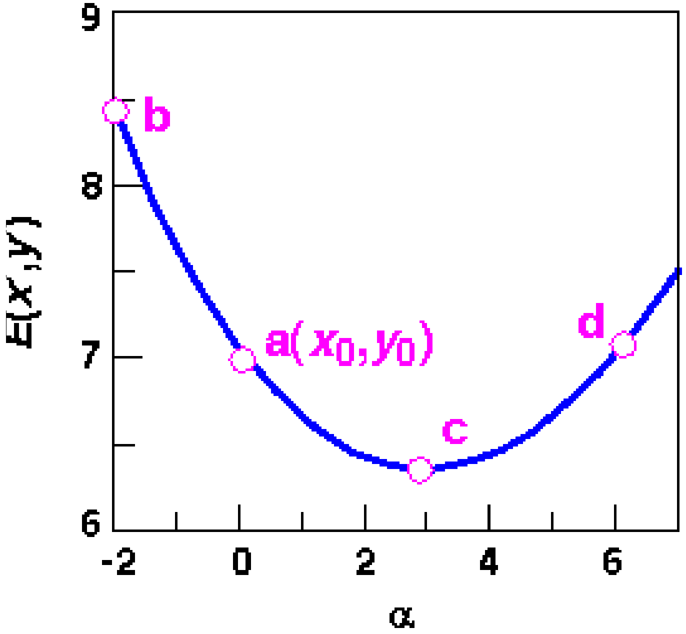
$$\vec{g}(\vec{x}) = \frac{\partial f}{\partial \vec{x}} = \mathbf{B}(\vec{x}-\vec{x}^0)$$

$$g_i(\vec{x}) = \frac{\partial f}{\partial x_i} = \sum_j \mathbf{B}_{ij}(x_j - x_j^0)$$

at the minimum the Hessian matrix must be additionally positive definite

Linear Search





$$(x', y') = \left(x_0 + \alpha \frac{\partial E}{\partial x}\bigg|_{x_0, y_0}, y_0 + \alpha \frac{\partial E}{\partial y}\bigg|_{x_0, y_0} \right)$$

The energy must be evaluated at 3-10 points to precisely locate the one-dimensional minimum

- □ The derivative vector from the initial point **a** (x_0, y_0) defines the line search direction.
- □ a line search amounts to a one-dimensional optimization along a direction vector determined at each iteration.

Newton algorithm

- start with an arbitrary start point $\vec{x}^1$
- calculate the gradient $\vec{g}(\vec{x}^1)$
- multiply with the inverse of the Hessian matrix and perform a step

$$\vec{x}^2 = \vec{x}^1 - \mathbf{B}^{-1}\vec{g}(\vec{x}^1)$$

  by inserting $\vec{g}(\vec{x}^1) = \frac{\partial f}{\partial \vec{x}} = \mathbf{B}(\vec{x}^1 - \vec{x}^0)$, one immediately recognises that $\vec{x}^2 = \vec{x}^0$ hence one can find the minimum in one step

- in practice, the calculation of **B** is not possible in a reasonable time-span, and one needs to approximate **B** by some reasonable approximation

Steepest descent

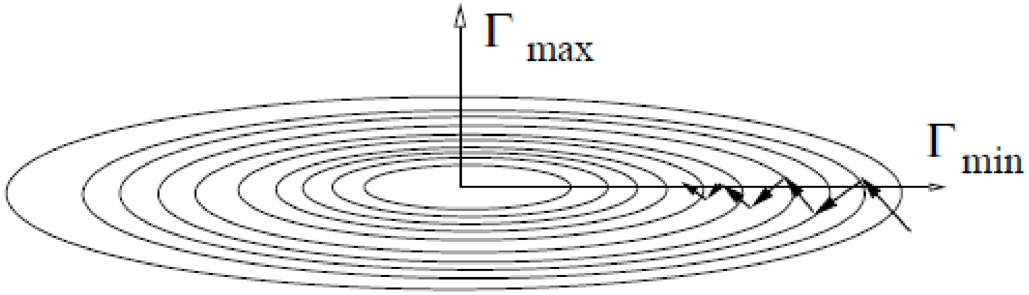
approximate **B** by the largest eigenvalue of the Hessian matrix → steepest descent algorithm (Jacobi algorithm for linear equations)

1. initial guess $\vec{x}^1$
2. calculate the gradient $\vec{g}(\vec{x}^1)$
3. make a step into the direction of the steepest descent

$$\vec{x}^2 = \vec{x}^1 - 1/\Gamma_{max}(B)\vec{g}(\vec{x}^1)$$

4. repeat step 2 and 3 until convergence is reached

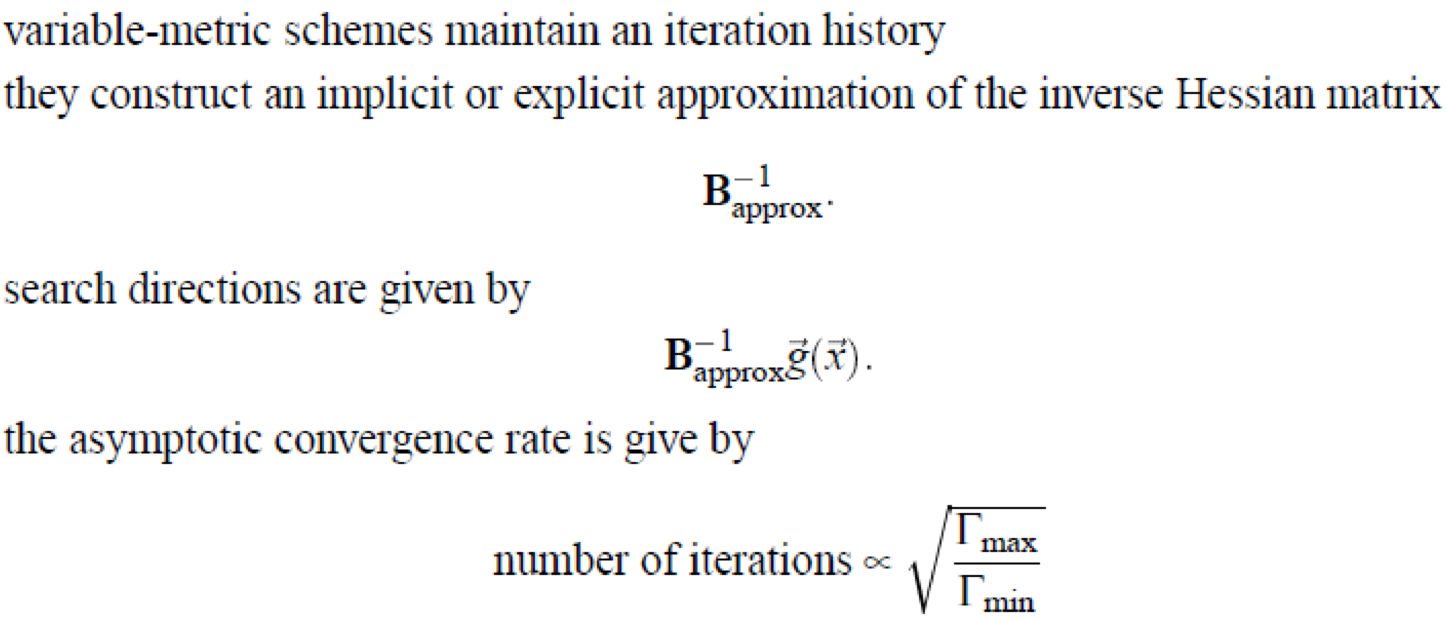
for functions with long steep valleys convergence can be very slow



Speed of convergence

- for ionic relaxation, the eigenvalues of the Hessian matrix correspond to the vibrational frequencies of the system
- the highest frequency mode determines the maximum stable step-width ("hard modes limit the step-size")
- but the soft modes converge slowest

Variable-metric schemes, Quasi-Newton scheme



variable-metric schemes maintain an iteration history
they construct an implicit or explicit approximation of the inverse Hessian matrix

$$\mathbf{B}^{-1}_{\mathrm{approx}}.$$

search directions are given by

$$\mathbf{B}^{-1}_{\mathrm{approx}}\vec{g}(\vec{x}).$$

the asymptotic convergence rate is give by

$$\text{number of iterations} \propto \sqrt{\frac{\Gamma_{\mathrm{max}}}{\Gamma_{\mathrm{min}}}}$$

Simple Quasi-Newton scheme, DIIS

direct inversion in the iterative subspace (DIIS)

- set of points

$$\{\vec{x}^i | i = 1, \ldots, N\} \quad \text{and} \quad \{\vec{g}^i | i = 1, \ldots, N\}$$

- search for a linear combination of x^i which minimises the gradient, under the constraint

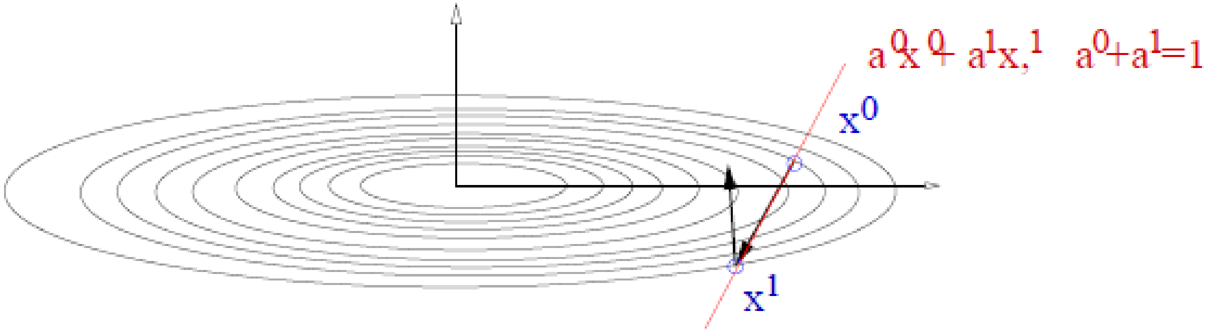
$$\sum_i \alpha_i = 1,$$

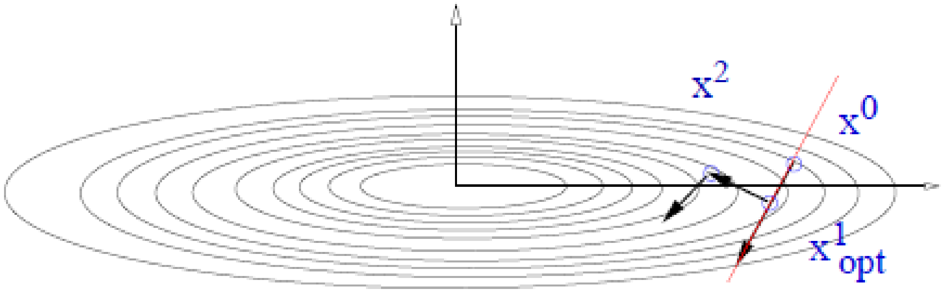
-

$$\begin{aligned} \vec{g}(\sum_i \alpha^i \vec{x}^i) &= \mathbf{B}\left(\sum_i \alpha^i \vec{x}^i - \vec{x}^0\right) = \mathbf{B}\left(\sum_i \alpha^i \vec{x}^i - \sum_i \alpha^i \vec{x}^0\right) \\ &= \sum_i \alpha_i \mathbf{B}(\vec{x}^i - \vec{x}^0) = \sum_i \alpha_i \vec{g}^i. \end{aligned}$$

gradient is linear in it's arguments for a quadratic function

1. steepest descent step from $\vec{x}^0$ to $\vec{x}^1$ (arrows correspond to gradients $\vec{g}_0$ and $\vec{g}_1$)
2. gradient along indicated red line is now know, determine optimal position $\vec{x}^1_{\mathrm{opt}}$
3. another steepest descent step form $\vec{x}^1_{\mathrm{opt}}$ along $\vec{g}_{\mathrm{opt}} = \vec{g}(\vec{x}^1_{\mathrm{opt}})$
4. calculate gradient $x^2 \Rightarrow$ now the gradient is known in the entire 2 dimensional space (linearity condition) and the function can be minimised exactly





Conjugate gradient

first step is a steepest descent step with line minimisation

search directions are "conjugated" to the previous search directions

1. gradient at the current position $\vec{g}(\vec{x}^N)$
2. conjugate this gradient to the previous search direction using:

$$\vec{s}^N = \vec{g}(\vec{x}^N) + \gamma \vec{s}^{N-1} \qquad \gamma = \frac{(\vec{g}(\vec{x}^N) - \vec{g}(\vec{x}^{N-1})) \cdot \vec{g}(\vec{x}^N)}{(\vec{g}(\vec{x}^{N-1})) \cdot \vec{g}(\vec{x}^{N-1})}$$

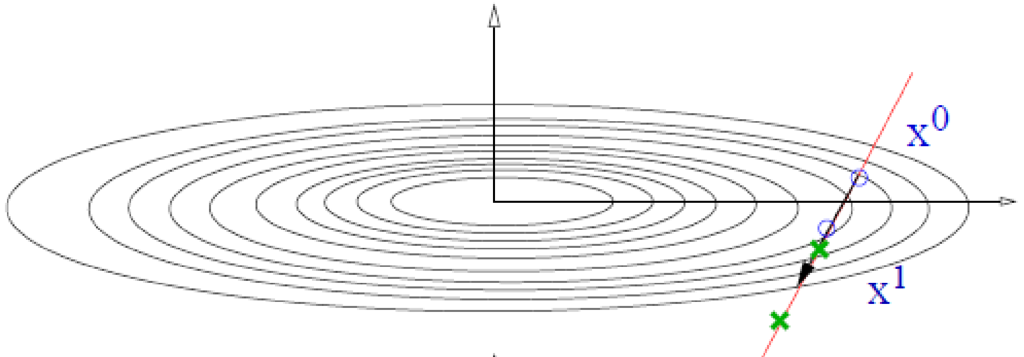
3. line minimisation along this search direction $\vec{s}^N$
4. continue with step 1), if the gradient is not sufficiently small.

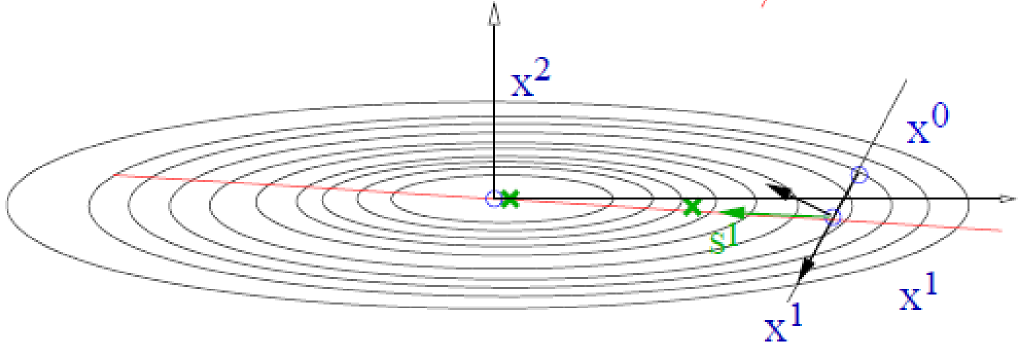
the search directions satisfy:

$$\vec{s}^N \mathbf{B} \vec{s}^M = \delta_{NM} \qquad \forall N, M$$

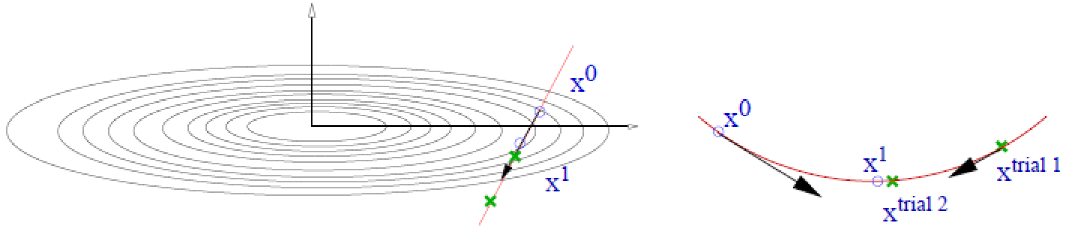
the conjugate gradient algorithm finds the minimum of a quadratic function with k degrees of freedom in $k+1$ steps exactly

1. steepest descent step from $\vec{x}^0$, search for minimum along $\vec{g}_0$ by performing several trial steps (crosses, at least one triastep is required) → $\vec{x}^1$
2. determine new gradient $\vec{g}_1 = \vec{g}(\vec{x}_1)$ and conjugate it to get $\vec{s}_1$ (green arrow)
   for 2d-functions the gradient points now directly to the minimum
3. minimisation along search direction $\vec{s}_1$





- CG requires a line minisations along the search direction



Damped molecular dynamics

instead of using a fancy minimisation algorithms it is possible to treat the minimisation problem using a simple "simulated annealing algorithm"

- regard the positions as dynamic degrees of freedom
- the forces serve as accelerations and an additional friction term is introduced
- equation of motion ($\vec{x}$ are the positions)

$$\ddot{\vec{x}} = -2 * \alpha \vec{g}(\vec{x}) - \mu \dot{\vec{x}},$$

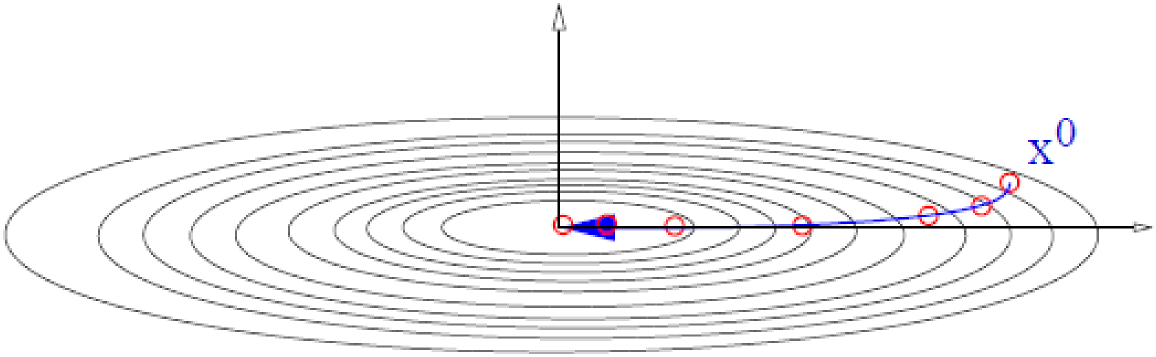
using a velocity Verlet algorithm this becomes

$$\vec{v}_{N+1/2} = \left((1 - \mu/2) \vec{v}_{N-1/2} - 2 * \alpha \vec{F}_N \right) / (1 + \mu/2)$$

$$\vec{x}_{N+1} = \vec{x}_{N+1} + \vec{v}_{N+1/2}$$

for $\mu = 2$, this is equivalent to a simple steepest descent step

- behaves like a rolling ball with a friction
  it will accelerate initially, and then deaccelerate when close to the minimum
- if the optimal friction is chosen the ball will glide right away into the minimum
- for a too small friction it will overshoot the minimum and accelerate back
- for a tool large friction relaxation will also slow down (behaves like a steepest descent)



Asymptotic convergence rate

- asymptotic convergence rate is the convergence behaviour for the case that the degrees of freedom are much large than the number of steps

  e.g. 100 degrees of freedom but you perform only 10-20 steps

- how quickly, do the forces decrease?

- this depends entirely on the eigenvalue spectrum of the Hessian matrix:

  – steepest descent: $\Gamma_{\max}/\Gamma_{\min}$ steps are required to reduce the forces to a fraction ε

  – DIIS, CG, damped MD: $\sqrt{\Gamma_{\max}/\Gamma_{\min}}$ steps are required to reduce the forces to a fraction ε

$\Gamma_{\max}, \Gamma_{\min}$ are the maximum and minimal eigenvalue of the Hessian matrix

Methodology for optimization calculations

Prerequisites

One of the most important steps: properly preparing the structure to be simulated.

It is impossible to provide a single recipe for a successful model

Considerations

When to use constraints and restraints.

When to use different algorithms.

What criteria to use for judging convergence of the optimization.

Applying constraints and restraints

Constraints allow you to restrict a calculation to the region or conformation of interest in a molecule

Fixed constraints

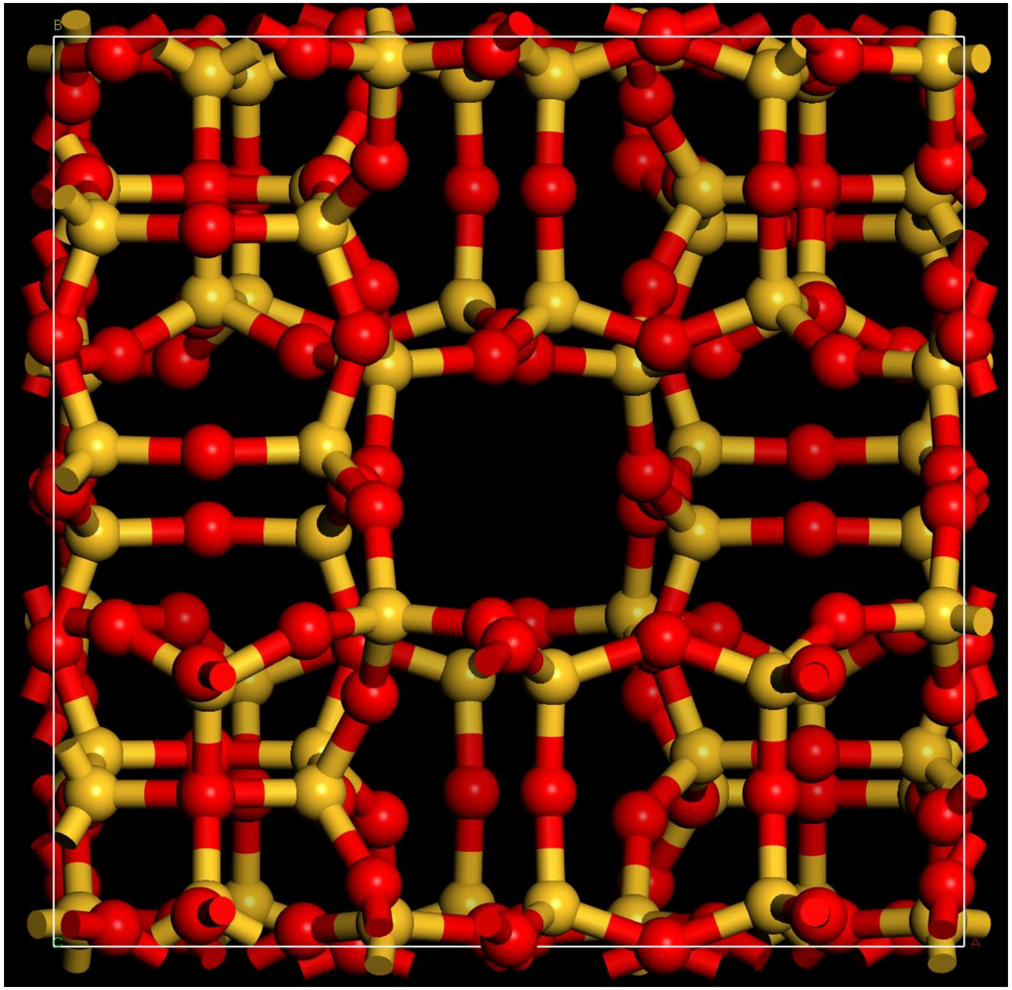
Fixed atoms or beads are constrained to a given location in space, so they cannot move at all

Restraints

A *restraint* is an energetic bias that tends to force the calculation toward a certain restriction but not requiring that it is met absolutely as in the case of constraints

1. **Distance restraints**
2. **Angle restraints**
3. **Torsion restraints**

Examples in Molecules and Solids Calculations



MS Examples

When to use different algorithms

Structure size and distance from the minimum

- □ The conjugate gradient and steepest descents methods can be used with models of any size.
- □ Most Newton methods cannot be used with very large models, need sufficient disk space to store a second-derivative matrix.

(Newton, Conjugated gradient algorithms …) assumes surface is quadratic
Newton method : sensitive to the surface due to the inverse of Hessian matrix

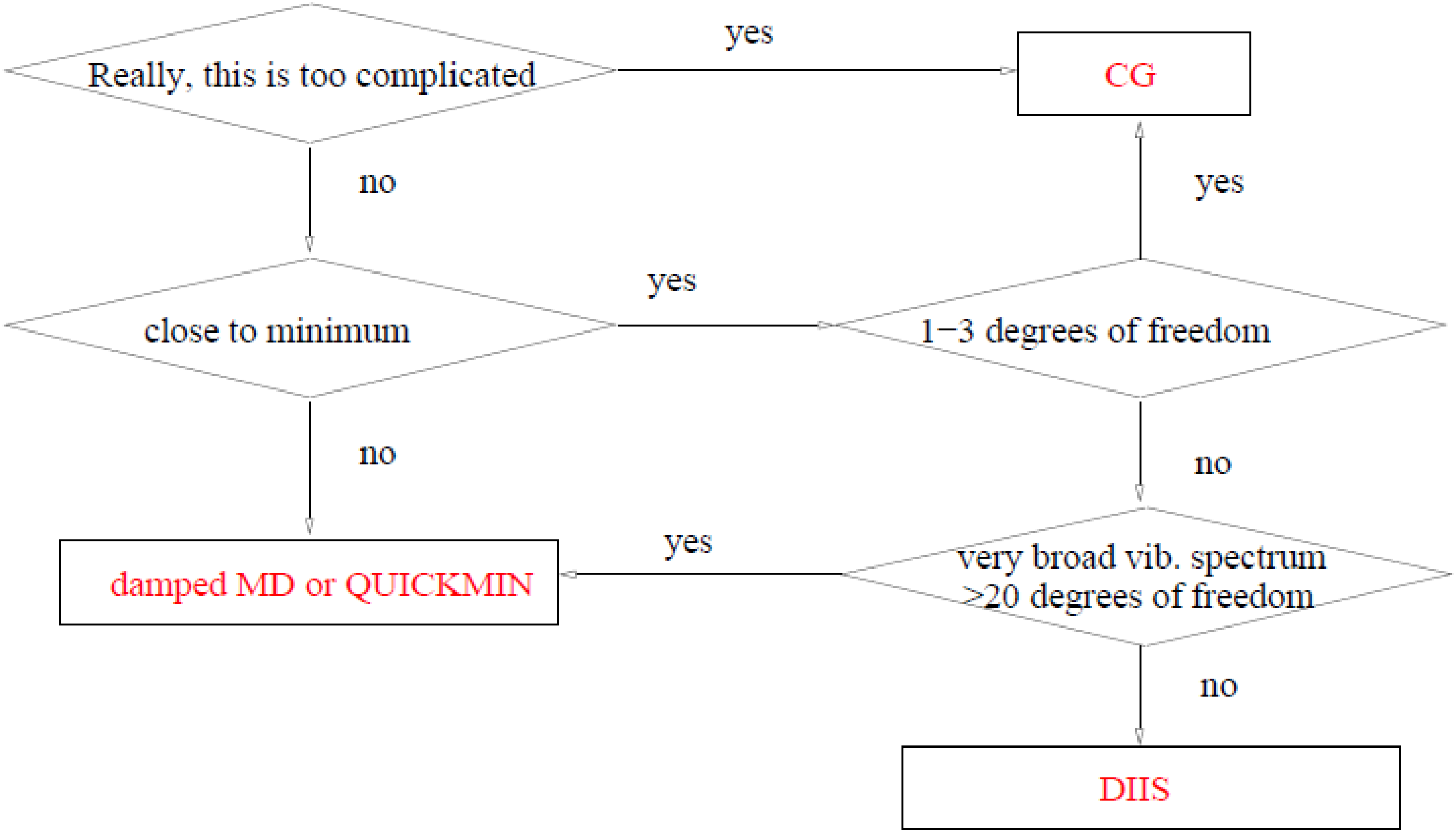
a general rule, steepest descents is often the best optimizer to use for the first 10-100 steps, after which the conjugate gradient and/or a Newton optimizer

When to use different algorithms

Starting structures and choice of force field

Pre-procedure with a highly distorted system (examples with MS)

decision chart



Storage requirement for different algorithms

Algorithm	Variant	Memory needed for	Scales as[1]
Steepest descents		First derivatives	$3N$
Conjugate gradient	Polak-Ribiere	First derivatives, gradient from previous iteration	$3N$
	Fletcher-Reeves	First derivatives, gradient from previous iteration	$3N$
	Powell	First derivatives, gradient from previous iteration	$3N$
Newton-Raphson	Full, iterative	Hessian, eigenvectors	$(3N)^2$
	BFGS	First derivatives, Hessian update, scratch vectors	$(3N)^2$
	DFP	First derivatives, Hessian update, scratch vectors	$(3N)^2$
	Truncated	Hessian	$(3N)^2$

General storage requirements of optimization algorithms

[1] N=number of atoms (number of degrees of freedom).

Bad Conditions

- the convergence speed depends on the eigenvalue spectrum of the Hessian matrix
  - larger systems (thicker slabs) are more problematic (acoustic modes are very soft)
  - molecular system are terrible (week intermolecular and strong intramolecular forces)
  - rigid unit modes and rotational modes can be exceedingly soft

  the spectrum can vary over three orders of magnitudes ⇒ 100 or even more steps might be required ionic relaxation can be painful

- to model the behaviour of the soft modes, you need very accurate forces since otherwise the soft modes are hidden by the noise in the forces

  `EDIFF` must be set to very small values (10^{-6}) if soft modes exist

Convergence criteria

□ **Mathematical definition**

Mathematically, a minimum is defined as the point at which the derivatives of the function are zero and the second-derivative matrix is positive definite. Numerical calculations.

□ **Application to chemical models**

In a molecular optimization, the atomic derivatives may be summarized as an average, a root-mean-square (rms) value, or the largest value. An rms derivative is a better measure than the average, because it weights larger derivatives more should always check that the maximum derivative is not unreasonable.

□ **How close to absolute convergence is good enough?**

simply want to relax overlapping atoms before beginning a dynamics run, minimizing to a maximum derivative of 1.0 kcal mol^{-1} Å^{-1} is usually sufficient.

to perform a normal mode analysis, the maximum derivative must be less than 10^{-5}

□ **Local or global minimum?**

□ **Other termination criteria**

Generally, set a maximum number of iterations. That is, the run ends when either the convergence criteria or the maximum number of iterations is reached, whichever occurs first.

Optimization of Molecule

- Building initio structure
- Symmetry?
- Some parameters control the Convergence tolerance of geometry optimization

  Max Iterations, Max Step Size, Energy, Max. Force, Max Displacement …
- Other parameter: charge, strain, functional of DFT ...
- Step by Step Process

  coarse, medium, fine … … force-field, molecular mechanic, first-principles

Examples …

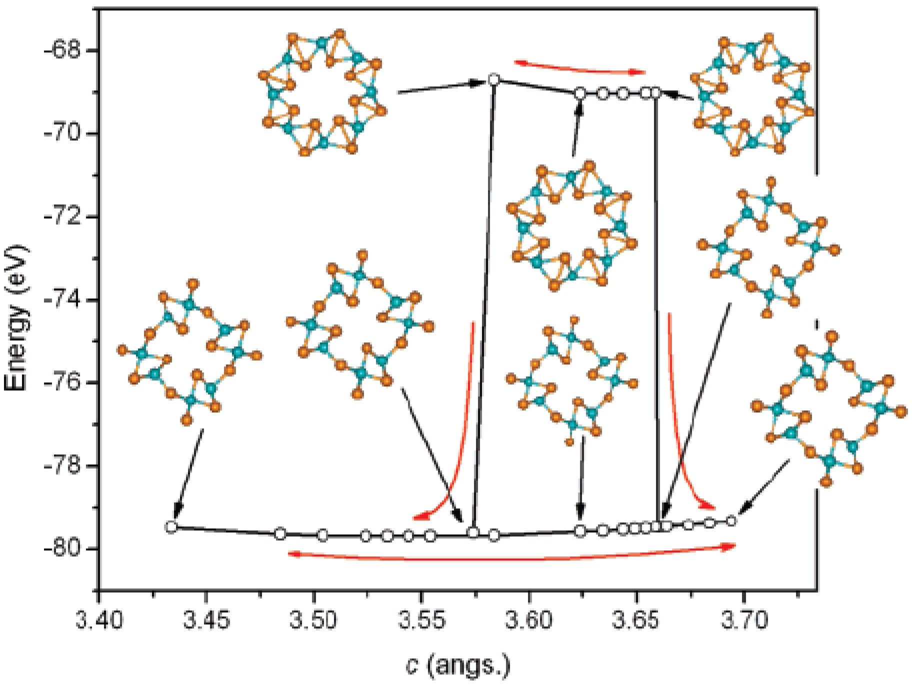
Optimization of 1D nanotube

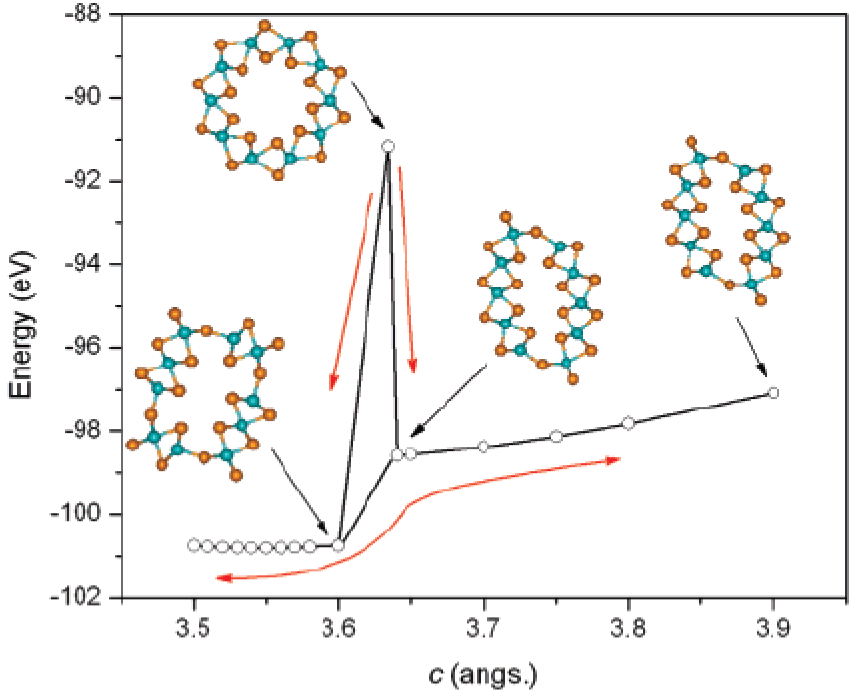
Two tasks: optimization of lattice constants and atomic positions.

1D system: Scan of lattice constant “c” and optimize structure at each point of “c”

Examples: optimization of 1D carbon nanotubes

Example of 1D $MoTe_2$ Nanotubes





Possible distortion of $MoTe_2$ nanotubes

Optimization of 3D crystals

Optimization of lattice constants (a, b, c, alpha, beta, gamma) and atomic positions

Symmetry?

a=b=c, alpha=beta=gamma=60

Examples: Diamond